Help Contents

Essentials and new feature summary for getting up to speed with Quattro Pro.

Notebook Tasks for using standard spreadsheet features, plus data analysis, databases, and Workgroup Desktop (optional).

Graph Tasks for using all graph, drawing, and slide show features.

Application Building Tasks for creating custom applications.

@Functions and formula reference.

Macros programming reference.

Menu Commands reference.

Objects and Properties reference.

Essentials

Quattro Pro for Windows differs from earlier spreadsheet applications in several key areas. Before attempting any tasks, read the following topics in this Help system. For a step-by-step introduction to Quattro Pro, choose Interactive Tutors from the Quattro Pro Help menu.

New Features	Describes what's new in version 5.0 of Quattro Pro for Windows.
Using Quattro Pro Help	How to use Quattro Pro Help and Object Help for SpeedBar button identification.
Interactive Tutors	Interactive lessons that work with your data.
Experts	An alternate way to perform certain spreadsheet tasks.
Using Notebooks	Use Notebooks to keep track of multiple sets of spreadsheet data.
Object Inspector	Provides direct access to the properties of each object.
Work Areas	Explains each major part of Quattro Pro.
Quattro Pro Graphs	Important background information for working with graphs.
Mouse Techniques	Teaches mouse skills that boost your productivity.
Keyboard Techniques	Lists shortcut key combinations.

Undoing Mistakes	Specifies those areas of Quattro Pro where you can use the Undo command.
DOS Spreadsheet Users	For users of DOS versions of Quattro Pro and Lotus 1-2-3.

Notebook Tasks

The Notebook window is the background for most of your work in Quattro Pro; for an introduction to notebooks, see the Essentials topic Using Notebooks. To learn about parts of the screen in each of Quattro Pro's work areas, use the mouse and Ctrl+right-click to display Object Help for each part of the screen.

Entering Data	Entering text, formulas, and dates; referencing data blocks.
Editing	Changing and deleting data, rows, columns, and notebook pages, and naming data blocks.
Formatting Blocks and Pages	Displaying and changing block and page properties.
Using Files	Opening, closing, and saving notebook files.
Using Quattro Pro Windows	Manipulating windows and panes within Quattro Pro.
Printing	Printing notebooks and graphs; defining page layouts; setting up printers and print settings.
Advanced Editing	Using advanced features to modify blocks of cells (including grouping pages, DDE, OLE, and maintaining block names).
Linking Notebooks	Working with groups of pages and links between notebook files.
Data Analysis	Using advanced tools to analyze data structures.
Using Databases	Setting up, editing, and querying databases in notebooks and other applications.
Importing and Exporting Data	Importing, exporting, parsing, combining, and extracting files.
Setting Global Properties	Displaying and changing application and notebook properties.
Building Custom SpeedBars	Using the SpeedBar Designer to create new Speedbars quickly.
Workgroup Desktop Topics	Using the Workgroup Desktop to exchange notebooks (Workgroup edition only).

See Also

Graph Tasks

Application Building Tasks

Menu Commands
Objects and Properties

Graph Tasks

You can build graphs from your spreadsheet data, then display them on the spreadsheet, in a separate window, in a slide show, or send them to a printer or slide service. If you're new to Quattro Pro graphics, read Graph Basics. Use Ctrl+right-click to learn about parts of the screen in Quattro Pro's graph work areas.

Building Graphs	Creating numeric and text graphs from spreadsheet data.
Customizing Graph Properties	Displaying and changing graph properties; controlling the appearance of graph markers, grids, and text.
Enhancing Graphs	Using the drawing tools, creating slide shows, and importing/exporting graphics.
Analytical Graphing	Creating graphs from calculations based on spreadsheet data.

See Also

Notebook Tasks

Application Building Tasks

Menu Commands

Objects and Properties

Using Databases

Quattro Pro includes conventional spreadsheet database features and provides direct access to the power of full-fledged database applications with the Database Desktop. Click one of the topics below for spreadsheet database information or click Database Desktop Help to open its help file.

Step-By-Step Directions

Menu Commands

See Also

Application Building Tasks

You build user interfaces for custom applications in Quattro Pro's dialog window, displayed with Tools| UI Builder. Click one of the topics below for a list of subtopics. Press Ctrl and right-click on each part of the dialog window to learn about that part.

Step-By-Step Directions

Application Basics

Building Dialog Boxes and SpeedBars

Building Menus

Menu Commands

Dialog Menu

Property Menu

Objects (Controls) and Properties

Dialog Window Objects

Properties in the Dialog Window

See Also

Custom SpeedBars

Notebook Tasks

Graph Tasks

Menu Commands

Objects and Properties

@Function Descriptions

@Functions perform special calculations. @Function descriptions can be viewed in an alphabetically-sorted index, or in a series of lists for each major category:

Function Index

Step-By-Step Directions

Using @Functions

@Function Arguments

@Function Categories

Database	Perform analytic calculations on records in a database.
Date and Time	Calculate dates and times, or perform calculations involving business days.
Engineering	Perform number conversion, Boolean operations, and calculations involving complex numbers or Bessel functions.
Financial	Calculate investments and cash flow.
Logical	Compare and evaluate relationships between values.
Mathematical	Calculate trigonometric functions and other mathematical values.
Miscellaneous	Provide current information such as table values.
Statistical	Perform analytic calculations on a list of values.
String	Manipulate text strings or labels.

Macro Command Descriptions

A "macro" is a stored series of keystrokes, special keys, and commands that can be played back. You create macros by recording them or typing them in. Macro command descriptions can be viewed in an alphabetically-sorted index, or in a series of lists for each major category.

Macro Command Index

Step-By-Step Directions

Using Macros

Macro Command Categories

Keyboard	Emulate the action of various keys on the keyboard.
Screen	Affect the display.
Interactive	Let you create macros that display dialog boxes or pause for the user to enter data from the keyboard.
Program Flow	Let you branch and loop in a macro.
Cell	Affect the data stored in specified cells.
File	Work with data within files other than the active notebook file.
Command Equivalents	Emulate operations you can perform with menu commands.
DDE	Communicate with other Windows applications.
UI Building	Let you change the menu bar.
Object	Let you create, move, resize, select, and change the properties of objects.
Miscellaneous	Insert characters and comments, and perform a variety of other tasks.
Analysis Tools	Let you perform analyses of numerical data.

/ Commands are commands used by Quattro Pro for DOS to emulate menu commands. See the Quattro Pro for DOS User's Guide for more information on menu-equivalent commands.

Keyboard Techniques

Throughout Quattro Pro documentation, instructions emphasize the use of a mouse. The following topics list Quattro Pro keyboard commands:

Special Keys	Useful keyboard shortcuts.
Edit Mode Keys	When Quattro Pro is in edit mode.
Point Mode Keys	When Quattro Pro is in point mode.
Print Preview Keys	When previewing a notebook before printing.
Navigation Keys	For moving around notebook pages.
Function Keys	Keys labeled F1, F2, through F12.

See Also
Mouse Techniques

Menu Commands

File	Work with files, print, exit Quattro Pro.
Edit	Clipboard operations (cut, copy, paste); undo, goto, search.
Block	Move, copy, fill, name, transpose, and reformat blocks; insert and delete rows, columns, pages; copy only values of blocks.
Data	Sort and query records, parse strings into fields, set up a data entry form, perform what-If and frequency calculations, access external databases.
Tools	Use macros; group pages; combine, extract, and import files; update links; perform advanced math calculations; check spelling (optional); run Optimizer and Solve For; display Scenario Manager and Consolidator SpeedBars; and start SpeedBar designing and UI building.
Graph	Create and display graphs and slide shows.
Draw	Add drawn objects and bitmaps to graphs and slides.
SpeedBar	Work with custom SpeedBars in the SpeedBar Designer.
Dialog	Work with custom SpeedBars and dialog boxes in the UI Builder.
Property	Alter properties associated with each type of object.
Window	Rearrange window display.
Help	Start Help or view version information and current memory usage.

Objects and Properties

See Object Inspector Menus for an introduction to working with objects and properties in Quattro Pro.

Basic Properties

Active Block

Active Page

Active Notebook

Application

Properties in Each Window

Properties in the Notebook Window

Properties in the Graph Window

Properties in the Dialog Window

Objects You Can Inspect

Notebook Window Objects

Graph Window Objects

Dialog Window Objects

New Features

The following sections introduce the new features in version 5.0 of Quattro Pro for Windows:

Quattro Pro Work Areas

Quattro Pro has five work areas:

Notebook window	is where you enter and display data. It contains all standard spreadsheet features along with many enhancements. This is where you will do most of your work in Quattro Pro.
Graph window	is where you can build and customize graphs based on data from the Notebook window.
Dialog window	is where you can build custom applications that run within Quattro Pro
Graphs page	is where you link graphs together to make slide shows, or link dialog boxes you have created to build a custom application. The Graphs page is the last page in each notebook.
Print Preview screen	is where you preview your work before printing.

See Also

Notebooks

Object Inspector Menus

Mouse Techniques

Using Notebooks

Notebooks are a new concept in spreadsheet products. They provide a way to organize many spreadsheets together in the same file.

A notebook is a collection of 256 spreadsheet pages and a Graphs page, which is the last page. Each spreadsheet page is a grid made up of columns and rows. The Graphs page contains icons, each representing a graph, slide show, or dialog box you've created.

Each notebook is saved as its own file. The default file name for the first notebook is NOTEBK1.WB1.

There are three major ways to take advantage of notebooks:

- by breaking up a spreadsheet into small pieces on separate pages
- by gathering logically-related data into the same file
- by consolidating similarly-formatted spreadsheets into the same file

Breaking Up a Large Spreadsheet

If you're translating a large spreadsheet from Quattro Pro for DOS or another application, you can make it much easier to work with by breaking it up into separate pieces for each page.

To reach an individual page, you can click the page's tab (this is easier than scrolling to different parts of one large spreadsheet). Also, when you write a formula that refers to cells on another page, the page name appears in the formula, making it easier to see what you're referencing.

Gathering Related Data

Instead of saving a budget, a schedule, an inventory, or other related information in different files on disk, you can make them separate pages in the same notebook. This gives you one file name to remember, not many.

By default, notebook pages are labeled A through IV, but you can give them descriptive names to remind you of their contents. See Naming a Page for details.

Consolidating Data

If you're working with data that conforms to a given template or layout, notebooks give you efficient ways to enter and format data. Grouping the pages together before entering standard column heading information or before making formatting changes, for example, speeds up your work. See Grouping Notebook Pages for details.

Moving Around Within a Notebook

To move to a different page in a notebook, click its tab. If its tab isn't in view, use the tab scroller to reveal additional tabs. The tab scroller works like those in most Windows applications.

To move quickly to the last page in a notebook (the Graphs page), click the SpeedTab button; it is located immediately to the right of the page tabs, and has a horizontal arrow on its surface. The Graphs page contains icons representing each of the graphs, slide shows, and dialog boxes you've created in the notebook. To switch back to the last active spreadsheet page, click the SpeedTab button again. Notice that the arrow on the SpeedTab button changes direction depending on whether the Graphs page is active or not.

After you click the tab of the page you want, you can use the scroll bars to move to different parts of the page with the scroll bars.

See Also

Object Inspector Menus

Mouse Techniques

Object Inspector Menus

Object Inspector menus provide a quick way to display all of the changes that can be made to individual objects. You just point to most objects and press the right mouse button (right-click) to display a SpeedMenu. Choose its Properties command to list the object's properties. (For window title bars, the Object Inspector appears as soon as you right-click.)

An Object Inspector is available for these objects:

- blocks
- pages
- notebooks
- graphs in windows, and their elements, such as bars, axes, or text boxes
- floating objects, such as graphs, graphic images, or SpeedButtons that appear in a layer above spreadsheet cells
- dialog boxes you create and their elements, such as radio buttons or edit fields
- the Quattro Pro application itself

Each of these objects has properties, which are characteristics particular to that type of object. For example, blocks have a Font property that can be set to Bold, so the text of entries in that block appear in boldface type. One property of a page is the name that appears on its tab. Each notebook has its own Palette property for controlling the colors available. Quattro Pro's system defaults, such as the default storage directory or file-name extension, are application properties.

To change the properties of an object, right-click it and choose the Properties command. A different Object Inspector menu appears depending on the type of object you right-click. From the left side of the Object Inspector, choose the property you want to change. The options displayed on the right change to correspond to the chosen property. Click Active Block Object Inspector for details.

Next, choose settings for the current property. You can go on to change other properties for the current object. To help you track changes, the property name turns blue if you change its setting. If the Object Inspector has an example box, it shows the result of your choices. When you're finished, choose OK.

You can also change properties of an object by using the Property menu. Property|Current Object displays the Object Inspector for the currently selected object. If no object is currently selected, the properties that control the active window are displayed. Property|Application displays the application Object Inspector. Other commands, such as Active Notebook or Active Page, are available depending on whether a notebook, graph, or dialog window is active.

To see where to right-click for the most common objects in each Quattro Pro window, see Locating Objects.

See Also

Notebooks

Mouse Techniques

Locating Objects

Different properties are available in each Quattro Pro window. Choose the window you are working in:

Notebook Window	The standard Quattro Pro window
Graph Window	Where you edit a graph
Dialog Window	Where you create custom dialogs

Quattro Pro Graphs

Using graphs to analyze your data is faster and easier than examining the data cell-by-cell. A graph presents a set of data as a picture. It may uncover a trouble spot, display a trend, or illustrate a correlation between categories of data in your spreadsheet.

A single graph can appear in three places in Quattro Pro: as a floating graph in a notebook window, in a separate graph window, and as an icon on the Graphs page. Graphs always exist on the Graphs page and in a graph window; only floating graph display is optional.

In a Notebook Window:

Create a floating graph directly on a spreadsheet page when you want to view or print your graph on the same page as your data. The Notebook window SpeedBar has a tool that creates floating graphs. You can also add titles to the graph, change the graph type, or redefine the data series plotted in the graph in this window. For all other changes, you must double-click the floating graph, or use Graph| Edit, to display the graph in a graph window. (The floating graph updates automatically as you modify the graph in a graph window.)

In a Graph Window:

Create a graph in its own graph window when you want the graph to appear in a slide show or on a printed page, but not on a spreadsheet. Every graph (including floating graphs) has its own window. Object Inspectors for graph elements are available ONLY when you display a graph in its graph window. You can also enhance graphs using the drawing tools and color palette on the graph window SpeedBar. The graph window Draw menu lists commands you need to rearrange drawn elements and import and export graphics.

On the Graphs Page:

The Graphs page displays an icon for each graph in the notebook. You can right-click an icon to rename a graph, delete or cut an icon to delete a graph, or copy and paste an icon through the Windows Clipboard to copy a graph. You can also print a series of graphs, or create an onscreen slide show with special effects. To see the Graphs page, click the SpeedTab button (which looks like an arrow) at the bottom of the notebook window.

Mouse Techniques

Quattro Pro for Windows fully supports standard Windows mouse functions, and also features several highly useful enhancements. Use the mouse as follows:

To select a:	
cell	Click the cell.
block	Drag it by clicking a cell in one corner, holding down the left mouse button, moving to the opposite corner and releasing the mouse button. Or click one corner, hold down the Shift key, and click the opposite corner.
noncontiguous block	Hold down the Ctrl key while you drag blocks.
row	Click the corresponding row number in the border.
column	Click the corresponding column letter in the border.
page	Click the Select-All box at the intersection of the row and column borders.
cell on another page	Click the page tab, then click the cell.
3-D block	First select the 2-D block in the first page in the 3-D block, then hold down the Shift key while clicking the last page in the 3-D block. A black line appears under the tabs of the selected pages.

To change object properties:
Right-click an object and choose the Properties command to display its Object Inspector. This lets you change many settings at once, rather than choosing commands one by one.

To move a block:
Select a block of cells, release the left mouse button, then left-click, hold down the button for a moment until the hand icon appears, and drag the selected block to move it to another location. This Drag and Drop procedure has the same result as the Block|Move command.

To copy a block:
Select a block of cells, release the left mouse button, then hold down the Ctrl key and, at the same time, left-click, hold down the button until the hand icon appears, and drag the selected block to another location. This Drag and Drop procedure has the same result as the Block|Copy command.

SpeedBars
Quattro Pro includes SpeedBars which allow you to bypass menu commands by clicking buttons. Each Quattro Pro window has its own unique SpeedBar; for a definition of each SpeedBar button and other parts of the screen, display Object Help.

You can choose to open and close SpeedBars (see Opening and Closing SpeedBars).

See Also
Object Inspector Menus

Notebooks

Undoing Mistakes

The Edit|Undo command lets you reverse most kinds of operations after you've carried them out. For example, if you make an entry in a cell, then change your mind and want to remove it, choose Edit|Undo immediately after making the entry. The entry will be removed and whatever was previously entered in the cell will be returned.

You can identify your last action by the wording of the Undo command. For example, after making a cell entry, the Undo command reads "Undo Entry." After changing the Zoom Factor property in the notebook Object Inspector, the Undo command reads "Undo Notebook Property."

If you change your mind again after using Undo and want to reinstate the first change, choose Edit|Redo. As with Undo, Redo specifies the type of operation available to be redone.

Some actions are undoable only if you enable Undo in the application Object Inspector. You can identify these actions because the Undo command is dimmed immediately after the action is taken. In all situations except where program speed and available memory is absolutely crucial, it's recommended that you keep Undo enabled.

To enable Undo for the maximum variety of actions,

1. Move the mouse pointer to the Quattro Pro window title bar and click the right mouse button.
2. Choose Startup.
3. Check Undo Enabled in the Options box, and choose OK.

See Also

Object Inspector Menus

Mouse Techniques

Notebooks

DOS Spreadsheet Users

If you have used DOS-based versions of Quattro Pro or 1-2-3, read the following topics for guidelines on using files and macros created with those DOS spreadsheets. Be sure to read the other Essentials topics for a description of major new product features.

Alternative Menus

File Compatibility

Macro Compatibility

See Also

Notebooks

Object Inspector Menus

Mouse Techniques

Alternative Menu Systems

When you're first using Quattro Pro for Windows, you may want to load an alternative menu system. Display the Application properties and choose Macro and then Slash Key to display a list of available menu systems (see Application Macro Property for more information). Using an alternative menu system can help you maintain productivity while you learn the program, but using the default Quattro Pro for Windows interface makes it easier to access all commands and features.

See Also

File Compatibility

Macro Compatibility

File Compatibility

You can load files created with DOS versions of Quattro Pro, 1-2-3 (versions 2.x and 3.x), Excel, and several databases from within Quattro Pro. Just retrieve the file as you would any other; Quattro Pro automatically identifies the file type (by its extension or the first record in the file) and translates it. You can run most macros contained in the file.

To save a 1-2-3 file as a Quattro Pro file, just change the file's extension when you save it; Quattro Pro for Windows translates the file. The same is true for files in other formats.

If you don't include a file-name extension, Quattro Pro for Windows automatically adds the default file extension .WB1.

1-2-3 files

When loading a 1-2-3 file, linking is converted to Quattro Pro's syntax.

Here is an example:

Quattro Pro syntax	**1-2-3 2.x syntax**
[filename]celladdress	<<filename>>celladdrs
`+[BUDGET.WK1]B4`	`+<<BUDGET.WK1>>B4`

See Also

Alternative Menu Systems

Macro Compatibility

Macro Compatibility

Macros using Quattro Pro for DOS menu equivalents are still compatible. For example, the statement {/Graph;Type} still executes as Graph|Graph Type.

See Also
Quattro Pro for DOS Macros

Graph Expert, Step 1

This dialog box lets you choose data to graph. If you include column and row labels in the graph data block, they appear in your graph as x-axis and y-axis labels.

Quattro Pro chooses a graph type for your data, based on the number of rows and columns and the total number of data points in the graph data block. You'll have a chance to change the graph type later.

If plotted data is hidden in the graph, you may need to swap rows and columns or reverse the data series. Try checking the check boxes at the bottom of the right pane in Graph Expert, Step 1. See if the graph in the left pane looks better when you make a change. If not, change it back.

You can choose Tip to switch between the graph and an explanation pane.

Graph Expert, Step 2

This dialog box lets you choose a main graph type:

Certain graphs are best suited for plotting certain types of data:

Bar graphs	compare values of different items at specific points in time--to contrast monthly commissions for each sales representative, for example. Rotated bar graphs are plotted horizontally instead of vertically.
Line graphs	show the progression of values over time--to track sales, for example. Area graphs are like filled-in line graphs.
Pie graphs	compare individual values to other values and to the whole--how yearly expenses break down into categories, for example. Use them to focus on the individual values in a single series. Doughnut graphs, available through this type choice, are like pie graphs with a hole in the middle.
Specialty graphs	include high-low graphs, radar graphs, and combination graphs.

Choose Expert's Choice if you want the Graph Expert to choose an appropriate type for your graph.

Graph Expert, Step 3: Bar Graph

This dialog box offers a number of bar graph choices:

Bar graphs	compare values of different items at specific points in time--to contrast monthly commissions for each sales representative, for example.
Stacked bar graphs	show the relationship of each value to the total--how total sales are divided between regions, for example. 100% stacked bar graphs show the percentage each series contributes to the total.
Comparison graphs	have lines connecting the boundaries between series. This makes it easier to compare series values or proportions.
Multiple column graphs	plot each series as a separate sub-graph.

Choose a type and click Next Step to add titles.

Graph Expert, Step 3: Rotated Bar Graph

Rotated graphs are like standard graphs, but they are plotted horizontally instead of vertically. Choose from among these types of rotated graphs:

Bar graphs	compare values of different items at specific points in time--to contrast monthly commissions for each sales representative, for example.
Stacked bar graphs	show the relationship of each value to the total--how total sales are divided between regions, for example. 100% stacked bar graphs show the percentage each series contributes to the total.
Comparison graphs	have lines connecting the boundaries between series. This makes it easier to compare series values or proportions.

Choose a type and click Next Step to add titles.

Graph Expert, Step 3: Pie Graph

This dialog box lets you choose from among these graph types:

Pie graphs	compare individual values to other values and to the whole--how yearly expenses break down into categories, for example. Use them to focus on the individual values in a single series.
Doughnut graphs	like pie and column graphs, plot a single series with each value plotted as a percentage of the whole. You can add text or graphics to the interior "hole" with graph window drawing tools.
Multiple pie graphs	plot each series as a separate sub-graph.

Choose a type and click Next Step to add titles.

Graph Expert, Step 3: Line or Area Graph

This dialog box lets you choose from among various line, area, and surface graphs:

Line graphs	show the progression of values over time--to track sales, for example.
Area graphs	show the relationship of each value to the total over time--how each sales representative contributed to total sales over a 12 month period, for example.
Surface graphs	plot rows and columns as intersecting lines on a surface that is suspended in a 3-D frame. Surface graphs are useful for plotting functions such as f(x) and f(x,y), and parametric curves (x(t), yt(t)).

The graph in the upper right corner is an XY graph. It only plots data if the first column of your graph data block contains dates.

Choose a type and click Next Step to add titles.

Graph Expert, Step 3: Specialty Graph

This dialog box lets you choose from among these graph types:

Radar graphs	show x-axis values as imaginary lines radiating from a common center, like spokes of a wheel, with y-axis values plotted on each "spoke." They are useful for highlighting trends, depending on the shapes drawn by the plot lines, or simplifying series comparisons.
High-low graphs	illustrate the difference between corresponding values in two series. Though most often used in tracking daily stock prices, high-low graphs can be used whenever you want to compare the difference between pairs of values.
Combination Graphs	combine different types of graphs in one -- lines with bar, for example.

Choose a type and click Next Step to add titles.

Graph Expert, Step 4

This dialog box lets you add titles to your graph. The main title and subtitle appear above the graph. The y-axis title appears along the left side and the x-axis title runs along the bottom unless the axes are reversed or the graph is rotated.

Destination indicates how to display the graph.

Check Notebook Page to display the graph inserted in a notebook page. When you choose Create Graph, it appears over the selected data block. You can click inside its borders and drag it to a new location, if you prefer.

If you want to edit the graph, check Graph Window. When you choose Create Graph, the graph appears in a graph window, ready for editing. The SpeedBar contains drawing tools so you can add drawn objects and text to the graph, or you can right-click any part of the graph and choose the Properties command to change its properties. For more information on customizing graphs, see Customizing Graph Properties.

Consolidation Expert, Step 1

This dialog box lets you choose two or more source blocks to be consolidated. Select a block, then choose Add to add it to the source block list. To delete a block from the list, highlight it and choose Delete. To choose a block in another notebook, click Browse. Find the notebook in the directory lists, then enter the block at the bottom of the dialog box and choose OK.

If your blocks include row or column labels, you can use them to sort the data. For example, if one block contains columns labeled Costs and Revenues, and the other block reverses the columns to show Revenues then Costs, when you specify a destination block in a later step and perform the consolidation, all the data under Costs will be combined and so will the data under Revenues, even though the columns are in a different order.

When you've selected all blocks to be combined, click Next Step to choose a consolidation operator: SUM, AVG (average), COUNT, MIN (minimum), MAX (maximum), STD (standard deviation), STDS (sample standard deviation), VAR (variance), and VARS (sample variance).

Consolidation Expert, Step 2

This dialog box lets you choose a consolidation operator: SUM, AVG (average), COUNT, MIN (minimum), MAX (maximum), STD (standard deviation), STDS (sample standard deviation), VAR (variance), and VARS (sample variance).

Each operator works like the statistical @function of the same name to combine equivalent cells of each source block.

Choose Next Step to specify a destination block.

Consolidation Expert, Step 3

This dialog box lets you specify a destination block. It can be the whole block or just the upper left cell of the block. If you choose just one cell, be sure there is enough room to the right and down from that cell to hold the combined blocks. Otherwise, data in unprotected cells is overwritten.

If the source blocks contain labels, you can use labels in the destination block to sort and filter data. Labeled data will appear beneath matching labels in the output block, so you can control the order of output by ordering labels in the destination block. Also, if you include labels in the destination block, but don't include every label used in the source blocks, consolidated data only appears for labels included in the destination block. If source blocks have labels, but you include no labels in the destination block, all source block labels appear in the consolidation.

When you have specified a destination block and checked whether to use column and/or row labels, choose Next Step.

Consolidation Expert, Step 4

This dialog box lets you name the consolidation setup and perform the consolidation. Enter a name, then click Consolidate to save the setup and enter combined data in the destination block. The consolidation setup is saved when you save the notebook.

Now, you can perform the same consolidation with different data. Add different data to the source blocks, Then, use Tools|Consolidator to display the Consolidator SpeedBar. Select the name you gave to the consolidation setup and click the Consolidate button. For details, see Using the Consolidator.

Scenario Expert, Step 1

The Scenario Manager lets you create and save scenarios -- sets of conditions and results. For example, you might have a Worst Case scenario with sales revenues much lower than you expect, while the Best Case scenario shows greater-than-expected sales and how they boost profits.

Scenario "changing cells" are the cells you change for each scenario -- sales revenue, in the previous example. "Result cells" contain formulas that reference the changing cells. They show the results of entering different data in the changing cells. If you haven't yet entered these into your notebook, close the Scenario Expert and create change and result cells for your scenario set.

In this dialog box, you specify one or more changing cells. They can be connected within a block or separated (noncontiguous). To specify noncontingous cells or blocks as changing cells, select the first then press Ctrl and drag or click the mouse to select the next, and so on.

When you're done, choose Next Step to name the scenario.

Scenario Expert, Step 2

This dialog box lets you name each scenario and enter values into its changing cells. You must enter a name, but you don't need to enter a different value into each changing cell--just change values as appropriate for the scenario you are defining.

When you're done, click Add Scenario and choose Next Step to highlight changing and result cells in the notebook or delete scenarios.

Scenario Expert, Step 3

This dialog box lets you highlight scenario values in your notebook and delete unwanted scenarios.

To show a scenario, highlight its name in the list and click Show Scenario. If values are hidden, click in the scenario title bar and drag the window out of the way. Changing cells are yellow and result cells are green.

If you don't want to keep a scenario, highlight it in the list and click Delete Scenario.

When you're done, click Next Step to create a Scenario Summary Report for all scenarios.

Scenario Expert, Step 4

This dialog box lets you create a Scenario Summary Report showing changing and result cells and their values for each scenario.

To create a report, click Create Report. The report appears and the Scenario Expert closes. The report appears on a new page, the Scenarios page. You can display it and print it just like any other notebook page.

If you already used the Scenario Expert to create a Scenario Summary Report, you'll be prompted to replace the existing report. If you choose No, the Scenario Expert closes without creating a new report.

To close the Scenario Expert without creating a report, choose Exit.

Once you create scenarios in the Scenario Manager, you can use Tools|Scenario Manager to display and edit them. For details, see Using the Scenario Manager.

Experts List

Quattro Pro offers these Experts to help with various tasks:

Graph Expert	Builds a graph from the block of data you select. If you include labels in the block, they appear in the graph. If you select the data block before you use the Graph Expert, the graph appears as soon as you open the Expert.
Scenario Expert	Lets you create and display groups of scenarios (data conditions and results) based on models in your spreadsheet notebook. Before you use this Expert, enter a set of values (the conditions) and formulas based on them (the results).
Consolidation Expert	Lets you combine data blocks using statistical operators (SUM, AVG, COUNT, MIN, MAX, STD, STDS, VAR, VARS). You can include column and row labels in the blocks to combine. Then, combined data is sorted according to label placement in the output block.
Performance Expert	Helps you determine if Quattro Pro's Compiled Formulas feature will speed up calculation times with your particular hardware and notebooks.
Analysis Expert	Offers 19 tools for financial, statistical, and engineering analysis.

See Also
Experts

Performance Expert, Step 1

The Performance Expert helps determine if the active notebook will show improved performance if you use compiled formulas. This dialog box tests your computer hardware to see if its CPU, accessories, and memory are suitable for compiling formulas. The check beside each item in the right pane indicates suitability. If an item isn't checked, it doesn't meet the requirements for compiling formulas.

The last item in the dialog box indicates whether the hardware is adequate. If Yes is checked, choose Next Step. If No, cancel this Expert; compiled formulas aren't likely to improve performance on this computer system.

Performance Expert, Step 2

If other conditions are appropriate, compiled formulas show best performance improvements in large notebooks that take a long time to recalculate. This dialog box lets you indicate how long the notebook takes to recalculate with uncompiled formulas.

Check Yes if recalculation time is over 5 seconds. Otherwise, check No. Then choose Next Step.

Performance Expert, Step 3

Compiled formulas help the most when a notebook contains lots of basic mathematical operations. This dialog box lets you indicate the main type of formulas contained in the active notebook.

Check Math Calculations if you work mainly with numbers; check Text Manipulation if you work mainly with text strings using the string @functions (@MID, @LENGTH, and so on).

Performance Expert, Step 4

Compiled formulas are most effective when you create a notebook, then use it to calculate values in classic spreadsheet fashion, as an electronic ledger book.

If you often edit the notebook by adding and deleting columns and rows, moving blocks of data, and reformatting it, compiled formulas are less helpful.

This dialog box lets you indicate how you work with the active notebook. Check Entering Data if you use the notebook mainly to enter data into cells for calculation by standard formulas that reference those cells. Check Frequent Editing if you rearrange the notebook or change its formulas frequently.

When you're done, choose Next Step.

Performance Expert, Step 5: Don't Compile

This Performance Expert step gives the results of the performance analysis. Read the recommendation in the right pane. Compiled formulas aren't recommended; click the Exit button to close the Performance Expert.

If you want to try compiled formulas anyway, display Recalc Settings properties in the active notebook Object Inspector: right-click the notebook title bar, be sure Recalc Settings is selected in the property list at the left, then check Compile Formulas at the bottom of the right pane. For more information on this setting, see Compiling Formulas.

Performance Expert, Step 5: Compile

This Performance Expert step gives the results of the performance analysis. Read the recommendation in the right pane. Compiled formulas are recommended; you can click Try Out Compiled Formulas to use them.

If you want to go back to uncompiled formulas, display Recalc Settings properties in the active notebook Object Inspector: right-click the notebook title bar, be sure Recalc Settings is selected in the property list at the left, then uncheck Compile Formulas at the bottom of the right pane. For more information on this setting, see Compiling Formulas.

Analysis Expert, Step 1

For a brief description of an analysis tool, highlight its name in the list and read the description in the left pane. To use a tool, double-click its name or highlight the name and choose Next Step. For context-sensitive information within the Analysis Expert, click Tip.

To view detailed information on edit fields and controls in the right pane (Step 2 or higher) , choose Tip, then choose Help.

@@(Cell)

@@ is used to reference a cell that contains another cell address or block name that is written as a label. @@ translates the label into a cell or single-cell block reference and returns the contents of that cell. @@ does not accept a block name for a block that is not a single-cell block.

Examples

`@@("A15")` = the contents of A15

`@@("BLOCK_NAME")` = the contents of the single-cell block named BLOCK_NAME

`@@(A3)` = 50 if A3 contains the label 'A1 and cell A1 contains the value 50

`@@(A3)` = the label 'Total if A3 contains the label 'Block, which is the name of cell C9, which contains the label 'Total

`@@(A1)` = ERR, where A1 contains the label 'B1..B5

`@@("A1")` = B1..B5, where A1 contains the label 'B1..B5

`@SUM(@@(A1))` = the sum of B1..B5, where A1 contains the label 'B1..B5, because Quattro Pro translates the label into cell coordinates for a non-single cell block

@ABS(X)

@ABS returns the absolute (positive) value of *X*.

Examples

`@ABS(-100)` = 100

`@ABS(100)` = 100

`@ABS(0)` = 0

@ACOS(X)

@ACOS returns the arc cosine of *X*. The result is the angle (in radians) whose cosine is *X*. To convert radians to degrees, use @DEGREES.

Examples

`@ACOS(1)` = 0

`@ACOS(0.5)` = 1.047198

`@DEGREES(@ACOS(0.5))` = 60

`@ACOS(@ABS(B10))` = the arc cosine of the absolute value of B10

`@ACOS(2)` = ERR (means that X is greater than 1)

@ASIN(X)

@ASIN calculates the arc sine of *X*. The result is the angle (in radians) whose sine is *X*. To convert radians to degrees, use @DEGREES.

Examples

`@ASIN(1)` = 1.570796

`@ASIN(0.25)` = 0.25268

`@DEGREES(@ASIN(0.5))` = 30

`@ASIN(-2)` = ERR (X is less than -1)

@ATAN(X)

@ATAN calculates the arc tangent of *X*. The result is the angle (in radians) whose tangent is *X*. To convert radians to degrees, use @DEGREES.

Examples

`@ATAN(0.5)` = 0.463648

`@ATAN(1)` = 0.785398

`@DEGREES(@ATAN(1))` = 45

@ATAN2(X,Y)

@ATAN2 calculates the arc tangent of the angle represented by the point with (x,y) coordinates *X* and *Y*. The result is the angle (in radians) whose tangent is *Y*/*X*. The result is between -pi and pi, with the quadrant chosen appropriately according to the sign of the result. If both *X* and *Y* are 0, the result is ERR.

Note: The order of arguments is the same as for 1-2-3, but opposite that of the ATAN2 function in FORTRAN and other programming languages.

To convert radians to degrees, use @DEGREES.

Examples

`@ATAN2(1,2)` = 1.107149

`@DEGREES(@ATAN2(1,1))` = 45

@AVG(List)

@AVG calculates the arithmetic mean of all values in *List*, using the formula:

Sum of *List* divided by *N*

If *List* contains more than one item, they must be separated by commas. If any of the cells referenced contains ERR, the resulting value is ERR.

Note: @AVG ignores blank cells in a block when it makes its calculations. Cells containing blank labels, however, are treated as 0; make sure blank cells are empty.

Examples

	B	C	D
1			
2	January	February	March
3	$652	$833	$599
4	$456	$305	$522
5	$68	$59	$73

`@AVG(5,20,10,5)` = 10

`@AVG(B3..D3)` = $694.67

`@AVG(225,B3..D5)` = $379.20

`@AVG(B3..B5,PART)` = $395.50 when C3..C5 is named PART

@CELL(Attribute,Block)

@CELL returns the requested attribute of the upper left cell in *Block*.

If you type in or point to a single-cell address when entering *Block*, Quattro Pro converts it to a block reference.

You can enter attributes in either upper- or lowercase, but you must surround them with double quotes. You can also reference a cell containing an attribute.

Note: @CELL does not recalculate automatically; press F9 to obtain the current value.

Examples

	A	B	C	D
1				
2		January	February	March
3	Advertising	$652	$833	$599
4	Car expenses	$456	$305	$522
5	Cleaning	$80	$80	$80

`@CELL("prefix",A3)` = '

`@CELL("format",B5)` = C0

`@CELL("type",D4)` = v

`@CELL("address",A3)` = $A$3

`@CELL("row",B4)` = 4

See Also

Attribute Arguments

Attribute Arguments

You may enter any of these as Attribute arguments for @CELL, @CELLINDEX, and @CELLPOINTER:

"address"	The address of the upper left cell in Block.
"row"	The row number of the upper left cell in Block.
"col"	The column number of the upper left cell in Block; 1=A, etc.
"sheet"	Page number of the upper left cell in Block (1 to 256, corresponding to notebook pages A through IV).
"NotebookName"	Referenced notebook name, 8 characters or fewer.
"NotebookPath"	Full path name of the referenced notebook.
"TwoDAddress"	2-D address of the referenced cell--`$G$23`, for example. The page name is never returned, even if the referenced cell is on another page or in another notebook.
"ThreeDAddress"	3-D address of the referenced cell--`$A:$G23`, for example. The page name is always returned.
"FullAddress"	Full address of the referenced cell--`[NOTEBK1]$A:$G23`, for example. The notebook and page names are always returned.
"contents"	The contents of the upper left cell in Block.
"type"	The type of data in the upper left cell in Block: b if the cell is blank, v if the cell contains a number or any formula, l if the cell contains a label.
"prefix"	The label-prefix character of the upper left cell in Block: ' if label is left-aligned, ^ if label is centered, " if label is right-aligned, \ if label is repeating.
"protect"	The protected status of the upper left cell in Block: 0 if cell is not protected, 1 if cell is protected.
"width"	The width of the column containing the upper left cell in Block (between 1 and 1024).
"rwidth"	The width of the block.
"format"	The numeric format code of the upper left cell in Block:

Fn	Fixed (n = 0-15)
Sn	Scientific (n = 0-15)
Cn	Currency (n = 0-15)
,n	Commas used to separate thousands (n = 0-15)
G	General
+	+/- (bar graph format)
Pn	Percent (n = 0-15)
D1-D5	Date
	D1 = DD-MMM-YY
	D2 = DD-MMM
	D3 = MMM-YY
	D4 = MM/DD/YY, DD/MM/YY, DD.MM.YY, YY-MM-DD

	D5 = MM/DD, DD/MM, DD.MM, MM-DD
D6-D9	Time
	D6 = HH:MM:SS AM/PM
	D7 = HH:MM AM/PM
	D8 = HH:MM:SS-24hr, HH.MM.SS-24hr, HH,MM,SS-24hr, HHhMMmSSs
	D9 = HH:MM-24hr, HH.MM-24hr, HH,MM, HHhMMm.
T	Show Formulas (Text)
H	Hidden
U	User-defined

@CELLINDEX(Attribute,Block,Col,Row,<Page>)

Same as @CELL, but returns the requested attribute of the cell in the specified column and row of *Block* on optional *Page*. The upper left corner of *Block* is column 0, row 0. Likewise, the first page is 0.

Note: @CELLINDEX does not recalculate automatically. Press F9 to obtain the current value.

Examples

	A	B	C	D
1				
2		January	February	March
3	Advertising	$652	$833	$599
4	Car expenses	$456	$305	$522
5	Cleaning	$80	$80	$80

`@CELLINDEX("prefix",A1..D5,0,2)` = '

`@CELLINDEX("format",B3..D5,0,2)` = C0

`@CELLINDEX("type",B3..D5,2,1)` = v

`@CELLINDEX("address",A1..D5,0,2)` = $A$3

`@CELLINDEX("row",A1..D5,1,3)` = 4

See Also
Attribute Arguments

@CELLPOINTER(Attribute)

@CELLPOINTER is similar to @CELL in that it returns the requested attribute of a cell. The only difference is that it reads the cell containing the selector. You cannot specify another cell. However, if you move the selector to a different cell and then press F9, the results of the @CELLPOINTER formula are updated.

You can enter attribute names in either upper- or lowercase, but each must be enclosed by double quotes.

This @function is useful in macros and @IF statements for quickly determining certain characteristics about the current cell, such as whether there is a label or a value currently in it. For example, this function statement tells Quattro Pro to write "value" in the cell if the current cell is a value; otherwise, it writes "label":

```
@IF(@CELLPOINTER("type")="v","value","label")
```

Examples

These examples refer to cell A1, which contains the date value 11/19/91.

`@CELLPOINTER("address")` = $A$1
`@CELLPOINTER("col")` = 1
`@CELLPOINTER("contents")` = 33561
`@CELLPOINTER("format")` = D4
`@CELLPOINTER("type")` = v

See Also

Attributes Arguments

@CHAR(Code)

@CHAR returns the onscreen character corresponding to the given code. This is useful in generating symbols not found on the keyboard.

Refer to Appendix C of *Building Spreadsheet Applications* or any standard ANSI table for the codes corresponding to each character.

Examples

@CHAR(33) = !

@CHAR(34) = "

@CHAR(35) = #

@CHAR(36) = $

@CHOOSE(Number,List)

@CHOOSE selects and enters a value from the supplied list. The value it chooses depends on the value of *Number*: 0 chooses the first value in the list; 1 chooses the second; 2 chooses the third, and so on. If you specify a cell address for *Number*, Quattro Pro uses the number contained in the cell. If the cell is blank, the first value is chosen.

The *List* values can be cell addresses, strings, numbers, or a mixture of the three. The total characters entered cannot exceed 1024.

@CHOOSE operates on integers only. If you supply a non-integer (such as 1.6433), the decimal values are disregarded. @VLOOKUP and @HLOOKUP perform similar tasks in tables.

Examples

`@CHOOSE(0,"Howie","Sarah","Chris")` = Howie

`@CHOOSE(1,"Howie","Sarah","Chris")` = Sarah

`@CHOOSE(2,"Howie","Sarah","Chris")` = Chris

`@CHOOSE(A15,"Howie","Sarah","Chris")` = Howie, if A15 is 0; Sarah if A15 is 1; Chris if A15 is 2.

`@CHOOSE(3,"Howie","Sarah","Chris")` = ERR (Number is too large).

@CLEAN(String)

@CLEAN removes all nonprintable characters (0-31) from a string.

@CODE(String)

@CODE returns the ANSI code of the first character in *String*. This is the opposite of @CHAR, which returns the character corresponding to the given code.

Examples

`@CODE("!")` = 33

`@CODE("Sam")` = 83 (code for S)

`@CODE("#")` = 35

`@CODE("$")` = 36

`@CODE("?")` = 63

`@CODE(hello)` = syntax error (missing quotes)

@COLS(Block)

@COLS returns the number of columns within the given block.

Examples

`@COLS(A1..IV1)` = 256

`@COLS(A1..A1)` = 1

`@COLS(NAME)` = 30 (if the NAME block contains 30 columns)

@COMMAND(CommandEquivalent)

@COMMAND returns the current value of a Quattro Pro for Windows command equivalent. It is most often used in macros to base the next action on a particular menu setting or to save current settings so they can be restored later.

CommandEquivalent must be enclosed in double quotes. To view a list of acceptable arguments, press Shift+F3 and choose *Command Equivalents*, or see Appendix A of *Building Spreadsheet Applications.*

@COMMAND returns strings; even if the setting is a number, it is returned as a string. Not all *CommandEquivalent* entries return a useful value. In general, @COMMAND only returns values for command equivalents that take arguments, usually menu commands that display a current setting or status.

Like @CELL, @COMMAND statements do not recalculate automatically as many other @functions do. Press F9 to obtain the current value.

A related @function that uses Quattro Pro for DOS menu equivalents is @CURVALUE. Another related @function, @PROPERTY, returns settings for requested object properties.

Examples

`@COMMAND("Print.Block")` = the currently specified print block

`@COMMAND("Print.Copies")` = the number entered after Copies in the Spreadsheet Print dialog box

@COS(X)

@COS returns the cosine of the angle *X*. *X* must be given in radians, not degrees. To convert degrees to radians, use @RADIANS.

Examples

`@COS(@RADIANS(60))` = 0.5

`@COS(@RADIANS(75))` = 0.258819

`@COS(@RADIANS(45))` = 0.707107

`@COS(@PI/3)` = 0.5

@COUNT(List)

@COUNT returns the number of nonblank cells in *List*. If more than one block is listed, they must be separated by commas.

Any single cells in *List* are counted as 1, even if they are blank. You can work around this by always using blocks (for example, if A4 is blank, use A3..A4, instead of just A4).

Examples

	A	B	C	D
1		January	February	March
2	John	$652	$833	$599
3	Mary	$456	$305	$522
4	Ralph	$68	$59	$73
5	Anna	$80	$80	$80
6				

`@COUNT(B2..B5)` = 4

`@COUNT(B1..B6)` = 5

`@COUNT(A6)` = 1

`@COUNT(A6..B6)` = 0

`@COUNT(C1..C5, D3..D6)` = 8

@CTERM(Rate,Fv,Pv)

@CTERM calculates the number of time periods required for an investment of *Pv* to reach a value of *Fv*, while earning interest of *Rate* per compounding period.

@CTERM assumes that the investment is an ordinary annuity. @NPER, which is calculated differently than but is related to @CTERM, uses an optional argument, Type, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

Assuming that your savings account has an annual interest rate of 7%, how long would it take a $3000 deposit to reach $5000? The answer is

```
@CTERM(7%,5000,3000) = 7.55 years
```

If the Rate figure is given for years, the result is in years as well. If you're working with monthly interest, multiply the answer by 12 to get a result in months.

You can also use @NPER to solve this problem:

`@NPER(7%,0,-3000,5000,0)` = 7.55

Other examples:

`@CTERM(0.07,5000,3000)` = 7.550042

`@CTERM(0.10,5000,3000)` = 5.359612

`@CTERM(0.12,5000,3000)` = 4.50747

`@CTERM(0.12,10000,7000)` = 3.147261

@CURVALUE(GeneralAction,SpecificAction)

@CURVALUE returns the current value of a menu command setting. It is used in macros, usually to base the next action on a particular menu setting. Both *GeneralAction* and *SpecificAction* must be enclosed by double quotes. They must together create one of the Quattro Pro for DOS menu-equivalent commands.

Not all *GeneralAction*/*SpecificAction* combinations return a useful value. In general, only menu commands that display a current setting or status have menu equivalents that are useful for @CURVALUE.

Like @CELL, @CURVALUE statements do not recalculate automatically as many other @functions do. Press F9 to obtain the current value.

Note: This @function is included to ensure compatibility with Quattro Pro for DOS. If you're running a DOS-version macro that includes this command, it may now return ERR instead of a value. This is because some settings previously made in Quattro Pro for DOS are now handled through Windows and aren't included in Quattro Pro for Windows. Most of these settings are found in the hardware and printing areas. The equivalent @function for Quattro Pro for Windows is @COMMAND.

Examples

`@CURVALUE("print","block")` = the block currently specified with File|Print (or /Print|Block in the Quattro Pro for DOS slash menu system)

`@CURVALUE("file","save")` = the name of the last file saved

@DATE(Yr,Mo,Day)

@DATE returns the "serial number" of the date specified with year, month, and day arguments. This serial number can range from -109,571 (January 1,1600) to 474,816 (December 31, 3199). December 30, 1899 is 0, so a positive number represents the number of days from December 30, 1899 up to the date referenced in the formula. Date serial numbers are used in notebook calculations. (The fractional portion of a date serial number is used for the time @functions.)

To display a date serial number in a date format, choose Numeric Format in the Block property menu. This shows the date in its more common form (for example, `Jan-1-94` instead of `34335`).

Any illegal dates return ERR as their value, for example, `@DATE(87,2,29)`. (This date corresponds to February 29, 1987, which is impossible; 1987 was not a leap year.)

Examples

`@DATE(93,1,1)` = 33970 (January 1, 1993)

`@DATE(91,9,13)` = 33494 (September 13, 1991)

`@DATE(-300,1,1)` = -109571 (January 1, 1600)

`@DATE(102,1,1)` = 37257 (January 1, 2002)

@DATEVALUE(DateString)

@DATEVALUE returns a serial date value that corresponds to the value in *DateString*. If the value in *DateString* is not in the correct format or is not enclosed in quotes, ERR or a syntax error message is returned. If *DateString* is entered using the international format, the year, month, and day must be in the same order as the current international date format (set in the Applications property menu) and the separator character must also agree.

You can display resulting date string values in standard date formats by choosing Numeric Format in the Block property menu.

There are five valid formats for *DateString*:


DD-MMM-YY ("04-Jul-92").


DD-MMM ("04-Jul") (assumes the current year).


MMM-YY ("Jul-92") (assumes the first of the month).


The Long International date format specified as the system default, one of which is MM/DD/YY ("07/04/92").


The Short International date format specified as the system default, one of which is MM/DD ("07/18"). This format assumes the current year.

Note: The easiest way to enter a date value is with the date prefix (Shift+Ctrl+D). @DATEVALUE is included for compatibility with other products.

Examples

`@DATEVALUE("07/04/92")` = 33789

`@DATEVALUE("JUL-92")` = 33786 (July 1, 1992)

`@DATEVALUE("04-may-93")` = 34093

`@DATEVALUE(07/04/94)` = 0.018617 (no quotes makes Quattro Pro divide the numbers)

`@DATEVALUE("May-04-1992")` = ERR

@DAVG(Block,Column,Criteria)

@DAVG averages selected field entries in a database. It includes only those entries in *Column* whose records meet the criteria specified in *Criteria*.

The field specified in your criteria and the field being averaged need not be the same. The field averaged is that contained within the column you specify as *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

These examples refer to the database and criteria tables.

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		**Criteria Table**		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

`@DAVG(A2..D8,3,A11..A12)` = $19,481 (average of July sales)

`@DAVG(A2..D8,3,B11..B13)` = $26,023 (average of California sales)

`@DAVG(A2..D8,3 C11..C12)` = $18,123 (average of CJ's sales)

`@DAVG(A2..D8,4,A11..C13)` = ERR (Column figure too high)

`@DAVG(A2..D8,2,A11..A12)` = 0 (labels are treated as 0)

@DAY(DateTimeNumber)

Converts the date/time serial number you supply as *DateTimeNumber* into the number associated with that day (1-31). Decimal (time) portions of the number are ignored.

Examples

`@DAY(33508)` = 27 (9/27/91)

`@DAY(32134)` = 23 (12/23/87)

`@DAY(@DATE(93,9,10))` = 10

`@DAY(474817)` = ERR because the number you entered was larger than 474816.

@DCOUNT(Block,Column,Criteria)

@DCOUNT counts selected field entries in a database. It includes only those entries in *Column* whose records meet the criteria specified in block *Criteria*.

The field specified in your criteria and the field being counted need not be the same. The field counted is that contained within the column you specify as *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	San Jose	RX	$25,000
5	Jul-91	Chicago	RX	$18,998
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

These examples refer to the database and criteria tables.

`@DCOUNT(A2..D8,3,A11..A12)` = 4 (number of July sales)

`@DCOUNT(A2..D8,3,B11..B13)` = 3 (number of California sales)

`@DCOUNT(A2..D8,3,C11..C12)` = 3 (number of CJ's sales)

`@DCOUNT(A2..D8,4,A11..C13)` = ERR (Column figure too high)

`@DCOUNT(A3..D8,3,A11..A12)` = 6 (incorrect--field names not included)

@DDB(Cost,Salvage,Life,Period)

@DDB determines accelerated depreciation values for an asset, given the initial cost, life expectancy, end value, and depreciation period. It calculates depreciation using the double-declining balance method.

Life must be equal to or greater than *Period*; both must be integers greater than 0. *Cost* must be equal to or greater than *Salvage*; both must be equal to or greater than zero.

@SLN and @SYD offer other depreciation methods.

Examples

Suppose you just bought a new $4000 computer. The dealer says you can sell it back to the store for $350 after eight years, but no one would want to buy it after that. In other words, Salvage is $350 and Life is 8. To calculate the double-declining depreciation allowance of this computer by the second year, enter this formula:

```
@DDB(4000,350,8,2)
```

The result is $750.

These examples show depreciation values for the first five years of a $15,000 investment with a salvage value of $3000 and a life of 10 years:

`@DDB(15000,3000,10,1)` = $3,000

`@DDB(15000,3000,10,2)` = $2,400

`@DDB(15000,3000,10,3)` = $1,920

`@DDB(15000,3000,10,4)` = $1,536

`@DDB(15000,3000,10,5)` = $1,229

@DDELINK([AppName | Topic]"DataToReceive", <nCols>, <nRows>, <nSheets>)

@DDELINK creates a "live" data link from another Windows application that supports DDE (Dynamic Data Exchange). Using @DDELINK is equivalent to choosing Edit|Paste Link with data from another application copied to the Clipboard.

When you enter @DDELINK into a cell, the linked data appears there. Unless you indicate otherwise, the data takes up as much space as it did in the original application. You can use *nCols, nRows,* and *nSheets* to specify smaller dimensions. If any of the arguments is 0 or omitted, the original dimension applies.

Caution: @DDELINK sets up a zone of cells that can be overwritten whenever data changes in the DDE-server application. Avoid storing other data near @DDELINK, and consider using the limit arguments.

Examples

This formula gets information from the field Task in the ObjectVision application TASKLIST.OVD and displays the data in the active notebook:

```
@DDELINK([VISION|TASKLIST]"Task")
```

The maximum size of the data block for the following formula is 5 cells by 5 cells:

```
@DDELINK([EXCEL|FILE1]"R1C1:R5C5")
```

With *nRows* = 3, the maximum size of the data block drops to 5 cells by 3 cells:

```
@DDELINK([EXCEL|FILE1]"R1C1:R5C5",0,3)
```

See Also

Creating DDE Links

@DEGREES(X)

@DEGREES converts the given number of radians to degrees, using this formula:

180 times *X* divided by pi

Examples

`@DEGREES(0.5)` = 28.64789

`@DEGREES(0.017)` = 0.974028

`@DEGREES(@PI/2)` = 90

@DMAX(Block,Column,Criteria)

@DMAX finds the maximum value of selected field entries in a database. It includes only those entries in *Column* whose records meet the criteria specified in block *Criteria*.

The field specified in your criteria and the field you are finding the maximum value for need not be the same. The field you are finding the maximum value for is that contained within the column you specify as *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

`@DMAX(A2..D8,3,A11..A12)` = $28,725 (highest July sale)

`@DMAX(A2..D8,3,B11..B13)` = $34,345 (highest California sale)

`@DMAX(A2..D8,3,C11..C12)` = $23,769 (highest of CJ's sales)

`@DMAX(A2..D8,4,A11..C13)` = ERR (Column figure too high)

`@DMAX(A3..D8,3,A11..A12)` = $34,345 (incorrect--field names not included)

@DMIN(Block,Column,Criteria)

@DMIN finds the minimum value of selected field entries in a database. It includes only those entries in *Column* whose records meet the criteria specified in block *Criteria*.

The field specified in your criteria and the field for which you are finding the minimum value need not be the same. The field for which you are finding the minimum value is that contained within the column you specify as *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

`@DMIN(A2..D8,3,A11..A12)` = $14,999 (smallest July sale)

`@DMIN(A2..D8,3,B11..B13)` = $14,999 (smallest California sale)

`@DMIN(A2..D8,3,C11..C12)` = $14,999 (smallest of CJ's sales)

`@DMIN(A3..D8,3,A11..A12)` = $15,600 (incorrect--field names not included)

`@DMIN(A2..D8,4,A11..C13)` = ERR (Column figure too high)

@DSTD(Block,Column,Criteria)

@DSTD finds the population standard deviation for selected field entries in a database. @DSTDS computes the standard deviation of sample data.

@DSTD includes only those entries in *Column* whose records meet the criteria specified in block *Criteria*.

The field specified in *Criteria* and the field for which you are finding the standard deviation need not be the same. The field for which you are finding the standard deviation is the field contained within *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field in *Block*. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

`@DSTD(A2..D8,3,B11..B13)` = $8,126 (population SD of California sales)

`@DSTD(A2..D8,3,C11..C12)` = $4,000 (population SD of CJ's sales)

`@DSTD(A2..D8,4,A11..C13)` = ERR (Column figure too high)

`@DSTDS(A2..D8,3,B11..B13)` = $9,952 (sample SD of California sales)

`@DSTDS(A2..D8,3,C11..C12)` = $4,899 (sample SD of CJ's sales)

@DSTDS(Block,Column,Criteria)

@DSTDS finds the sample standard deviation for selected field entries in a database. @DSTD computes the standard deviation of population data.

This @function isn't compatible with 1-2-3. If your file must be compatible with 1-2-3, use @DSTD instead.

@DSUM(Block,Column,Criteria)

@DSUM totals selected field entries in a database. It includes only those entries in *Column* whose records meet the criteria specified in *Criteria*.

The field specified in *Criteria* and the field you are finding the sum of need not be the same. The field you are finding the sum of is that contained within *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

`@DSUM(A2..D8,3,A11..A12)` = $77,924 (total of July sales)

`@DSUM(A2..D8,3,B11..B13)` = $78,069 (total of California sales)

`@DSUM(A2..D8,3,C11..C12)` = $54,368 (total of CJ's sales)

`@DSUM(A2..D8,1,A11..A12)` = 0

`@DSUM(A2..D8,4,A11..C13)` = ERR (Column figure too high)

@DVAR(Block,Column,Criteria)

@DVAR calculates the population variance for selected field entries in a database. @DVARS computes the variance of sample data.

@DVAR includes only those entries in *Column* whose records meet the criteria specified in *Criteria*.

The field specified in *Criteria* and the field for which you are calculating the variance need not be the same. The field analyzed is the field contained within *Column*.

You can specify all or part of your database as *Block*, but field names must be included for each field you include in the block. For more information about database blocks, see Setting Up a Database.

Examples

	A	B	C	D
1				
2	DATE	LOCATION	REP	AMOUNT
3	Jul-91	San Fran	CJ	$14,999
4	Jul-91	LA	RX	$28,725
5	Jul-91	Chicago	RX	$18,600
6	Jul-91	NY	CJ	$15,600
7	Aug-91	Chicago	CJ	$23,769
8	Aug-91	LA	RX	$34,345
9				
10		CRITERIA TABLE		
11	Date	Location	Rep	
12	Jul-91	San Fran	CJ	
13		LA		

@DVAR(A2..D8,3,B11..B13) = $66,028,355 (pop. variance of Calif. sales)

@DVAR(A2..D8,3,C11..C12) = $16,000,740 (pop. variance of CJ's sales)

@DVAR(A2..D8,4,A11..C12) = ERR (Column figure too high)

@DVARS(A2..D8,3,B11..B13) = $99,042,532 (sample variance of Calif. sales)

@DVARS(A2..D8,3,C11..C12) = $24,001,110 (sample variance of CJ's sales)

@DVARS(Block,Column,Criteria)

@DVARS calculates the sample variance for selected field entries in a database. @DVAR computes variance with population data.

This @function isn't compatible with 1-2-3. To use the file in 1-2-3, use @DVAR instead.

@ERR

@ERR returns the value ERR in the current cell and in any other cells that reference the current cell, either directly or indirectly. (Exceptions to this are @COUNT, @DCOUNT, @ISERR, @ISNA, @ISNUMBER, @ISSTRING, and @CELL formulas; these will not result in ERR if they reference an ERR cell.)

The ERR value resulting from this @function is the same as the ERR value produced by Quattro Pro when it encounters an error. It is often used with @IF to bring attention to error conditions.

ERR is a unique number, not to be confused with the label ERR.

Examples

@ERR = ERR

@IF(B6>B7,0,@ERR) = 0 (if B6>B7) or ERR (if B6<B7)

@EXACT(String1,String2)

@EXACT compares the values of *String1* and *String2*. If the values are exactly identical, including capitalization and diacritical marks (such as ~), it returns 1. If there are any differences, it returns 0.

If you're comparing literal strings, surround them with double quotes. If you use a block name or cell address, no quotes are necessary. You can compare the contents of label cells only. If you try to compare one or more numbers or empty cells, the result is ERR. When you compare labels, label prefixes are ignored.

To compare strings or cell contents without regard to capitalization or diacritical marks, use @IF. For example, @IF(C3=B3,1,0) returns 1 if the contents of the cells are the same but are capitalized differently.

Examples

@EXACT("client","Client") = 0

@EXACT("client","client") = 1

@EXACT(29,"29") = ERR (the first string is a value)

@EXACT(A1,"yes") = 1 (if A1 contains the label yes)

@EXACT(client,client) = syntax error (no quotes)

@EXACT("client","client","client ") = syntax error (more than two strings)

@EXP(X)

@EXP returns the mathematical constant e, raised to the *X*th power. This @function is the inverse of a natural logarithm, @LN.

Examples

@EXP(3.4) = 29.9641000474

@EXP(1) = 2.718281828459 (the actual value of e)

@SQRT(@EXP(2)) = 2.71828183

@LN(@EXP(2.5)) = 2.5

@FALSE

@FALSE returns the logical value 0 and is usually used in @IF formulas. The zero that it returns is the same as any other zero, but @FALSE makes the formula easier to read.

@TRUE is a related @function.

Examples

@FALSE = 0

@IF(C3=100,10,@FALSE) = 10 (if C3 = 100) or 0 (if C3 is not equal to 100)

@IF(C3=100,@TRUE,@FALSE) = 1 (if C3 = 100) or 0 (if C3 is not equal to 100)

@FILEEXISTS(FileName)

@FILEEXISTS returns a 1 if a file named *FileName* exists in the current file directory, and returns a 0 if it doesn't. *FileName* can be a block name containing a path or file name string. If entered as a literal string, *FileName* must be enclosed by quotes and must include any extension attached to the file name. To search for a file in a directory other than the default directory, include the directory path in *FileName*.

Examples

@FILEEXISTS("EXAMPLE.WB1") = 1 (if EXAMPLE.WB1 is in the working directory)

@FILEEXISTS("C:\DATA\EXAMPLE.WB1") = 1 (if EXAMPLE.WB1 is in the specified directory)

@FILEEXISTS(FILE_NAME) = 1 (if the block FILE_NAME contains a path and file-name label and if the file exists in that directory)

@FIND(Substring,String,StartNumber)

@FIND searches through *String* from left to right for *Substring*. If it finds *Substring*, it returns the character position of the first occurrence. *StartNumber* indicates where to begin the search: 0 = the first character in the string, 1 = the second, and so on. The value of *StartNumber* must not be more than the number of characters in *String* minus 1.

@FIND is case-sensitive and is also sensitive to diacritical marks used in non-English languages. You can overcome the case sensitivity of this @function by using @UPPER to force one or more of the strings into all uppercase letters. For example, the following formula forces both the substring in cell C3 and the string in cell C4 to uppercase, then searches for the substring:

@FIND(@UPPER(C3),@UPPER(C4),0)

@FIND is most often used in conjunction with two other string functions: @REPLACE (to perform "search and replace" operations on strings) and @MID (to access substrings).

If @FIND fails to find any occurrences of *Substring*, or if the *StartNumber* given is invalid, the result is ERR.

Examples

@FIND("i","find",0) = 1

@FIND("nd","find",2) = 2

@FIND("F","find",0) = ERR

@FIND("f","find",3) = ERR

@FIND("d","find",4) = ERR

@FIND(n,find,0) = syntax error (quote marks omitted from strings)

@FIND("hi",C4,0) = 1 (if C4 contains ship)

@FV(Pmt,Rate,Nper)

@FV returns the future value of an investment where *Pmt* is invested for *Nper* periods at the rate of *Rate* per period.

@FV assumes that the investment is an ordinary annuity. @FVAL, a related @function, uses an optional argument, Type, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

Assume you want to set aside $500 at the end of each year in a savings account that earns 15% annually. To determine what the account will be worth at the end of six years, enter this formula:

```
@FV(500,15%,6)
```

Your yearly payment of $500 will be worth $4,376.87 in six years. You could also use @FVAL:

```
@FVAL(15%,6,-500,0,0)
```

Note that in @FVAL, you have to be precise about whether a payment is out of your pocket (a negative number) or paid to you (a positive number).

Other examples:

`@FV(200,.12,5)` = $1,270.57

`@FV(500,0.9,4)` = $6,684.50

`@FV(800,0.9,3)` = $5,208.00

`@FV(800,0.9,A3)` = $40,929.67 (if A3 = 6)

@FVAL(Rate,Nper,Pmt,<Pv>,<Type>)

Like the related @function @FV, @FVAL returns the future value of an investment. The last two arguments, *Pv* and *Type*, are optional. If you omit the last one or both of them, Quattro Pro assumes their values are zero. These arguments let you define the problem as an annuity due (putting money into an account before it earns its interest for that year means you have an annuity due to you, which increases the future value). Be sure to enter negative numbers for money going out and positive numbers for money coming in to you.

This @function isn't compatible with 1-2-3. If your file must be compatible with 1-2-3, use @FV instead.

Examples

Assume you want to set aside $500 at the start of each year in a savings account that earns 15% annually. To determine what the account will be worth at the end of six years, starting at a present value of zero, enter this formula:

```
@FVAL(15%,6,-500,0,1)
```

Note that the payment is out of your pocket, so you enter a negative number. Your yearly payment of $500 will be worth $5,033.40 in six years, or $656.53 more than if you deposited the money at the last day of the year as in the example for @FV.

If the account already had $340 in it before your yearly deposits of $500, you could calculate the future value after six years with this formula:

`@FVAL(15%,6,-500,-340,1)` = $5,819.84

@HEXTONUM(Hex)

@HEXTONUM converts the hexadecimal number in the string to the corresponding decimal value. @NUMTOHEX performs the opposite conversion, from decimal to hexadecimal.

Examples

`@HEXTONUM("a")` = 10

`@HEXTONUM("10")` = 16

`@HEXTONUM("00FF")` = 255

`@HEXTONUM(A1)` = 10 (if cell A1 contains the label 'a)

See Also

Entering Number Conversion @Functions

@HLOOKUP(X,Block,Row)

The first row of *Block* contains the index values (comparison figures used to determine which column to search). Each cell of the index row must contain a value. If numbers, these values must be in ascending order.

@HLOOKUP provides an efficient way to access information stored in a table. @HLOOKUP searches horizontally through the first row of *Block* for the value *X*. When found, it returns the value itself (if *Row* = 0), or the value displayed the specified number of rows beneath it (as indicated by *Row*).

If *X* is a string, Quattro Pro looks for an exact case-sensitive match. If *X* is a number and Quattro Pro can't find an equal number, it locates the highest number in the row not more than *X*. If *X* is a number and the index row contains only labels, Quattro Pro stops at the rightmost column.

@HLOOKUP returns 0 if the referenced cell is blank. ERR is returned if:

Row is less than 0 or greater than the number of rows minus 1 in *Block*.

X is less than the smallest value in the topmost row of *Block*.

X and the index row entries are string values and Quattro Pro fails to find a match in the top row of *Block*.

X is a string or label and the index row entries are numeric values.

Examples

You might have a table listing days of the week by number:

0	1	2	3	4	5	6
Saturday	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday

You could use @HLOOKUP to enter the day of the week for a given date. For example,

`@HLOOKUP(@MOD(@DATE(92,11,11),7),A1..G2,1)` = Wednesday

Additional examples

	A	B	C	D
1	1	5	10	15
2	43	53	32	67
3	92	42	18	22
4	45	83	76	47

In the first example, Quattro Pro searches across the first row of the specified block (row 1), looking for the largest number equal to or less than 17. It stops at cell D1, then moves down the specified number of rows (3). It stops at cell D4 and returns the value 47.

`@HLOOKUP(17,A1..D4,3)` = 47

`@HLOOKUP(10,A1..D4,0)` = 10

`@HLOOKUP(6,A1..D4,2)` = 42

`@HLOOKUP(50,A1..D4,3)` = 47

`@HLOOKUP("18",A1..D4,2)` = ERR (index row values are not strings)

`@HLOOKUP(18,A1..D4,4)` = ERR (row value > # rows-1 in block)

To search vertically through a table, use @VLOOKUP.

@HOUR(DateTimeNumber)

@HOUR returns the hour portion of *DateTimeNumber*. *DateTimeNumber* must be a valid date/time serial number. Because only the decimal portion of a serial number pertains to time, the integer portion of the number is disregarded. The result is between 0 (12:00AM) and 23 (11:00PM).

To extract the hour portion of a string that is in time format (instead of serial format), use @TIMEVALUE with @HOUR to translate the time into a serial number. To return standard hours (1-12) instead of military hours (1-24), use @MOD with a parameter of 12. **See Also** @TIME.

Examples

`@HOUR(.25)` = 6

`@HOUR(.5)` = 12

`@HOUR(.75)` = 18

`@HOUR(@TIMEVALUE("10:08am"))` = 10

`@MOD(@HOUR(@TIMEVALUE("9:31:52 PM")),12)` = 9

@IF(Cond,TrueExpr,FalseExpr)

@IF evaluates the logical condition given as *Cond*. If the condition is found to be true, it returns the value given as *TrueExpr*. If the condition is false, it returns the value given as *FalseExpr*. *Cond* is true if it evaluates to any nonzero numeric value.

The formula entered as *Cond* can be any logical expression that can be evaluated as true or false; for example, `B6<0` or `C3*D2=53`.

You can use compound conditions by connecting expressions with #AND# or #OR#. If you use #AND#, both conditions given must be met to evaluate true. If you use #OR#, the expression is true if either of the conditions is met. For example, `A3<10#AND#A3>5` means that the value in A3 must be between 6 and 9 to evaluate true. You can also use the #NOT# operator to negate a condition. For example, `#NOT#(B3>10)` evaluates true if B3 is not greater than 10.

TrueExpr and *FalseExpr* can be numbers, formulas resulting in numbers, or text. If text, the string must be enclosed by double quotes; for example, @IF(D6=5,"John","Harry "). You can also use cell references to use the contents of other cells in the notebook. For example, @IF(B10<18,D5,C4) enters the contents of D5 if the condition is true, and enters the contents of C4 if the condition is false.

If the condition you specify with *Cond* searches a cell for a number and the cell contains a label, the label is evaluated as having a value of 0 and *FalseExpr* is returned. Likewise, if you search for a label and find a numeric value, *TrueExpr* results if the value of the referenced cell is 0; *FalseExpr* results if it is nonzero.

Although logical expressions typically reference other cells, this is not required. Any expression resulting in a numeric value is accepted; for example, A1=1 or A1="Fred". If the result of *Cond* is nonzero, *TrueExpr* is the result; otherwise, *FalseExpr* is the result.

@IF statements can be nested, or used within one another. In other words, *TrueExpr* can contain yet another test to further validate *Cond*.

For example, `@IF(B5>C6,@IF(B5>C7,1,2),3)` tells Quattro Pro to see if the contents of B5 are greater than C6. If they are, it then checks to see if B5 is greater than C7; if so, it enters a 1 in the cell. If not, it enters a 2. If B5 is not greater than C6, it enters a 3. There's no limit on the number of levels @IF expressions that you can nest, as long as the entire expression doesn't exceed 1024 characters.

Examples

`@IF(8=7,4,5)` = 5

`@IF(B4<100,"Yes","No")` = Yes if B4 < 100; otherwise, No

`@IF(C10=BLOCK,45,50)` = 45 if C10 = the cell named BLOCK; otherwise, 50

`@IF(C10,1,0)` = 0 if C10 = 0; otherwise, 1

@INDEX(Block,Column,Row,<Page>)

@INDEX searches through the table given as *Block* and returns the value specified with the *Column*, *Row*, and optional *Page* values. The upper left cell in *Block* is column 0, row 0. Likewise, the first page is 0. The *Column* and *Row* values are not the actual coordinates of the resulting cell, but instead are offset values. In other words, @INDEX begins in the top left cell of the given block, moves right the number of columns specified by *Column*, moves down the number of rows specified by *Row*, and through the number of pages specified by *Page* (if you've specified a *Page*). It then returns the value in the current cell.

The *Column*, *Row*, and *Page* values must be numbers equal to or greater than zero and less than the number of rows, columns, or pages in the block. If a fractional number is used (for example, 2.35), the fractional part is dropped (not rounded).

@HLOOKUP and @VLOOKUP are related functions.

Examples

	A	B	C	D
1	1	5	10	15
2	43	53	32	67
3	92	42	18	22
4	45	83	76	47

These examples reference cells in the data table:

`@INDEX(A1..D4,3,2)` = 22

`@INDEX(A1..D4,1,2)` = 42

`@INDEX(C2..D3,0,1)` = 18

`@INDEX(C2..D3,1,3)` = ERR (too many rows)

`@INDEX(A1..D4,-2,3)` = ERR (negative column number)

@INT(X)

@INT drops the fractional portion of *X*, returning its integer value. @ROUND rounds *X* to the nearest integer.

Examples

`@INT(499.99)` = 499

`@INT(0.1245)` = 0

`@INT(-2.3)` = -2

`@INT(C4)` = 5 if C4 contains a value between 5 and 6

@IPAYMT(Rate,Per,Nper,Pv,<Fv>,<Type>)

@IPAYMT and @PPAYMT tell what portion of a particular loan payment is interest and what portion is principal, respectively. For each month in the transaction period: @PAYMT(*Rate*, *Nper*, *Pv*, *Fv*, *Type*) = @IPAYMT(*Rate*, *Per*, *Nper*, *Pv*, *Fv*, *Type*) + @PPAYMT(*Rate*, *Per*, *Nper*, *Pv*, *Fv*, *Type*).

The last two arguments, *Fv* and *Type*, are optional. If you omit one or both of them, their values are assumed to be zero.

Examples

If you are two years into a 30-year, 10% mortgage on a $100,000 loan and your interest payment is tax-deductible, then `@IPAYMT(.1/12,2*12,30*12,100000)` returns your current month's deduction: -824.03.

@IRATE(Nper,Pmt,Pv,<Fv>,<Type>)

Like @RATE, @IRATE calculates the interest rate required for an investment or loan of *Pv* to reach *Fv* within *Nper* periods, given *Pmt*. The last two arguments, *Fv* and *Type*, are optional. If you omit one or both of them, their values are assumed to be zero.

@IRATE requires that the initial cash flow (*Pv* + *Type* * *Pmt*) and the last cash flow (*Fv* + (1-*Type*) * *Pmt*) have opposite signs. Otherwise, @IRATE returns ERR because the transaction is not simple and there may not be a meaningful rate.

@IRATE isn't compatible with 1-2-3. If your file must be compatible with 1-2-3, use @RATE instead.

Examples

Assume you're negotiating to buy a $15,000 new car. The salesperson says you can have the car for $500 a month for the next five years. To calculate the monthly percentage rate:

`@IRATE(5*12,-500,15000,0,0)` = 0.02632

Another example: Assume that you plan to deposit $2000 a year into a savings account that currently contains only $2.38. What interest rate must the account earn to generate $15,000 at the end of 5 years? Use this formula:

`@IRATE(5,-2000,-2.38,15000,0)` = 0.2038

@IRR(Guess,Block)

@IRR determines the internal rate of return on an investment.

Before using @IRR, you must set up a table of expected cash flow amounts over a period of time. Quattro Pro assumes that the amounts are received at regular intervals. Negative amounts are interpreted as cash outflows, and positive amounts as inflows. The first amount must be a negative number, to reflect the initial investment. These amounts can all be the same for each time period, or they can be different (including a mixture of negatives, positives, or zeros).

Typically, you make an investment (a negative cash flow) and then receive several dividends (positive cash flows). This is an example of a simple transaction, and @IRR gives the unique rate of return for this without requiring a *Guess*. More complex transactions, in which the direction of money changes several times, often do not have a meaningful value for @IRR. For more information, see @IRR with Multiple Rates of Return.

@IRR(*Guess*,*Block*) gives the number Rate which satisfies @NPV(Rate,*Block*,0) = 0. For a simple transaction, @NPV(@IRR(*Block*),*Block*,0) will give a number close to 0 (it may not be exactly 0 due to how numbers are rounded off).

Guess can be any value greater than -1. Values that are NA or less than or equal to -1 are ignored. Use @NA for *Guess* unless your cash flow has multiple rates of return (see below). If you use @NA, you will get ERR if your cash flow has more than one rate of return, rather than the rate of return that happens to be near your *Guess*.

Examples

	A	B	C
1	3000	-50000	-10000
2	700	-8000	1000
3	600	2000	1000
4	750	4000	1200
5	900	6000	2000

`@IRR(0,A1..A5)` = -1

`@IRR(0,B1..B5)` = -38.09%

`@IRR(0,C1..C5)` = -19.90%

See Also
Entering Cash Flow @Functions

@IRR with Multiple Rates of Return

In unusual cases, @IRR may have as many as N-1 roots, where N is the number of terms in the block. Consider the Block that has the values (-10, +150, -145). @IRR(@NA,Block) returns ERR because it is not simple. The two roots are 3.86% and 1296%, obtainable from guesses of 0 and 10, respectively. Both of these values are meaningful, if interpreted properly. For one, you are the borrower; for the other, the lender.

If you find a transaction with two roots, there is a mechanical way to determine which is the lender rate and which is the borrower rate. Pick a positive term in the Block, and increase it by a small amount. If the rate increases, it is a lender rate, and if the rate decreases, it is a borrower rate.

Most uses of @IRR are for analyzing an investment in which the first cash flow is negative, and the rate is a lender rate.

Some transactions have no rate of return at all. @IRR(Guess,Block), with Block having the values (-1,+1,-1), returns ERR regardless of the Guess. There is no rate of return that is meaningful for this cash flow.

See Also
Entering Cash Flow @Functions
@IRR

@ISERR(X)

@ISERR is normally used to check the contents of a cell for errors. If the cell contains ERR, 1 is returned; otherwise, 0 is returned. You can also use formulas or numeric values with @ISERR.

Examples

`@ISERR(C2)` =1 if C2 contains ERR; otherwise, 0

`@ISERR(10/0)` =1

`@ISERR(45+C3)` =1 if C3 is ERR; otherwise, 0

`@ISERR(C2/B3)` =1 if B3 is 0 or ERR, or if C2 is ERR; otherwise, 0

`@IF(@ISERR(A2),0,A5)` =0 if A2 is ERR; otherwise, it returns the value in A5

@ISNA(X)

@ISNA tests for the special value NA in a cell. If the cell contains an NA value, it returns 1; otherwise, it returns 0. NA is considered a special value; it appears in the notebook only through the use of @NA. Cells containing the label "NA" typed directly (not produced by @NA) are not recognized by @ISNA.

Examples

`@ISNA("NA")` = 0

`@ISNA(@NA)` = 1

`@ISNA(A18)` = 1 if A18 contains NA produced by @NA

@ISNUMBER(X)

@ISNUMBER examines *X* and determines if it contains a numeric value. If *X* is blank or contains a numeric value, ERR, or NA, @ISNUMBER returns a 1. If *X* is a label or text, @ISNUMBER returns a 0. @ISNUMBER is usually used with @IF to determine whether an entry is a value.

Examples

`@ISNUMBER(88)` = 1

`@ISNUMBER("88")` = 0 (quotes signify a text string)

`@ISNUMBER(9/15/87)` = 1

`@ISNUMBER(@ERR)` = 1 (ERR and NA are numeric values)

@ISSTRING(X)

@ISSTRING examines *X* and determines if it contains a label or text string. If *X* does (even if the string is empty), @ISSTRING returns 1. If *X* is blank or contains a numeric or date value, @ISSTRING returns 0.

Usually, @ISSTRING is used to test the contents of a cell. You can test any expression, however. Literal string arguments must be enclosed by double quotes.

Examples

`@ISSTRING(55)` = 0

`@ISSTRING(2/5/88)` = 0

`@ISSTRING("Hello, world.")` = 1

`@ISSTRING("Hello, "&"world.")` = 1

`@ISSTRING("55")` = 1

`@ISSTRING(A15)` = 1 if A15 contains a label or formula that results in a string, otherwise 0

`@ISSTRING(A15&A16&"!!!")` = 1 if A15 and A16 contain labels or formulas that result in strings

`@ISSTRING("")` = 1 ("" is an empty string)

`@ISSTRING(@NA)` = 0 (NA and ERR are considered numeric values)

@LEFT(String, Num)

@LEFT returns the leftmost *Num* characters of *String*. It lets you extract a specified number of characters starting from the left end of a string or label.

If *String* is a numeric or date value or a blank cell, @LEFT returns ERR. If *Num* is longer than the length of *String*, all of *String* is returned. The number of characters returned is never greater than the length of the string.

Examples

`@LEFT("Jennifer",5)` = Jenni

`@LEFT("Jennifer",15)` = Jennifer

`@LEFT("155",1)` = 1

`@LEFT("     Jennifer",6)` = J (including five leading spaces)

`@LEFT(123,1)` = ERR (123 is a value)

`@LENGTH(@LEFT("Jennifer",255))` = 8

@LENGTH(String)

@LENGTH returns the number of characters in *String*, including spaces. You can combine strings or cell addresses with an ampersand (&). When *String* is a text string, it must be enclosed by double quotes.

If you try to reference a blank cell with this @function, Quattro Pro returns ERR.

Examples

`@LENGTH("Hello, world.")` = 13

`@LENGTH(" Jennifer")` = 9 (including preceding space)

`@LENGTH("Greetings "&"earthling")` = 19 (including space after Greetings)

`@LENGTH(29584949)` = ERR (29584949 is a value, not a string)

`@LENGTH(A6&B10)` = total number of characters in A6 and B10

`@LENGTH(B10)` = ERR (if B10 is blank or a value)

@LN(X)

@LN returns the natural logarithm of *X*. A natural logarithm uses the mathematical constant e as a base. @LN produces the inverse of @EXP.

Examples

`@LN(3)` = 1.098612289

`@LN(@EXP(10))` = 10

`@LN(16)/@LN(2)` = 4

`@LN(-4)` = ERR (-4 is less than 0)

@LOG(X)

@LOG returns the base 10 logarithm of *X*.

Examples

`@LOG(1000) = 3`

`@LOG(10^23.8)` = 23.8

`@LOG(16)/@LOG(2)` = 4 (log to base 2 of 16)

@LOWER(String)

@LOWER returns *String* in lowercase characters. Numbers and symbols within a string are unaffected. Numeric and date values return ERR.

Examples

`@LOWER("UPPER")` = upper

`@LOWER("Hello, world.")` = hello, world.

`@LOWER("145 Bancroft Lane")` = 145 bancroft lane

`@LOWER(4839)` = ERR

`@LOWER(@LEFT("Johnson",1))` = j

@MAX(List)

@MAX returns the largest numeric or date value in *List*. If more than one block is listed, commas must separate the blocks. If any of the cells referenced contain ERR, the resulting value is ERR.

Examples

	A	B	C	D
1		Jan.	Feb.	Mar.
2	JA	$652	$833	$599
3	MH	$456	$305	$522
4	RB	$68	$59	$73
5	PD	$379	$379	$379
		------	------	------
6		$1,555	$1,576	$1,573

`@MAX(B3..B5)` = $456

`@MAX(C3..C5,D3..D5)` = $522

`@MAX(A1..D6)` = $1,576

`@MAX(B2..C5,D3)` = $833

@MEMAVAIL

@MEMAVAIL returns the number of bytes of memory currently available.

Example

`@MEMAVAIL` = 47819 (if 47,819 bytes of memory are available)

@MEMEMSAVAIL

This @function is included for compatibility with Quattro Pro for DOS; it always returns NA under Windows.

See Also
@MEMAVAIL

@MID(String,StartNumber,Num)

@MID extracts the first *Num* characters of *String* starting at *StartNumber*, which is the number of characters to the right of the first character (character 0). It is similar to @LEFT, which extracts *Num* characters of *String* beginning with the first character. The difference is that you can specify a character other than the first character in the string.

String can be any text string (enclosed by quotes) or reference to a cell containing a label. If *StartNumber* is greater than or equal to the length of *String* or if *Num* is 0, the result is "", or an empty string.

Examples

`@MID("Abraham Lincoln",8,7)` = Lincoln

`@MID("George Washington",7,4)` = Wash

`@MID("Theodore Roosevelt",19,5)` = ""

`@MID(A23,@FIND("Roosevelt",A23,0),@LENGTH("Roosevelt"))` = Roosevelt (if A23 = Franklin Roosevelt)

@MIN(List)

@MIN returns the smallest numeric value in *List*. If *List* contains more than one value, commas must separate the values. Labels are treated in all statistical functions as 0 and should therefore be excluded from *List*.

If *List* is entered as a block and one or more cells in that block are blank, the blanks are excluded from the calculation; otherwise, blanks are treated as 0.

Examples

	A	B	C	D
1		Jan.	Feb.	Mar.
2	JA	$652	$833	$599
3	MH	$456	$305	$522
4	RB	$68	$59	$73
5	PD	$379	$379	$379
		------	------	------
6		$1,555	$1,576	$1,573

`@MIN(B3..B6)` = $68

`@MIN(B2..D2,B4..D4)` = $59

`@MIN(B3..D3)` = $305

@MINUTE(DateTimeNumber)

@MINUTE returns the minute portion of *DateTimeNumber*. *DateTimeNumber* must be a valid date/time serial number. Because only the decimal portion of a serial number pertains to time, the integer portion of the number is disregarded. The result is between 0 and 59.

To extract the minute portion of a string that is in time format (instead of serial format), use @TIMEVALUE with @MINUTE to translate the time into a serial number. You can also use @TIME to enter a time value instead of a serial number.

Examples

`@MINUTE(.36554)` = 46

`@MINUTE(.2525)` = 3

`@MINUTE(35)` = 0

`@MINUTE(@TIME(3,15,22))` = 15

`@MINUTE(@TIMEVALUE("10:08 am"))` = 8

@MOD(X,Y)

@MOD divides the *X* value by *Y* and returns the remainder, or modulus, value. Because you cannot divide a number by zero, ERR results if the value of *Y* is zero.

Examples

`@MOD(3,1)` = 0 (3 divided by 1 leaves no remainder)

`@MOD(5,2)` = 1 (5 divided by 2 leaves a remainder of 1)

`@MOD(3,1.1)` = 0.8

`@MOD(4,0)` = ERR

@MONTH(DateTimeNumber)

@MONTH returns the month portion of *DateTimeNumber*. *DateTimeNumber* must be a valid date/time serial number. Only the integer portion is used. The result is between 1 (January) and 12 (December).

To extract the month portion of a string that is in date format (instead of serial format), use @DATEVALUE with @MONTH to translate the date into a serial number. You can also use @DATE to enter a date value instead of a serial number.

Examples

`@MONTH(69858)` = 4

`@MONTH(58494)` = 2

`@MONTH(.3773)` = 12

`@MONTH(@DATEVALUE("3/5/88"))` = 3

`@MONTH(@DATE(88,3,5))` = 3

`@MOD(@MONTH(@DATEVALUE("3/5/88")),12)` = 3

@N(Block)

@N inspects *Block* and returns the numeric value of the upper left cell. If that cell contains a label or is blank, it returns a 0.

This @function is used by other spreadsheet programs to avoid unnecessary ERR values resulting from labels included in calculations. This is unnecessary with Quattro Pro, however, because labels are already considered zero values in calculations. @N is included in Quattro Pro only for compatibility with other products.

@NA

@NA returns the special value NA (not available). Formulas that depend on a value entered as @NA return the value NA, unless there is an error, in which case they return ERR. NA is a unique number, not to be confused with the label NA.

@NA is used to indicate values not yet available (it won't work with labels). It ensures that formulas relying on information that is not provided don't display inaccurate data.

Examples

	A	B	C	D
1	QTR	North	South	West
2	1	$187,681	$151,136	$131,123
3	2	$170,072	NA	$149,181
4				
5	YTD	$357,753	NA	$280,304
6	AVG	$178,877	NA	$140,152

@NA has been entered for the South's Qtr 2 results. As you can see, the NA cascades through to the totals. When the @NA is replaced with a valid value, the totals will immediately reflect the correct figures.

`@NA` = NA

`@IF(B3=0,@NA,B3)` = NA if B3 = 0; otherwise, the value of B3

@NOW

@NOW returns the serial number corresponding to the current date and time. To display the number as a date or time, choose Numeric Format in the Block property menu.

The value generated by @NOW is updated to the current date and time each time you press the Calc key (F9), or perform any operation that recalculates the notebook.

The integer part of a date/time serial number pertains to the date; the decimal portion pertains to time. To extract just the date portion, use @INT(@NOW). To extract just the time portion, use @MOD(@NOW,1).

Examples

`@NOW` = 31905.572338 (5/8/87, 1:45 PM)

`@INT(@NOW)` = 31905 (5/8/87)

`@MOD(@NOW,1)` = 0.572338 (1:45 PM)

`@INT(@MOD(@NOW,7))` = 6 (the number of the day of the week)

@NPER(Rate,Pmt,Pv,<Fv>,<Type>)

Like @CTERM and @TERM, @NPER computes the number of payments needed to reach *Fv*, given *Pv*, *Pmt*, and *Rate*. The last two arguments of @NPER, *Fv* and *Type*, are optional. If you omit one or both of them, Quattro Pro assumes their values are zero.

Be sure to enter a negative number for money that's out of your pocket and a positive number for money that's coming in to you.

This @function isn't compatible with 1-2-3. If your file must be compatible, use @CTERM or @TERM instead.

Examples

Assume you have an IRA account that earns 11.5% interest paid annually at the start of the year, and you deposit $2000 into the account at the end of each year. The present account balance is $633. To determine how many payment periods it will take to reach a nest egg of $50,000, use @NPER:

`@NPER(11.5%,-2000,-633,50000,0)` = 12.12

The fractional part of the answer is not very meaningful; you can't be sure of having your nest egg until the end of the 13th year.

@NPV(Rate,Block,<Type>)

@NPV calculates the current value of estimated cash flow values in *Block*, discounted at *Rate*. It can help determine the current value of an investment, based on expected earnings.

The optional third argument, *Type*, can be 0 or 1, depending on whether the cash flows are at the beginning or the end of the period. The default value is 0, end of the period.

The cash flow table in *Block* should show expected income and debits over a period of time. Quattro Pro assumes that cash flow intervals are regular and the length of each interval is the same as the period on which interest is compounded. If monthly cash flow is estimated, *Rate* needs to show monthly interest. To convert annual interest to monthly interest, divide by 12.

Examples

	A	B	C	D	E
1	1992	1993	1994	1995	1996
2	-5000	+2000	+2000	+2000	+2000

Suppose you're considering investing $5000 this year, and you expect a return of $2000 in each of the next four years. Put the values -5000,+2000,+2000,+2000,+2000 in the block A2..E2. The net present value, using a discount rate of 10%, is `@NPV(.1,A2..E2,1)` which equals $1,340. Or, combine the initial investment with the present value of the four returns with `+A2+@NPV(.1,B2..E2,0)`. The result is the same.

Additional examples

	A	B	C	D
1	Jan	8000	200	3500
2	Feb	9000	350	4000
3	Mar	8500	-300	3000
4	Apr	9500	600	5000

`@NPV(1.25%,B1..B4)` = $33,908.92

`@NPV(15%/12,C1..C4)` = $820.83

`@NPV(15%/12,D1..D4)` = $15,006.51

`-2000+@NPV(15%/12,D1..D4)` = $13,006.51 (assumes an initial cash outflow of $2,000)

See Also
Entering Cash Flow @Functions

@NUMTOHEX(Decimal)

@NUMTOHEX converts the decimal number *Decimal* to its corresponding hexadecimal string value. @HEXTONUM performs the opposite conversion, from hexadecimal to decimal.

Examples

`@NUMTOHEX(10)` = 'A

`@NUMTOHEX(16)` = '10

`@NUMTOHEX(65535)` = 'FFFF

See Also

Entering Number Conversion @Functions

@PAYMT(Rate,Nper,Pv,<Fv>,<Type>)

@PAYMT calculates the periodic payment needed to reach *Fv*, given *Rate*, *Nper*, and *Pv*. The last two arguments of @PAYMT, *Fv* and *Type*, are optional. If you omit the last one or both of them, Quattro Pro assumes that their values are zero. Enter negative numbers for out-of-pocket money and positive numbers for money coming in.

Related @function @PMT isn't as flexible, but can be used if your file must be compatible with 1-2-3.

Examples

Assume you want to take out a 30-year $175,000 mortgage with a 17.5% annual interest rate with 12 payments a year, and you'd like to see the difference in your monthly payments if you paid at the start or at the end of the month. All you have to do is enter these two @functions:

`@PAYMT(17.5%/12,12*30,175000,0,0)` = -2566.07

`@PAYMT(17.5%/12,12*30,175000,0,1)` = -2529.19

If, on the other hand, your mortgage has a "balloon payment" that leaves you with unpaid principal at the end of the mortgage, you can still calculate the payment. Just insert the balloon payment amount (say, $80,000) as the future value component:

`@PAYMT(17.5%/12,12*30,175000,-80000,0)` = -2559.68

@PI

@PI returns the value of pi (3.141592653589794...), the classic ratio of a circle's circumference to its diameter.

To figure the area of a circle, given the radius in cell A1, enter this formula:

```
@PI*A1^2
```

Examples

`@PI*13` = 40.84 (circumference of circle with a diameter of 13)

`@PI*(7.5)^2` = 176.7146 (area of circle with a radius of 7.5)

`@PI*B3` = the circumference of a circle whose diameter is in B3

@PMT(Pv,Rate,Nper)

@PMT calculates the fully amortized periodic payment needed to repay a loan with a principal of *Pv* dollars at *Rate* percent per period over *Nper* periods. It assumes that interest is paid at the end of each period and the investment is an ordinary annuity (not an annuity due).

Rate must correlate with the unit used for *Nper*; if payments are monthly, *Rate* must equal the annual rate divided by 12.

You can enter the value for *Rate* as a percent or a decimal; for example, 9.5% or .095. The amount you specify for *Rate* must correlate with the unit used for *Nper*. In other words, if payments are made and interest calculated annually, the amount entered for *Nper* must represent years. If monthly, *Nper* must represent the number of months the loan covers. To calculate monthly payments using an annual interest rate, divide the interest rate by 12.

@PMT assumes that the investment is an ordinary annuity. Related @functions @PAYMT, @IPAYMT, and @PPAYMT let you use an optional argument, *Type*, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

To calculate a monthly payment (paid on the last day of the month) for a three-year loan of $10,000 at an annual 15% interest rate, enter

`@PMT(10000,15%/12,3*12)` = $346.65

You can also use @PAYMT to figure this payment (the negative result means the money is out of your pocket):

`PAYMT(15%/12,3*12,10000,0,0)` = $-346.65

Other examples:

`@PMT(1000,0.12,5)` = $277.41

`@PMT(500,0.16,12)` = $96.21

`@PMT(5000,16%/12,12)` = $453.65

`@PMT(12000,0.11,15)` = $1,668.78

`@PMT(10000,15%/12,36)` calculates a monthly payment for a three-year loan of $10,000 at an annual 15% interest rate

@PPAYMT(Rate,Per,Nper,Pv,<Fv>,<Type>)

@PPAYMT calculates the amount of a particular payment that is going toward the loan principal or investment *Pv* and is not interest.

@IPAYMT gives the part of the payment which is interest; @PAYMT calculates the total payment for each period.

Examples

Assume you're two years into a 30-year, 10% mortgage on a $100,000 loan. To determine what portion of this month's payment is principal, enter

`@PPAYMT(.1/12,2*12,30*12,100000)` = $-53.54

The negative result indicates the money is out of your pocket.

Another example:

`@PPAYMT (.15/4,24,40,10000,0,1)` = $-252.41 quarterly payments for a $10,000 loan at 15% annual percentage rate adjusted to a quarterly basis over a 10-year term

@PROPER(String)

@PROPER converts the first letter of every word in *String* to uppercase, and the rest of the characters to lowercase. A word is defined as an unbroken string of alphabetic characters. Any blank spaces, punctuation symbols, or numbers mark the end of a word.

Examples

`@PROPER("GEORGE washINGTON")` = George Washington

`@PROPER("FIRST QUARTER")` = First Quarter

`@PROPER("JOHN J.  SMITH")` = John J. Smith

`@PROPER("1979's results")` = 1979'S Results

`@PROPER(A1)` = John J. Smith (where cell A1 contains JOHN J. SMITH)

@PROPERTY(Object.Property)

@PROPERTY is similar to @COMMAND and @CURVALUE. It returns the current setting of *Property* for the requested *Object*.

Object.Property must be enclosed in double quotes. See Property Reference for lists of objects and properties you can enter as arguments.

@PROPERTY returns strings; even if the setting is a number, it is returned as a string.

@PROPERTY can be used in macros to read current settings so they can be restored at the end of the macro.

Like @CELL, @PROPERTY statements do not recalculate automatically. Press F9 to obtain the current value.

Examples

`@PROPERTY("Active_Block.Selection")` = the coordinates of the currently selected block.

`@PROPERTY("Active_Block.Protection")` = Protect if the currently selected block is protected; otherwise, it returns Unprotect.

`@PROPERTY("Sales:A1..D12.Protection")` = Protect if block A1..D12 on page Sales is protected; otherwise, it returns Unprotect.

`@PROPERTY("Graph8:G$Pane.Fill_Style")` = "Bitmap,Crop to fit,C:\QPW\tiger.bmp" if the Graph Pane Fill Style for Graph8 is a cropped bitmap named Tiger from directory C:\QPW.

`@PROPERTY("B4.Font.Typeface")` = Courier when cell B4 of the active page is set to display Courier type.

@PV(Pmt,Rate,Nper)

@PV calculates the present value of an investment where *Pmt* is received for *Nper* periods and is discounted at the rate of *Rate* per period.

@PV assumes that the investment is an ordinary annuity. Related @function @PVAL lets you use an optional argument, *Type*, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

Assume you want to buy a new van that costs $12,000. The dealer presents two offers: Pay $12,000 cash up front, or pay $350 per month for the next five years with 7% interest. The present value of the loan is

`@PV(350,7%/12,5*12)` = $17,675.70

The loan is worth over $5000 more than paying the cost all at once.

You can also use @PVAL. The car loan example becomes

`@PVAL(7%/12,5*12,-350,0,0)` = $17,675.70

Other examples:

`@PV(277,0.12,5)` = $998.52

`@PV(600,0.17,10)` = $2,795.16

`@PV(100,0.11,12)` = $649.24

@PVAL(Rate,Nper,Pmt,<Fv>,<Type>)

@PVAL calculates the present value of an investment where *Pmt* is received for *Nper* periods and is discounted at the rate of *Rate* per period.

Enter negative numbers for money that's out of your pocket and positive numbers for money coming in to you. The last two arguments, *Fv* and *Type*, are optional. If you omit the last one or both of them, Quattro Pro assumes their values are zero.

This @function isn't compatible with 1-2-3. If your file must be compatible, use the related @function @PV instead.

Examples

Your grandfather leaves you $24,000 in cash over the next 12 years ($2000 a year) or you can have all his government bonds, which mature in 15 years to a worth of $30,000. To determine which is worth more, compute the present value of the $24,000. Assume you can invest the money as you accumulate it in a 10% money market account.

`@PVAL(10%,12,2000,0,0)` = -13,627.38

The result is negative because the money you invest is considered an outgoing cash flow. Now compare this figure with the present value of the $30,000, which you won't receive for 15 years:

`@PVAL(10%,15,0,30000,0)` = -7,181.76

These results tell you that the $24,000 spread over 12 years is the more valuable choice.

@RADIANS(X)

@RADIANS converts the given number of degrees to radians, using this formula:

pi times *X* divided by 180

One degree is equal to approximately 0.017 radians.

Examples

`@RADIANS(1)` = 0.017453

`@RADIANS(57)` = 0.994838

`@RADIANS(@DEGREES(3.5))` = 3.5

`@RADIANS(A4)` = 0.994838 (where cell A4 contains the value 57)

@RAND

@RAND returns a fractional random number between 0 and 1. This offers a sampling of figures, useful for generating sample data for simulated situations.

To generate random numbers in another range, multiply @RAND by the difference between the new high and low ends, then add the new low end number. The formula is @RAND * (high number - low number) + low number.

For example, to indicate a range of 10 to 100, enter `@RAND*90+10`. This extends the upper limit to 100 and the lower limit to 10.

@RAND generates a new random number with each recalculation.

Examples

`@RAND` = a random number between 0 and 1

`@RAND*9+1` = a random number between 1 and 10

`@RAND*1000` = a random number between 0 and 1000

`@RAND+5` = a random number between 5 and 6

`-@INT(@RAND*90+10)` = a random integer between -10 and -100

@RATE(Fv,Pv,Nper)

@RATE calculates the interest rate required for an investment of *Pv* to be worth *Fv* within *Nper* compounding periods. If *Nper* represents years, an annual interest rate results; if *Nper* represents months, a monthly interest rate results, and so on.

@RATE assumes the investment is an ordinary annuity. The related @function @IRATE lets you use an optional argument, *Type*, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

This formula determines what yearly interest rate will double an initial investment of $2000 at the end of 10 years:

`@RATE(4000,2000,10)` = 7.18%

Other examples:

`@RATE(10000,7000,6*12)` = 0.50% (monthly)

`@RATE(1200,1000,3)` = 6.27% (yearly)

`@RATE(500,100,25)` = 6.65% (yearly)

@REPEAT(String,Num)

@REPEAT returns *Num* copies of *String* as one continuous label. This @function is similar to the repeating label prefix (\) in that it repeats one or more characters. The difference is that you can specify exactly how many times you want the string to be repeated. The \ label prefix adjusts the display to fill the column, even when the width is changed. @REPEAT displays a fixed number of copies of *String* and does not change.

When you specify a text string with @REPEAT, it must be enclosed by double quotes.

Examples

`@REPEAT("-",20)` = ----------

`@REPEAT("good day!",3)` = good day!good day!good day!

`@REPEAT(A5,5)` = the contents of A5 repeated 5 times

`@REPEAT("-",@CELL("width",A1..A1))` = ---------- if column A is 12 characters wide. If you change the column width, you can press F9 to adjust the repeat string to fill the cell.

@REPLACE(String,StartNum,Num,NewString)

@REPLACE lets you replace characters in text with a new text string. It searches through the given *String* from left to right beginning with the first character (character 0) until it reaches character position *StartNum*. Then it removes *Num* number of characters from the string, replacing them with *NewString*.

Both *String* and *NewString* can be either cell references or text strings. If text strings, they must be enclosed by double quotes.

To replace one string with another, specify 0 as *StartNum*. For *Num*, enter a number equal to or greater than the number of characters in *String*.

To insert one string into another string, specify 0 as *Num*.

To add one string to the end of another, specify as *StartNum* a number one greater than the number of characters in *String*.

To delete part or all of a string, specify "" as *NewString*.

Examples

`@REPLACE("McDougal Corp.",2,6,"Douglas")` = McDouglas Corp.

`@REPLACE("Leslie J. Cooper",7,3,"")` = Leslie Cooper

`@REPLACE("Sales Salaries",6,0, "Reps' ")` = Sales Reps' Salaries (There must be a space between Reps and the final quotation mark.)

`@REPLACE("355 Howard",11,0," St.")` = 355 Howard St. (There must be a space between " and St.)

You can use @REPLACE with other string functions. For instance, to replace one word with another within a sentence, you can use @FIND and @LENGTH to simplify the search-and-replace operation. For example,

```
@REPLACE(A7,@FIND("man",A7,0),@LENGTH(   "man"),"person")
```

searches through A7 for man, then replaces man with person.

@RIGHT(String,Num)

@RIGHT returns the last *Num* characters of *String* counting from right to left. It extracts a specified number of characters from the right side of a string or label.

If *String* is a not a valid string, @RIGHT returns ERR. If *Num* is 0, the result is "", or an empty string. If *Num* is greater than or equal to the number of characters in *String*, the entire string is returned.

Examples

`@RIGHT("Jennifer Meyer",5)` = Meyer

`@RIGHT("Jennifer Meyer",25)` = Jennifer Meyer

`@RIGHT("Jennifer    ",6)` = fer (including 3 subsequent spaces)

`@RIGHT("155",1)` = 5

`@RIGHT(123,1)` = ERR (123 is a value)

`@RIGHT(A10,5)` = the last five characters of A10

`@RIGHT(A16,@LENGTH(A16) - @FIND("Roosevelt",A16,0))` = Roosevelt (if A16 = Theodore Roosevelt)

@ROUND(X,Num)

@ROUND adjusts the precision of *X* to *Num* decimal places. *Num* specifies the power of 10 to which *X* is rounded. If *Num* is positive, *X* is rounded *Num* digits to the right of the decimal point. If *Num* is negative, *X* is rounded *Num* digits to the left of the decimal point. For example, if *Num* is -3, *X* is rounded to the nearest thousand.

If *Num* is 0, *X* is rounded to an integer. If *Num* is not an integer, it is truncated to an integer.

Examples

`@ROUND(12345.54321,0)` = 12346

`@ROUND(12345.54321,2)` = 12345.54

`@ROUND(12345.54321,-2)` = 12300

@ROWS(Block)

@ROWS returns the number of rows within the given block.

Examples

`@ROWS(A1..A1)` = 1

`@ROWS(A1..C15)` = 15

`@ROWS(B100..B8192)` = 8093

`@ROWS(NAME)` = 30 (if the block NAME contains 30 rows)

@S(Block)

@S returns the string value of the upper left cell of *Block*. If that cell contains a numeric or date value or is blank, it returns "" (an empty string).

If you enter a single cell address instead of a block, Quattro Pro changes it to a one-cell block (such as C3..C3). Quattro Pro also transforms cells prefixed with an exclamation point (as used in 1-2-3) to a one-cell range (!C3 changes to C3..C3).

Examples

	A	B	C	D
1	COMPANY	REP	SALES	COMMISSION
2	ABC Inc.	Jones	$123,630	$3,115
3	Rogers Co.	Marcus	$160,330	$4,040
4	Klein Sales	Wong	$145,330	$3,662

`@S(A1..A6)` = COMPANY

`@S(A2..A2)` = ABC Inc.

`@S(C2..C4)` = (blank)

`@S(B1..B1)&" = "&@S(B2..B2)` = REP = Jones

`@S(B3)&@S(C3)` = Marcus

@SECOND(DateTimeNumber)

@SECOND returns the second portion of *DateTimeNumber*. *DateTimeNumber* must be a valid date/serial number. Because only the decimal portion of a serial number pertains to time, the integer portion of the number is disregarded. The result is between 0 and 59.

To extract the second portion of a string that is in time format (instead of serial format), use @TIMEVALUE with @SECOND to translate the time into a serial number. You can also use @TIME to enter a time value instead of a serial number.

Examples

`@SECOND(.3655445)` = 23

`@SECOND(.2543222)` = 13

`@SECOND(35)` = 0

`@SECOND(@TIME(3,15,22))` = 22

`@SECOND(@TIMEVALUE("10:08:45 am"))` = 45

`@SECOND(@TIMEVALUE("10:08 am"))` = 0

@SHEETS(Block)

@SHEETS returns the number of pages within *Block*. This @function is similar to @COLS and @ROWS.

Examples

`@SHEETS(B:A1..D:IV1)` = 3

`@SHEETS(A1..C7)` = 1

`@SHEETS(A..E:NAME)` = 5

@SIN(X)

@SIN returns the sine of the angle *X*. *X* must be given in radians, not degrees. To convert degrees to radians, use @RADIANS.

Examples

`@SIN(@RADIANS(30))` = 0.5

`@SIN(@PI/6)` = 0.5

@SLN(Cost,Salvage,Life)

@SLN calculates the straight-line depreciation allowance for an asset over one period of its life, using this formula:

(*Cost* - *Salvage*) divided by *Life*

To compute accelerated depreciation with the sum-of-the-years'-digits method (allowing higher depreciation values in the first years of the asset's life), use @SYD. To calculate depreciation using the double-declining balance method, use @DDB.

Examples

Assume you just bought a new $4000 computer. The dealer says you can sell it back to the store for $350 after eight years, but that no one would want to buy it after that. In other words, the Salvage value of that computer is $350 and its Life is 8. To determine the depreciation allowance of the computer for each year of its life, enter this formula:

`@SLN(4000,350,8)` = 456.25

Other examples:

`@SLN(15000,3000,10)` = $1,200

`@SLN(5000,500,5)` = $900

`@SLN(1800,0,3)` = $600

@SQRT(X)

@SQRT returns the square root of *X*. If *X* is a negative value, the result of @SQRT is ERR. If *X* is a string or reference to a cell containing a label, the @function returns 0.

Examples

`@SQRT(9)` = 3

`@SQRT(2)` = 1.414213562

`@SQRT(144)` = 12

`@SQRT(@SQRT(16))` = 2

`@SQRT(-4)` = ERR

@STD(List)

@STD returns the population standard deviation (the square root of the population variance) of all values in *List*. @STDS computes the standard deviation of sample data.

List can be any combination of single cell references, cell blocks, and numeric values. When more than one component is used, all components must be separated by commas. @STD treats any labels within a cell block as zero and ignores any blank cells. If the *List* contains only blank cells, however, @STD returns ERR.

@STD determines how much individual values in *List* differ from the average (mean) of all values in *List*. It can be used to verify the reliability of the average; the lower the value returned by @STD, the less individual values vary from the average.

Examples

	A	B	C	D
1		January	February	March
2	John	$652	$833	$599
3	Mary	$456	$305	$522
4	Ralph	$68	$59	$73
5	Anna	$80	$80	$80
6				

`@STD(B4..D4)` = $5.79

`@STD(C2..C5,260)` = $279.97

`@STD(B2..D5)` = $270.20

`@STD(A15..D20)` = ERR (because the entire block is blank)

`@STDS(B4..D4)` = $7.09

`@STDS(B2..D5)` = $282.22

@STDS(List)

@STDS returns the sample standard deviation (the square root of the sample variance) of all values in *List*. @STD computes population standard deviation.

This @function isn't compatible with 1-2-3. If your file must be compatible with 1-2-3, use @STD instead.

Examples

	A	B	C	D
1		January	February	March
2	John	$652	$833	$599
3	Mary	$456	$305	$522
4	Ralph	$68	$59	$73
5	Anna	$80	$80	$80
6				

`@STD(B4..D4)` = $5.79

`@STD(C2..C5,260)` = $279.97

`@STD(B2..D5)` = $270.20

`@STD(A15..D20)` = ERR (because the entire block is blank)

`@STDS(B4..D4)` = $7.09

`@STDS(B2..D5)` = $282.22

@STRING(X,DecPlaces)

@STRING converts *X* to a string, rounding *X* to the decimal precision indicated by *DecPlaces*.

Once a number or date has been converted to a label using @STRING, no display formatting can be done with it. To format strings derived from numbers as anything other than General format, you must build a macro that uses the {CONTENTS} keyword.

Examples

`@STRING(3.59,0)` = 4

`@STRING(98.6,2)` = 98.60

`@STRING(0.3902,0)` = 0

`@STRING("Harry",0)` = 0

`@STRING(A1,2)` = 10.00 (where A1 = 10)

@SUM(List)

@SUM returns the total of all numeric values in *List*. *List* can be any combination of single cell references, cell blocks, and numeric values. When more than one component is used, they must be separated by commas. Any labels or blank cells within a cell block are ignored by @SUM.

Any dates in the block will be converted to serial numbers and included in the calculation. Since this will throw off your sum, avoid including dates in the @SUM argument block.

If you use a mouse, the SpeedBar SpeedSum button offers a convenient way to total columns, rows, or both. It can total rows and columns in the selected block, but you don't need to enter a formula.

Examples

	A	B	C	D
1		January	February	March
2	John	$652	$833	$599
3	Mary	$456	$305	$522
4	Ralph	$68	$59	$73
5	Anna	$80	$80	$80
6				

`@SUM(B4..D4)` = $200

`@SUM(C2..C5,260)` = $1,537

`@SUM(A5,534)` = $534

`@SUM(B2..B5,D2..D5)` = $2,530

`@SUM(B2..D5)` = $3,807

@SUMPRODUCT(Block1,Block2)

@SUMPRODUCT(*Block1*, *Block2*) returns the dot product of the vectors corresponding to the blocks. Quattro Pro multiplies each corresponding cell from *Block1* and *Block2* and then totals those results. The blocks must be the same size (same number of rows and same number of columns), or else the blocks must both be one-dimensional (either a row or a column) and they must have the same length. If the blocks don't match, @SUMPRODUCT returns ERR.

This @function isn't compatible with 1-2-3. If your notebook must be compatible with 1-2-3, don't use @SUMPRODUCT.

Examples

Assume the following values for these cells:

A1 = 1, B1 = 5, A2 = 2, B2 = 6, A3 = 3, B3 = 7, A4 = 4, B4 = 8

`@SUMPRODUCT(A1..A2,B1..B2)` = 17 (because 1*5 + 2*6 = 5+12 = 17)

`@SUMPRODUCT(A1..A4,B1..B4)` = 70

`@SUMPRODUCT(A1..A4,B1..B5)` = ERR (blocks are not the same size)

@SYD(Cost,Salvage,Life,Period)

@SYD calculates depreciation amounts for an asset using an accelerated depreciation method. This allows higher depreciation in the earlier years of the asset's life. @SYD uses this formula to compute depreciation:

((*Cost* - *Salvage*)(*Life* - *Period* + 1)) divided by (*Life*(*Life* + 1)/2)

Cost must be equal to or greater than *Salvage*; both must be equal to or greater than 0. *Life* must be equal to or greater than *Period*; both must be equal to or greater than 1.

@DDB and @SLN offer other methods of calculating depreciation.

Examples

Assume you just bought a new $4000 computer. The dealer says you can sell it back to the store for $350 after eight years, but that no one would want to buy it after that. The Salvage value of that computer is $350 and its Life is 8. To see what the depreciation allowance of this computer will be by the second year (using this method of depreciation), enter this formula:

`@SYD(4000,350,8,2)` = 709.72

These examples show depreciation values for the first five years of an asset's life. These can be compared to those calculated with @DDB, which distributes more of the depreciation in the first year of life.

`@SYD(12000,1000,5,1)` = $3,667

`@SYD(12000,1000,5,2)` = $2,933

`@SYD(12000,1000,5,3)` = $2,200

`@SYD(12000,1000,5,4)` = $1,467

`@SYD(12000,1000,5,5)` = $733

`@DDB(12000,1000,5,1)` = $4,800

`@DDB(12000,1000,5,2)` = $2,880

`@DDB(12000,1000,5,3)` = $1,728

`@DDB(12000,1000,5,4)` = $1,037

`@DDB(12000,1000,5,5)` = $555

@TAN(X)

@TAN returns the tangent of the angle *X*. *X* must be given in radians, not degrees. To convert degrees to radians, use @RADIANS.

Examples

`@TAN(4)` = 1.157821

`@TAN(@PI/4)` = 1

`@TAN(@RADIANS(45))` = 1

@TERM(Pmt,Rate,Fv)

@TERM computes the number of payment periods required in order to accumulate an investment of *Fv*, making regular payments of *Pmt* and accruing interest at the rate of *Rate*.

@TERM assumes the investment is an ordinary annuity. The related @function @NPER uses an optional argument, *Type*, to indicate whether the investment is an ordinary annuity or an annuity due.

Examples

To determine how long it will take to accrue $50,000 by depositing $2000 at the end of each year into a savings account that earns 11% annually, enter this formula:

`@TERM(2000,11%,50000)` = 12.67

Quattro Pro determines that it will take 12.67 years to accumulate $50,000 in your account. (Depending upon how your bank pays interest, your balance might not exceed $50,000 until the end of the 13th year.)

If, on the other hand, the money is not coming in to you but is being paid out by you, you can enter the future value as a negative number.

You can also use @NPER to calculate this example:

`@NPER(11%,-2000,0,50000,0)` = 12.67

Other examples:

`@TERM(300,6%,5000)` = 11.9 years

`@TERM(500,7%,1000)` = 1.94 years

`@TERM(500,.07,1000)` = 1.94 years

`@TERM(1000,10%,50000)` = 18.8 years

`@TERM(100,5%,1000)` = 8.3 years

@TIME(Hr,Min,Sec)

@TIME returns the date/time serial number represented by *Hr*:*Min*:*Sec*. Any fractional portions of *Hr*, *Min*, and *Sec* are rounded. You can display the resulting time string values in standard time formats by choosing Numeric Format in the Block property menu.

Examples

`@TIME(3,0,0)` = 0.125 (3:00 am)

`@TIME(3,30,15)` = 0.14600694444 (3:30:15 am)

`@TIME(18,15,59)` = 0.76109953704 (6:15:59 pm)

`@TIME(B15,23,45)` = 0.099826388889 (when the value in B15 is 2)

`@TIME(@HOUR(C3),A4,B10)` = 0.5751388889 (1:48:12 pm) (when C3 = 01:23:13 pm (formatted date/time serial number), A4 = 48, and B10 =12)

@TIMEVALUE(TimeString)

@TIMEVALUE returns a serial time value that corresponds to the value in *TimeString*. If the value in *TimeString* is not in the correct format, or is not enclosed in quotes (if entered as a literal string), an ERR value is returned.

You can display resulting time string values in standard time formats by choosing Numeric Format in the Block property menu.

There are four valid formats for *TimeString*:

HH:MM:SS AM/PM (03:45:30 PM)

HH:MM AM/PM (03:45 PM)

The Long International time format chosen as a system default, one of which is HH:MM:SS (15:45:30)

The Short International time format chosen as a system default, one of which is HH:MM (15:45)

Examples

`@TIMEVALUE("03:30:15 AM")` = 0.1460069444

`@TIMEVALUE("03:00")` = 0.125

`@TIMEVALUE("18:15:59")` = 0.76109953704

`@TIMEVALUE("3.45")` = ERR

`@TIMEVALUE(@TIME(12,30,45))` = 0.521354

`@TIMEVALUE(A1)` = 0.125 if A1 contains the label '03:00

@TODAY

@TODAY enters the numeric value of the system's date. It is equal to the expression @INT(@NOW).

@TRIM(String)

@TRIM removes any extra spaces from *String*; that is, spaces following the last nonspace character or preceding the first nonspace character, and duplicate spaces between words. Strings with no extra spaces are not affected. If *String* is empty or contains a numeric value, it returns ERR.

Examples

`@TRIM(" too    many    spaces ")` = "too many spaces"

`@TRIM("no extra spaces")` = "no extra spaces"

`@TRIM(125)` = ERR

@TRUE

@TRUE returns the logical value 1 and is usually used in @IF formulas. The 1 it returns is the same as the regular numeral 1, but @TRUE makes the formula easier to read.

See Also
@FALSE.

Examples
`@TRUE` = 1

`@IF(C3=100,@TRUE,10)` = 1 (if C3 = 100) or 10 (if C3 is not equal to100)

`@IF(C3=100,@TRUE,@FALSE)` = 1 (if C3 = 100) or 0 (if C3 is not equal to 100)

@UPPER(String)

@UPPER returns *String* in uppercase characters. Numbers and symbols within a string are unaffected. If *String* is blank, or contains a numeric or date value, the result is ERR.

Examples

`@UPPER(4839)` = ERR

`@UPPER(@LEFT("johnson",1))` = J

`@UPPER("upper")` = UPPER

`@UPPER("Hello, world.")` = HELLO, WORLD.

`@UPPER("145 Bancroft Lane")` = 145 BANCROFT LANE

@VALUE(String)

@VALUE converts *String* into a numeric value. *String* can contain arithmetic operators (but don't place arithmetic operators within quotes). *String* must not contain embedded spaces. Dollar signs, commas, and leading and trailing spaces are ignored.

This @function is useful for converting imported data that has not already been converted into values.

Examples

`@VALUE("  3.59")` = 3.59 (leading spaces are stripped)

`@VALUE("  98.6  ")` = 98.6 (leading and trailing spaces are stripped)

`@VALUE("98.6 4")` = ERR (an embedded space is not allowed)

`@VALUE(3+4)` = 7

`@VALUE("3+4")` = ERR (arithmetic operators within quotes are not allowed)

`@VALUE(" 88.039")` = 88.039

`@VALUE(A10)` = 56.34 (where cell A10 = '$56.34)

`@VALUE("34,200")` = 34200

`@VALUE(A1)` = ERR (where cell A1 = '1800 Green Hills Road)

@VAR(List)

@VAR calculates the population variance of all nonblank, numeric cells in *List*, using the n method (biased). Use @VARS to compute the variance of a data sample.

If *List* contains text, a reference to a single cell containing a label (for example, `@VAR(B1)` where B1 = Adam), or label cells within references to multiple cell blocks (such as `@VAR(B1..B5)`), @VAR treats the string as having a value of 0. @VAR ignores blank cells within a referenced block of cells, but returns ERR if every cell in the block is blank.

Examples

`@VAR(23,24,25)` = .666666667

`@VAR("Adam",53)` = 702.25 (same as for @VAR(0,53)

`@VAR(B1..B4)` = 54.6875 (if B1=10, B2=15, B3="Susan", B4=20; the string in B3 is treated as if it were 0)

`@VARS(23,24,25)` = 1

`@VARS("Adam",53)` = 1404.5 (same as for @VARS(0,53)

`@VARS(B1..B4)` = 72.9167 (if B1=10, B2=15, B3="Susan", B4=20; the string in B3 is treated as if it were 0)

@VARS(List)

@VARS calculates the sample variance of all nonblank, numeric cells in *List*, using the n-1 method (unbiased). @VAR computes population variance.

This @function isn't compatible with 1-2-3. If your file must be compatible with 1-2-3, use @VAR instead.

@VERSION

@VERSION returns the version number of Quattro Pro.

@VLOOKUP(X,Block,Col)

@VLOOKUP works along the same basic principles as @HLOOKUP, except that rows and columns are reversed.

@VLOOKUP searches (vertically) down the first column of *Block* for value *X*. When found, it returns the value itself (if *Col* = 0), or the value displayed the specified number of columns to the right (as indicated by *Col*).

X can be a character string or a number, the address or block name of any cell containing a label or value, or any expression that results in a number or string. If *X* is a string, the match must be exact; the lookup is case-sensitive. If *X* is a number and @VLOOKUP can't find an equal number, it locates the highest number in the column not greater than *X*.

The second argument (*Block*) specifies the coordinates of the table to be used for the lookup. This table must have its index values in the leftmost column. These values (if numbers) must be in ascending numerical order. Also, there must be no blank cells in the index column. Blanks in the table to the right of the index column are treated as 0.

If *X* is a string value and Column = 0, @VLOOKUP returns the offset number of the row *X* is found in, not the value of *X*.

These instances result in ERR:

Column is less than 0 or greater than the number of columns in *Block*.

X is less than the smallest value in the first column of *Block*.

X is a string and an exact match in the index column is not found.

Examples

	A	B	C	D
1	5	52	84	43
2	10	32	67	45
3	15	42	18	22
4	20	83	76	47

In the first example, @VLOOKUP searches down the first column of the specified block (column A), looking for the largest number equal to or less than 17. It stops at cell A3, then moves across the specified number of columns (3). It stops at cell D3 and returns the value 22.

`@VLOOKUP(17,A1..D4,3)` = 22

`@VLOOKUP(10,A1..D4,0)` = 10

`@VLOOKUP(50,A1..D4,3)` = 47

`@VLOOKUP("18",A1..D4,2)` = ERR (no labels in block)

`@VLOOKUP(18,A1..D4,8)` = ERR (col value > # cols)

`@VLOOKUP(18,A1..C4,3)` = ERR (col value > # cols in given block)

To search horizontally through a table, use @HLOOKUP.

@YEAR(DateTimeNumber)

@YEAR returns the year portion of *DateTimeNumber*. The result will be between 0 (1900) and 199 (2099). To display the actual year, just add 1900 to the result of @YEAR. If you want to extract the year portion of a string that is in date format, use @DATEVALUE with @YEAR to convert the string into a serial number.

Examples

`@YEAR(22222)` = 60 (1960)

`@YEAR(A6)+1900` = 19nn, where nn is the year value in A6

`@YEAR(@DATEVALUE("12-Oct-54"))` = 54

New @Functions

In addition to the existing eight categories of functions (mathematical, statistical, database, logical, financial, date and time, string, and miscellaneous), there is a new category: engineering.

This section lists new @functions by category. Refer to the New @Functions Index for details and examples. For the complete listing of @functions, see @Functions Index.

New Date and Time @Functions

@ABDAYS	Adds (or subtracts) a given number of business days to a given date.
@ACDAYS	Adds a given number of calendar days to a given date.
@AMNTHS	Adds a given number of months to a given date.
@BDAYS	Returns the number of business days between two dates, inclusive of the second date.
@BUSDAY	Returns a given date if it is a business day, or the closest business day before (or after) the date.
@CDAYS	Returns the number of calendar days between two dates, inclusive of the second date.
@EMNTH	Returns the date of the last day of the month in which a specified date falls.
@FBDAY	Returns the date of the first business day of a month in which a specified date falls.
@HOLS	Returns the number of holidays between two dates, excluding holidays that fall on weekends.
@ISBDAY	Returns 1 if the specified date is a business day, or 0 if it is not.
@LBDAY	Returns the date of the last business day of a month in which a specified date falls.
@LWKDAY	Returns the date of the last given weekday in a given month.
@MDAYS	Returns the number of calendar days in a given month of a given year.
@MNTHS	Returns the number of whole months between two dates.
@NBDAY	Returns the date of the first valid business day after a given date.
@NWKDAY	Returns the date of the nth occurence of a given weekday in a given month.
@PBDAY	Returns the date of the first valid business day before a given date.
@WKDAY	Returns the day of the week that a given date falls on.
@YDAYS	Returns the number of calendar days in a given year.
@YDIV	Returns the date of the beginning of the year division in which a given date or given number of divisions always falls.
@YEARFRAC	Returns the year fraction representing the number of whole days between a given starting and ending date.

See Also

New @Functions

Setting Holidays

@Functions and Formulas

Entering Formulas

@Function Arguments

Setting Holidays

Business date functions have three arguments to specify which dates are holidays: Saturday, Sunday, and Holidays. By default, Saturday and Sunday are holidays. Setting the argument Saturday to 1 specifies that Saturday is a business day; setting Sunday to 1 specifies that Sunday is a business day. Use the argument Holidays to specify holidays that don't fall on weekends (unless weekends are used as business days). You can set Holidays to a block containing holiday dates (see the next figure), the date of a single holiday, or 0 to specify no special holidays.

	A	B	C
7	01/01/93	07/05/93	11/26/93
8	02/15/93	09/06/93	12/23/93
9	05/31/93	11/25/93	12/24/93

See Also

New Date and Time @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Entering Number Conversion @Functions

Input numbers that include non-numeric characters (such as hexadecimal or ASCII values) must be enclosed in quotation marks (for example, "1AF3C"). Numbers that exceed 14 digits, except decimal numbers, must also be enclosed in quotation marks.

For the 64-bit number conversion @functions, numbers must be in the following ranges for each base:

64-bit Base Ranges

Base	Range
Signed decimal	-9223372036854775808 to +9223372036854775807
Unsigned decimal	0 to 18446744073709551615
Hexadecimal	0000000000000000 to FFFFFFFFFFFFFFFF
Octal	0000000000000000000000 to 1777777777777777777777

If a 64-bit number conversion @function results in a string value greater than , the @function returns ERR.

See Also

Number Conversion Engineering @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Entering Boolean @Functions

The shift @functions, @SHLB, @SHRB, @SHLH, and @SHRH, shift the bits in a number to the left or right. You can use the shift @functions to perform quick multiplication of integers. Each binary shift to the left is equivalent to multiplying the number by 2. Each binary shift to the right is equivalent to dividing the number by 2.

Binary		Decimal
00011	=	3
00110	=	6
01100	=	12
11000	=	24

As you shift bits off one end of a number, bits are added to the other end. Shift, however, is not equal to rotate. For example, the binary number 1001 shifted left is 0010. The same number rotated left is 0011. To perform this rotation, you can use a nested bit test @function to set the BitIn argument:

```
@SHLB(1001,@BITTB(1001,3),1,4) = 0011
```

The addition @functions, @ADDB and @ADDH, return the sum of two numbers. The subtraction @functions, @SUBB and @SUBH, return the difference of two numbers.

Note: To add or subtract negative numbers with the Boolean @functions, use two's complement notation. Two's complement notation is a code used to give meaning to a binary string. The sign bit of two's complement notation is the leftmost bit of the word. A number is positive if the sign bit is 0 and negative if it is 1. Negative integers are converted to two's complement by inverting each bit (that is, changing each 1 to a 0 and each 0 to a 1), and then adding 1.

The overflow @functions, @SHLBO, @SHRBO, @SHLHO, @SHRHO, @ADDBO, @ADDHO, @SUBBO, @SUBHO, return the overflow bit (either 0 or 1) of a shift, addition, or subtraction operation. For example, if you shift the binary number 1001 to the left and maintain four places, the result is 0010. The overflow bit, that is, the bit that has shifted into the fifth place not shown, is 1.

The AND @functions, @ANDB and @ANDH, combine the zeros of the input words. Any bit that is 0 in either number sets the corresponding output bit to 0.

The OR @functions, @ORB and @ORH, combine the ones of the input words. Any bit that is 1 in either number sets the corresponding output bit to 1.

The exclusive OR @functions, @XORB and @XORH, set an output bit to 1 when two corresponding input bits are not equal.

The following table shows the results of AND, OR, and exclusive OR operations of Bit A and Bit B.

Bit A	Bit B	A AND B	A OR B	A XOR B
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

The invert @functions, @INVB and @INVH, invert individual bits of a number; that is, all bits with a value of 1 change to 0, and all bits with a value of 0 change to 1.

The bit manipulation @functions, @BITSB, @BITRB, @BITTB, @BITSH, @BITRH, and @BITTH, let you set or reset bits, or test the value of a bit in a number.

The concatenation @functions, @CATB, @CATH, @CATNB, and @CATNH, link numbers together in

a chain. For example, the concatenation of the two binary numbers 11111 and 10101 is 1111110101.

The argument Bits defines the word size (in number of binary bits) for both input and output. The default for Bits is the number of bits in the largest input number. If you perform an operation with two numbers of different word size, the smaller number is left-padded with zeros. If Bits is less than the length of an input value, then the excess most significant digits are truncated. For Boolean @functions using hexadecimal numbers, note that each hexadecimal digit equals 4 bits.

The argument BitIn represents either the binary bit inserted during a shift, the carry bit for addition, or the borrow bit for subtraction; it can be either 0 (the default) or 1.

See Also

Boolean Engineering @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New Financial @Functions

Annuity	The investment @functions involve a series of periodic payments over a term measured in the number of payment periods. This set of @functions allows you to compute one value, knowing three of the other values.
Bill	These @functions compute values for Treasury billls.
Bond	These @functions compute values for bonds.
Cash Flow	These @functions operate on tables of data that record income and expenditures.
CD	These @functions compute values for certificates of deposit.
Stock	These @functions compute values for common stock.

See Also

Calendar Conventions

New @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Calendar Conventions

Financial @functions support four different calendar conventions to count the difference in days between two dates. The optional Calendar argument lets you specify which calendar convention to use.

Calendar	Description
30/360	The 30/360 calendar convention assumes all months have 30 days and every year has 360 days. Using the 30/360 calendar, the number of years, months and days between two dates are counted separately. Then, the number of days between two dates is the sum of three quantities: the number of years times 360, the number of months time 30, and the number of days.
Actual/Actual	The Actual/Actual calendar convention considers the actual number of days between two dates and the actual number of days in the year. For example, February 28, 1994 and August 31, 1994 are 184 days apart. February 28, 1994 and March 1, 1994 are 1 day apart.
Actual/360	The Actual/360 calendar convention considers the actual number of days in each month, but assumes 360 days in the year.
Actual/365	The Actual/365 calendar convention considers the actual number of days in each month, but assumes 365 days in the year, thus making no provision for leap year.

New Annuity @Functions

@AMAINT	Calculates the accumulated interest paid on an amortized loan after n payments.
@AMINT	Calculates the periodic interest rate for an amortized loan.
@AMPMT	Calculates the periodic payment for an amortized loan.
@AMPMTI	Calculates the interest portion of the nth periodic payment of an amortized loan.
@AMPRN	Calculates the initial principal of an amortized loan.
@AMRES	Calculates the end value of an amortized loan or the future value of an annuity.
@AMRPRN	Calculates the remaining balance of an amortized loan after n payments.
@AMTERM	Calculates the length of an amortized loan, expressed as number of payments.
@MTGACC	Returns the new loan term, the payoff-date, or interest saved by paying extra monthly principal for a home loan.
@YLD2YLD	Converts a yield expressed in one compounding frequency and time length to that in another frequency and/or time length.

See Also

Annuity @Function Arguments
Calendar Conventions
New @Functions
New Financial @Functions
@Functions and Formulas
Entering Formulas

Annuity @Function Arguments

Argument	Description
Adv	Number of cash flows (payments, deposits) made before the annuity begins.
Fv	Future value.
Int	Interest charged on the loan per period (not per year; many loans quote annual interest rates that must be divided by the number of payments per year).
Nper	Number of periods of the loan or investment (should be an integer greater than 0).
Odd	Number ≥ 0 that specifies the number of periods between the start of a loan and the first payment (for example, if the loan is made two and a half months before the first monthly payment is due, use 2.5)
Payment	Cash flow made each period.
Per	A specified loan or investment period, 1 thorugh Nper.
Pmt	Payment.
Principal	Amount of money loaned or the initial deposit on an annuity that increases principal (like depositing $2,500 to open a savings account)
Pv	Present value.
Rate	Interest rate (should be greater than -1).
Residual	Remaining principal and interest at the end of a loan that the annuity didn't take care of
ResOff	Number of periods after the annuity ends before the residual must be paid; express it as a fraction of a period (for example, in a monthly loan, 1.5 means 1.5 months before the residual is due)
Simp	Specifies how the interest is calculated: 0 for compounded interest, 1 for simple interest
Term	The total number of cash flows (payments or deposits) to make
Type	0 if payments are at the end of each period, 1 if at the beginning. This optional argument lets you use financial @functions to compute either an ordinary annuity, where periodic payments are made at the end of each period, or an annuity due, where payments are made at the beginning of each period. Quattro Pro assumes that Type = 0 unless you indicate otherwise.

Note: In the following @functions, as well as @NPV and @IRR, amounts with positive signs represent money received, and amounts with negative signs represent money paid: @FVAL, @IRATE, @IPAYMT, @NPER, @PAYMT, @PPAYMT, and @PVAL. This convention applies to arguments and to the results of the @functions. In 1-2-3-compatible @functions (such as @PV, @PMT, @FV, @RATE, @TERM, and @CTERM) the amounts are usually all positive regardless of which way the money changes hands.

Non-integer values are allowed for Nper, and the @functions give results that are consistent with other spreadsheet programs, but which are actually not very meaningful. If you borrow money from a bank for, say, 15.2 months with interest paid monthly, giving Nper a value of 15.2 in the financial @functions will only be a rough indicator of what the bank will tell you to pay. In order to compute the figures the way the bank would, you have to consider two transactions, one for 15 months and one for 0.2 months.

The functions asume that there is no residual unless a nonzero value is specified for the optional argument Residual. When a residual is specified, the functions assume that it's paid along with the last

payment. When it's not, a positive value should be specified for the optional argument ResOff. For example, if the residual is paid three months after the last monthly payment, ResOff = 3. Compound interest is used during any fractional component of ResOff unless Simp = 1.

Advance payments, specified by Adv, are made on or before the first day of the loan period. They're included in the total payment count. The functions assume zero advance payments unless a nonzero value is specified for the optional argument Odd.

Odd specifies the time period between the beginning of a loan (or issue of an annuity) and the date of the first periodic payment, and does not necessarily constitute exactly one normal payment period. For example, if a loan begins on March 19, 1993 and monthly payments are due the first of every month beginning April 5, 1993, the first payment period is 17 days long. Since the implied normal first payment period, March 5 to April 5, is 31 days long, Odd = ⁄ .

See Also

Annuity Financial @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New Cash Flow @Functions

@DURAT	Calculates the Macaulay duration of a cash flow stream.
@FUTV	Calculates the future value of a cash flow stream.
@NETPV	Calculates net present value of a stream of cash flows.
@PIRATE	Calculates the internal rate of return for a stream of cash flows.
@SCMARG	Calculates the discount scenario margin, the margin to add to each discount rate in order to arrive at a given net present value.

See Also

Calendar Conventions
New @Functions
New Financial @Functions
Entering Cash Flow @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Entering Cash Flow @Functions

@IRR and @NPV are similar to annuity @functions; they assume value changes based on Rate. These @functions differ from annuity @functions because they operate on tables of data that record income and expenditures. @IRR calculates the internal rate of change, while @NPV calculates the current value of cash flow values given a set interest rate.

Cash flow analysis is a process of listing a stream of cash gains and losses (positive and negative cash flows), modifying them using a percentage or percentages (discount rate(s)), and determining their future value, present value, or rate of growth (or decline; both are called the internal rate of return).

One example of a cash flow stream is a savings account: the stream consists of deposits (positive cash flows) and withdrawals (negative cash flows) in chronological order. The current balance is the net present value of the cash flows.

If you decided to set up a what-if scenario by adding estimated deposits and withdrawals to the savings account, this estimated balance is the future value of the stream. Interest is added to the account using percentages called discount rates. A discount rate doesn't just reduce a cash flow (like a fee); it can also increase a cash flow (like accruing interest).

Unlike an annuity, this stream of cash flows doesn't always occur periodically, and doesn't have a fixed interest rate for each cash flow.

You can use cash flow functions to estimate the net present value of a cash flow stream, project the future value of the stream, compute the gains the stream is making as a percentage, or compute how the discount rates must change to achieve a specific future value.

In Quattro Pro, a stream of cash flows is specified by a column (or row) of values. The Flows argument of a cash flow function is set to this block. Positive values add cash to the stream; negative values subtract from it. For example, if your savings account had two deposits of $50, one withdrawal of $25, and one deposit of $75 (in that order), you could use A2..A5 or B2..E2 of the next figure to represent it in a cash flow function:

	A	B	C	D	E
1	Cash Flows				
2	$50	$50	$50	($25)	$75
3	$50				
4	($25)				
5	$75				

If the stream contains a series of equal cash flows, you can add an additional column (or row) to specify how many times a given cash flow repeats. For example, you can replace A2..A5 of the previous figure with A2..B4 of the next figure:

	A	B
1	Cash Flows	
2	2	$50
3	1	($25)
4	1	$75

The first column (or row) of the block specifies how many times each cash flow occurs. For example, in the previous figure the value 2 (in A2) specifies that two cash flows of $50 occur in the stream, not one.

Note: Quattro Pro uses the size of the cash flow block to determine whether you're specifying a column of cash flows or a row of cash flows. It assumes that blocks with more than two rows contain cash flows in the second column; blocks with more than two columns contain cash flows in the second row. In the case of a two-column, two-row block, Quattro Pro assumes that the cash flows are in the second row.

You can use the argument, Filter, Start, and End to make Quattro Pro automatically exclude cash flows that don't fall in a certain range, such as all deposits, or any withdrawals less than $20. Excluded cash flows aren't included in the function calculations. Use Filter to specify the rules for exclusion, as shown in the next table.

Filter	Cash flows are excluded when
0	No filtering (the default)
1	Cash flow < Start
2	Cash flow ≤ Start
3	Cash flow > Start
4	Cash flow ≥ Start
5	Start < Cash flow < End
6	Start ≤ Cash flow ≤ End

As shown, Start and End are used differently, depending on the setting of Filter. They always bind the cash flows in some way; Start and End could be a range of cash flows values to use (Filter set to 5) or an upper limit for values (Filter set to 1, Start set to the upper limit).

The Discrate argument of a cash flow function specifies how the cash flows are discounted to achieve their future or net present value. It can be a single percentage (like 0.05 for 5%) that applies to all the cash flows, or a column (or row) of discount rates, one for each cash flow in the Flows block (see the previous section). Positive discount rates decrease the cash flow; negative ones increase it. The next figure shows a stream of cash flows (in A2..B4) and their corresponding discount rates (in C2..C4).

	A	B	C
1	Cash Flows		Discounts
2	2	$50	5.0%
3	1	($25)	-2.5%
4	1	$75	7.5%

The first two cash flows (specified by A2..B2) are discounted by 5% (as specified by C2). The third is discounted by -2.5% (an increase, as specified by the negative percentage in C3), and the final cash flow is discounted by 7.5% (as specified by C4).

You can use the Simp argument to specify how Quattro Pro applies discount rates to cash flows. The next table shows the discounting methods available.

Simp	Discounting
0	Compounded
1	Mixture of compounded and simple
2	Simple

In addition to Simp, PathDep, which is used only when Discrate is a block, specifies whether path-dependent discounting is used. When path-dependent discounting is used (PathDep is set to 1), the set of discount rates are chained together to determine future or net present value. If the order of discount rates changes, the future or net present value can be affected.

When path-dependent discounting isn't used (PathDep is set to 0, the default), each cash flow is affected by its associated discount rate; other discount rates in Discrate don't affect it.

By default, cash flow functions assume that each cash flow occurs periodically (every month, every year, and so on). The arguments Odd and Periods let you specify irregular periods. You normally use one or the other, so these arguments appear in the function descriptions as Odd|Periods.

If the length of time of the first period is odd, specify a number for Odd. For example, if a series of cash flows are monthly, and the first period is half a month long, set Odd to 0.5; if the first period is one and a half months, set Odd to 1.5.

Note: In @FUTV, Odd specifies the length of the last period.

If several cash flows are unevenly spaced, specify a block for Periods. Periods is a column (or row) of numbers that specify the duration of each cash flow in the Flows block. Like Odd, each value in the block is expressed as a fraction of the regular period. For example, the next figure shows a cash flow block in B2..B4, and a Period block in A2..A4.

	A	B
1	Periods	Flows
2	1	$50
3	3.5	$75
4	2	($25)

The value 1 in A2 specifies that the first cash flow ($50) occurs at a regular period. You decide what this period is; it could be a week, a month, or a year. Assuming the regular period is a month, the value 3.5 (in A3) specifies that the second cash flow ($75) occurs three and a half months after the first. The final value, 2, specifies that two months elapse between the second cash flow and the third.

Like Flows, the Periods block can have an additional column (or row) added to specify how many times a given period length repeats. Periods doesn't have to be the same size as Flows. For example, in the next figure, the cash flow stream is A2..B5.

	A	B	C	D
1	Cash flows		Periods	
2	4	$5	1	0.56745
3	4	$10	11	1
4	7	$11	4	1.5
5	1	$110		

Periods is C2..C4, and specifies that the first cash flow is 0.56745 periods away, the next 11 cash flows occur one period apart, and the last four cash flows are 1.5 periods apart.

See Also

Cash Flow @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Bond @Function Arguments

Argument	Description
Settle	Settlement date for the trade
Maturity	Redemption date for the bond
Coupon	Annual coupon rate expressed as a decimal
Issue	Issue date, that is, the date at which the bond is first offered for sale and begins accruing interest
FirstCpn	Date on which the first coupon period ends; if the first coupon period is longer or shorter than the other periods, the first coupon payment date is explicitly specified at issue; the size of the first coupon payment is linearly prorated in accordance with the length of the first coupon period
Redemption	Redemption value per par of 100
Freq	Frequency of coupon payments in the number of payments per year; the default is 2 (semiannual)
Calendar	Calendar to observe; the default is 0 (30/360)
Yield	Internal rate of return expressed as a decimal
Price	Quoted price of the bond assuming a par value of 100 and not including accrued interest
LastCpn	Date on which the last coupon period ends

See Also

New Financial @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

New Mathematical @Functions

@CEILING	Rounds a number up to the nearest integer.
@DFRAC	Converts a decimal number to a whole number and fractional component.
@EVEN	Rounds a number up to the nearest even integer.
@FACT	Calculates the factorial of a number.
@FACTDOUBLE	Returns the double factorial of a number.
@FIB	Calculates the nth term of a Fibonacci sequence.
@FLOOR	Rounds a number down, toward zero.
@FRACD	Converts a number with a fractional component to a decimal.
@GCD	Calculates the greatest comman divisor of x and y.
@LCM	Calculates the least common multiple of x and y.
@LINTERP	Performs linear interpolation between sets of xy pairs.
@MDET	Calculates the determinant of a matrix.
@MROUND	Returns a number rounded to the desired multiple.
@MULT	Calculates cumulative product of a set of numbers.
@MULTINOMIAL	Returns the multinomial of a set of numbers.
@ODD	Rounds a number up to the nearest odd integer.
@QUOTIENT	Returns the integer portion of a division.
@RANDBETWEEN	Returns a random number between the numbers you specify.
@SERIESSUM	Returns the sum of a power series based on a formula.
@SQRTPI	Returns the square root of a number multiplied by pi.

See Also

New @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New Miscellaneous @Functions

@ARRAY	Returns the result of an expression (a formula or @function) using array syntax.
@BLOCKNAME	Returns the name of a specified block.
@BLOCKNAME2	Returns the block name created in a given notebook for a specified block.
@BLOCKNAMES	Returns a two-column table showing the block names that intersect with a specified block.
@BLOCKNAMES2	Returns a two-column table showing the block names created in a given notebook that intersect with a specified block.
@FIRSTBLANKPAGE	Returns the page letters for the first unnamed blank page in a notebook that isn't part of a group.
@FIRSTINGROUP	Returns the page letters for the first page in the specified group.
@GETGROUP	Returns the name of the group that contains the page for a specified block, or the name of the group in the specified block that contains a specified page name.
@INDEXTOLETTER	Returns a one- or two-character string for the index number of a page or column.
@ISLEGALPAGENAME	Returns 1 if the specified page name is valid (whether it exists or not); otherwise, returns 0.
@LASTBLANKPAGE	Returns the page letters for the last unnamed blank page in a notebook that isn't part of a group.
@LASTINGROUP	Returns the page letters for the last page in the specified group.
@LETTERTOINDEX	Returns the index number for column letters or page letters.
@PAGEINDEX	Returns the index number for a notebook page with a specified name.
@PAGEINDEX2	Returns the index number for a notebook page with a specified name in a given notebook.
@PAGENAME	Returns the name of a notebook page with a given index number.
@PAGENAME2	Returns the name of a notebook page with a given index number in a specified notebook.
@PAGENAMES	Returns a two-column table showing the page letters and corresponding page names for the active notebook.
@PAGENAMES2	Returns a two-column table showing the page letters and corresponding page names for a specified notebook.

See Also

New @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New Statistical @Functions

The statistical @functions perform aggregation, counting, and analysis operations on a group of values expressed as a list (or lists) of one or more arguments. These arguments can be numeric values or block values.

Descriptive	These @functions return a value that helps you summarize and describe a group of values.
Inferential	These @functions return a value (or values) that helps you draw conclusions about a group (or groups) of values.

See Also

New @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New Descriptive Statistical @Functions

@GEOMEAN	Returns the geometric mean of all numeric values in a list.
@HARMEAN	Returns the harmonic mean of all numeric values in a list.
@KURT	Returns the kurtosis (peakedness or flatness) of a data set.
@LARGEST	Returns the k-th largest value in a data set.
@MEDIAN	Returns the median of a data set.
@MODE	Returns the most common value in a data set.
@PERCENTILE	Returns the value from a group of values at a specified percentile.
@PERCENTRANK	Returns the percentage rank of a value in a data set.
@QUARTILE	Returns the quartile of a data set.
@RANK	Returns the rank of a number in a list of numbers.
@SKEW	Returns the skewness of a distribution.
@SMALLEST	Returns the k-th smallest value in a data set.
@STANDARDIZE	Returns a normalized value.
@TRIMMEAN	Returns the mean of all numeric values in a list with a fraction of values excluded.

See Also

New @Functions
New Statistical @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

New Inferential Statistical @Functions

@AVEDEV	Performs the average of the absolute deviations of data points from their means.
@BETA	Returns the beta function.
@BETADIST	Returns the cumulative beta probability density function.
@BETAI	Returns the incomplete beta function.
@BETAINV	Returns the inverse of the cumulative beta probability density function.
@BINOMDIST	Returns the binomial probability mass function.
@CHIDIST	Returns the cumulative chi-square distribution.
@CHIINV	Returns the inverse of the cumulative chi-square distribution.
@CHITEST	Computes the probability that the actual and expected frequencies are similar by chance using the chi-square test.
@COMB	Calculates the number of unordered subgroups of given size in a group.
@CONFIDENCE	Returns the confidence interval for a population mean.
@CORREL	Returns the correlation coefficient of two data sets.
@COVAR	Returns the covariance of two data sets.
@CRITBINOM	Returns the smallest value for which the cumulative binomial distribution is less than or equal to a criterion value.
@DEVSQ	Returns the sum of the squares of the deviations.
@EXPONDIST	Returns the exponential distribution.
@FDIST	Returns the F distribution function.
@FINV	Returns the inverse of the cumulative F distribution function.
@FISHER	Returns the Fisher transformation.
@FISHERINV	Returns the inverse of the Fisher transformation.
@FORECAST	Returns a value along a linear trend.
@FTEST	Returns the result of the F-test.
@GAMMADIST	Returns the gamma distribution function.
@GAMMAINV	Computes the inverse of the cumulative Gamma distribution function.
@GAMMALN	Returns the natural logarithm of the gamma function.
@GAMMAP	Returns the incomplete gamma function.
@GAMMAQ	Returns the complement of the incomplete gamma function.
@HYPGEOMDIST	Returns the hypergeometric distribution.
@INTERCEPT	Returns the intercept of the linear regression line.
@LOGINV	Returns the inverse of the lognormal distribution.
@LOGNORMDIST	Returns the lognormal distribution.
@NEGBINOMDIST	Returns the negative binomial distribution.
@NORMDIST	Returns the normal cumulative distribution.
@NORMINV	Computes the inverse of the cumulative normal distribution function.

@NORMSDIST	Computes the standard normal cumulative distribution.
@NORMSINV	Returns the inverse of the standard normal cumulative distribution.
@PEARSON	Returns the Pearson product moment correlation coefficient.
@PERMUT	Calculates the number of ordered subgroups of given size in a group (permutations).
@POISSON	Returns the Poisson probability distribution.
@PROB	Returns the probability that values in a range are between two limits.
@RSQ	Returns the square of the coefficient of correlation of the linear regression line through data points in known xs and known ys.
@SLOPE	Returns the slope of the linear regression line.
@STEC	Returns the standard error of the regression coefficient.
@STEYX	Standard error of the predicted y-value for each x.
@SUMSQ	Returns the sum of the squares of the arguments.
@SUMX2MY2	Returns the sum of the differences of the squares of the corresponding values in two arrays.
@SUMX2PY2	Returns the sum of the sum of the squares of corresponding values in two arrays.
@SUMXMY2	Returns the sum of squares of differences of corresponding values in two arrays.
@SUMXPY2	Returns the sum of the squares of corresponding values in two arrays.
@SUMXY	Sum of the products of the corresponding numbers in two arrays.
@SUMXY2	Sum of the product of values and the squares of the corresponding numbers in two arrays.
@TDIST	Returns the Student's t-distribution.
@TINV	Returns the inverse of the Student's t-distribution.
@TTEST	Returns the probability associated with the Student's t-test.
@WEIBULL	Returns the Weibull distribution.
@ZTEST	Returns the two-tailed probability value of a z-test.

See Also

New @Functions

New Statistical @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

New @Functions Index

This section lists the new @functions in alphabetical order. Each entry includes syntax, arguments, and a short description. For the complete @functions index, see @Function Index

@ABDAYS
@ACCRINT
@ACCRINTM
@ACDAYS
@ADDB
@ADDBO
@ADDH
@ADDHO
@AMAINT
@AMINT
@AMNTHS
@AMPMT
@AMPMTI
@AMPRN
@AMRES
@AMRPRN
@AMTERM
@ANDB
@ANDH
@ARRAY
@ASCTOHEX
@AVEDEV
@BASE
@BDAYS
@BESSELI
@BESSELJ
@BESSELK
@BESSELY
@BETA
@BETADIST
@BETAI
@BETAINV
@BINOMDIST
@BINTOHEX
@BINTOHEX64

@BINTONUM
@BINTONUM64
@BINTOOCT
@BINTOOCT64
@BITRB
@BITRH
@BITSB
@BITSH
@BITTB
@BITTH
@BLOCKNAME
@BLOCKNAME2
@BLOCKNAMES
@BLOCKNAMES2
@BUSDAY
@CATB
@CATH
@CATNB
@CATNH
@CDAYS
@CEILING
@CHIDIST
@CHIINV
@CHITEST
@COMB
@COMPLEX
@CONFIDENCE
@CONVERT
@CORREL
@COUPDAYBS
@COUPDAYS
@COUPDAYSNC
@COUPNCD
@COUPNUM
@COUPPCD
@COVAR
@CRITBINOM
@DELTA
@DEVSQ
@DFRAC

@DISC
@DURAT
@DURATION
@EMNTH
@ERF
@ERFC
@EVEN
@EXPONDIST
@FACT
@FACTDOUBLE
@FBDAY
@FDIST
@FEETBL
@FIB
@FINV
@FIRSTBLANKPAGE
@FIRSTINGROUP
@FISHER
@FISHERINV
@FLOOR
@FORECAST
@FRACD
@FTEST
@FUTV
@GAMMADIST
@GAMMAINV
@GAMMALN
@GAMMAP
@GAMMAQ
@GCD
@GEOMEAN
@GESTEP
@GETGROUP
@HARMEAN
@HEXTOASC
@HEXTOBIN
@HEXTOBIN64
@HEXTONUM64
@HEXTOOCT
@HEXTOOCT64

@HOLS
@HYPGEOMDIST
@IMABS
@IMAGINARY
@IMARGUMENT
@IMCONJUGATE
@IMCOS
@IMDIV
@IMEXP
@IMLN
@IMLOG10
@IMLOG2
@IMPOWER
@IMPRODUCT
@IMREAL
@IMSIN
@IMSQRT
@IMSUB
@IMSUM
@INDEXTOLETTER
@INTERCEPT
@INTRATE
@INVB
@INVH
@ISBDAY
@ISLEGALPAGENAME
@KURT
@LARGEST
@LASTBLANKPAGE
@LASTINGROUP
@LBDAY
@LCM
@LETTERTOINDEX
@LINTERP
@LOGINV
@LOGNORMDIST
@LWKDAY
@MDAYS
@MDET
@MDURATION

@MEDIAN
@MNTHS
@MODE
@MROUND
@MTGACC
@MULT
@MULTINOMIAL
@NBDAY
@NEGBINOMDIST
@NETPV
@NORMDIST
@NORMINV
@NORMSDIST
@NORMSINV
@NUMTOBIN
@NUMTOBIN64
@NUMTOHEX64
@NUMTOOCT
@NUMTOOCT64
@NWKDAY
@OCTTOBIN
@OCTTOHEX
@OCTTONUM
@ODD
@ODDFPRICE
@ODDFYIELD
@ODDLPRICE
@ODDLYIELD
@ORB
@ORH
@PAGEINDEX
@PAGEINDEX2
@PAGENAME
@PAGENAME2
@PAGENAMES
@PAGENAMES2
@PBDAY
@PEARSON
@PERCENTILE
@PERCENTRANK

@PERMUT
@PIRATE
@POISSON
@PRICE
@PRICEDISC
@PRICEMAT
@PROB
@QUARTILE
@QUOTIENT
@RANDBETWEEN
@RANK
@RECEIVED
@RSQ
@SCMARG
@SERIESSUM
@SHLB
@SHLBO
@SHLH
@SHLHO
@SHRB
@SHRBO
@SHRH
@SHRHO
@SKEW
@SLOPE
@SMALLEST
@SQRTPI
@STANDARDIZE
@STEC
@STEYX
@STKOPT
@SUBB
@SUBBO
@SUBH
@SUBHO
@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXPY2

@SUMXY
@SUMXY2
@TBILLEQ
@TBILLPRICE
@TBILLYIELD
@TDIST
@TINV
@TRIMMEAN
@TTEST
@WEIBULL
@WKDAY
@XORB
@XORH
@YDAYS
@YDIV
@YEARFRAC
@YIELD
@YIELDDISC
@YIELDMAT
@YLD2YLD
@ZTEST

@ABDAYS

Format

```
@ABDAYS(Date, Days, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing the date to which a number of business days should be added
Days	=	integer representing number of business days to add; can be negative
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@ABDAYS adds or subtracts Days business days from Date and returns a serial date number.

If Date falls on a weekend or a holiday, one business day out of Days is used to bring Date forward to the next business day; if Days is negative, Date is taken backward to the previous business day. For example, if 20 business days are added to June 5, 1993, the result is the same as adding 19 business days to June 7, 1993 since June 5 falls on a Saturday.

Example

This formula calculates the date that precedes January 12, 1994 by 90 business, assuming that Saturday, Sunday, and the dates in the block A7..C9 are holidays:

`@ABDAYS(@DATE(94,1,12),-90,A7..C9)` = 34213 (September 1, 1993)

See Also

Setting Holidays

@ACCRINT

Format

```
@ACCRINT(Settle, Maturity, Coupon, <Issue>, <FirstCpn>, <Par>, <Freq>,
  <Calendar>)
```

Settle	=	number representing the settlement date
Maturity	=	number representing the maturity date
Coupon	=	coupon rate; 0 ≤ Coupon ≤ 1
Issue	=	number representing the issue date
FirstCpn	=	number representing the first coupon date
Par	=	par value (the default is 1000)
Freq	=	frequency of coupon payments in the number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ACCRINT returns the accrued interest for a bond. Accrued interest represents an amount paid to the bond seller as compensation for owning the bond for a fraction of a coupon period. Interest accrues from the last coupon date to the settlement date. @ACCRINT returns the accrued interest per 1000 face value.

Dates for @ACCRINT must follow this pattern:

Issue < Settle < FirstCpn < Maturity

Example

This formula returns the accrued interest, as of May 15, 1993, on an 8.875% bond with a $100,000 face value, maturing February 15, 1998, dated November 22, 1992, and paying its first coupon on August 15, 1993:

```
@ACCRINT(@DATE(93,5,15),@DATE(98,2,15),0.08875,@DATE(92,11,22),@DATE(93,8,1
5), 100000)
```
= $4,264.93

@ACCRINTM

Format

```
@ACCRINTM(Issue, Settle, Coupon, <Par>, <Calendar>)
```

Issue	=	number representing the issue date; must be < Settle
Settle	=	number representing the settlement date
Coupon	=	coupon rate; 0 ≤ Coupon ≤ 1
Par	=	par value (the default is 1000)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ACCRINTM calculates the amount of interest accrued per par value between the issue date of the coupon and the settle date.

Example

This formula returns the accrued interest on a certificate of deposit (CD) with $1,000,000 face value settling March 11, 1990, dated December 15, 1989, and paying a coupon of 10% on an actual/360 basis:

```
@ACCRINTM(@DATE(89,12,15),@DATE(90,3,11),0.10,1000000,2)
```

= $23,888.89

@ACDAYS

Format

```
@ACDAYS(Date, Days, <Calendar>, <EndMnth>)
```

Date	=	number representing the date to add days to
Days	=	integer representing number of days to add; can be negative
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)
EndMnth	=	1 to indicate adherence to ends of months; 0 to indicate that ends of months are ignored (the default is 1)

@ACDAYS adds Days days to Date using an actual or 30/360 calendar and returns a serial date number. If Days is negative, @ACDAYS subtracts the absolute value of Days from Date.

With the 30/360 calendar, if the ending month is February and the remaining days push the result to the 29th or 30th, the result is forced to the true end of the month (28th or 29th, depending on whether Date is in a leap year).

More than one result is possible when using the 30/360 calendar. For example, adding 90 days to April 30 can yield either July 30 or July 31.

Examples

`@ACDAYS(@DATE(93,1,1),120)` = 34090 (May 1, 1993)

`@ACDAYS(@DATE(93,11,15),-270)` = 34015 (February 15, 1993)

See Also

@ABDAYS

@AMNTHS

@YDIV

@ADDB

Format

```
@ADDB(Binary1, <Binary2>, <BitIn>, <Bits>)
```

Binary1	=	first binary number
Binary2	=	second binary number
BitIn	=	input carry bit; can be either 0 (the default) or 1
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be $\leq$ 64

@ADDB returns the sum of two binary numbers. If Binary2 is omitted, @ADDB counts the bits in Binary1 that are set to 1; this bit counting operation is called addition reduction.

Use two's complement notation to represent negative numbers. If BitIn is 1, @ADDB adds one extra bit to the result.

Example

`@ADDB(100,100)` = 1000

`@ADDB(100,100,1,4)` = 1001

`@ADDB(101)` = 2

`@ADDB(1100,1,1,5)` = 01110

See Also

Entering Boolean @Functions

@ADDBO

Format

```
@ADDBO(Binary1, Binary2, <BitIn>, <Bits>)
```

Binary1	=	first binary number
Binary2	=	second binary number
BitIn	=	input carry bit; can be either 0 (the default) or 1
Bits	=	number of binary bits used for input; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be $\leq$ 64

@ADDBO returns the overflow bit (either 0 or 1) of the sum of two binary numbers. An overflow occurs when a bit is carried out of the word size specified by Bits. For example, if Binary1 = 10 and Binary2 = 10, the sum of the two numbers is 00, with 1 carry bit in the third place not shown.

Use two's complement notation to represent negative numbers. If BitIn is 1, @ADDBO adds one extra bit to the sum of the two numbers before returning the overflow.

Example

`@ADDBO(1000,111)` = 0

`@ADDBO(1000,111,1)` = 1

`@ADDBO(1100,100,1,4)` = 1

See Also

Entering Boolean @Functions

@ADDH

Format

```
@ADDH(Hex1, <Hex2>, <BitIn>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
BitIn	=	input carry bit; can be either 0 (the default) or 1
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; must be ≤ 64

@ADDH returns the sum of two hexadecimal numbers. If Hex2 is omitted, @ADDH counts the bits in Hex1 that are set to 1; this bit counting operation is called addition reduction.

Use two's complement notation to represent negative numbers. If BitIn is 1, @ADDH adds one extra bit to the result.

Example

`@ADDH("E00","100")` = F00

`@ADDH("100","100",1,16)` = 0201

`@ADDH("9")` = 2

`@ADDH("C","1",1,8)` = 0E

See Also

Entering Boolean @Functions

@ADDHO

Format

```
@ADDHO(Hex1, Hex2, <BitIn>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
BitIn	=	input carry bit; can be either 0 (the default) or 1
Bits	=	number of binary bits used for input; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@ADDHO returns the overflow bit (either 0 or 1) of the sum of two hexadecimal numbers. An overflow occurs when a bit is carried out of the word size specified by Bits. For example, if the binary equivalents for Hex1 and Hex2 are 10 and 10, the sum of the two numbers is 00 with 1 carry bit in the third place not shown.

Use two's complement notation to represent negative numbers. If BitIn is 1, @ADDHO adds one extra bit to the sum of the two numbers before returning the overflow.

Examples

`@ADDHO("8","F")` = 1

`@ADDHO("8","F",1,5)` = 0

`@ADDHO("C","4",1,4)` = 1

See Also

Entering Boolean @Functions

@AMAINT

Format

```
@AMAINT(Principal, Int, Term, n, <Part>, <Residual>, <ResOff>, <Adv>,
  <Odd>, <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate
Term	=	term of the loan, expressed in number of total payments
n	=	number of payments made; must be an integer from 0 to Term
Part	=	part of (n+1)th period passed (must be from 0 to 1; the default is 0)
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0); n ± Adv must be an integer
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMAINT calculates the accumulated interest paid on a loan after n payments. The accumulated interest is the sum of the interest portions of the first n payments plus the interest of an optional partial payment (specified by Part).

Term and n should include advance payments made at the beginning of the loan. Part handles payoff situations where the payoff date doesn't coincide with a periodic payment date. For example, if 20 out of 60 monthly payments of a loan have been made and 10 days have passed since the 20th payment date, Part = ⁄, assuming there are 31 days between the 20th and 21st monthly payments.

If Simp = 0, @AMAINT uses this formula:

$$I = n * Pa - Pr + B * (1 + Int)^{P}$$

If Simp = 1, @AMAINT uses this formula:

$$I = n * Pa - Pr + B * (1 + Int * P)$$

where Payment is the periodic payment and Balance is the remaining balance on the loan after n payments. Balance, like Principal, is the present value of an annuity paying Payment.

I = interest

Pa = payment

Pr = principal

B = balance

P = part

Examples

A loan of $10,000 was made on September 11, 1992 to be repaid in 48 monthly installments. The annual interest rate is 8.4% (8.4%/12 = 0.7% monthly rate). The first payment was paid in advance. This formula calculates the amount of paid interest after 15 regular payments:

`@AMAINT(10000,0.007,48,15,0,0,0,1)` = $840.74

For the same loan, this formula calculates how much interest accumulated on the loan as of March 3, 1993, assuming the borrower made regular payments. The previous payment (the 18th) fell on February 11, 1993; there are 21 days between the previous payment and March 3. Interest accrues as simple interest over fractional periods.

`@AMAINT(10000,0.007,48,18,21/29,0,0,1,1,1)` = $1,020.69

@AMINT

Format

```
@AMINT(Principal, Term, Payment, <Residual>, <ResOff>, <Adv>, <Odd>,
  <Simp>, <Prec>)
```

Principal	=	initial loan principal
Term	=	term of loan, expressed in number of total payments
Payment	=	periodic payment (for example, if Term is expressed in months, Payment must be a monthly payment)
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that Residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)
Prec	=	required precision of result (the default is 0.000001); must be ≥ 0

@AMINT calculates the interest rate for one payment of an amortized loan. For example, if the arguments passed correspond to a monthly loan, the interest rate returned represents a monthly rate. Use Prec to specify how close @AMINT must be to the actual interest rate.

Example

A loan for $50,000 was made on March 15, 1993. Repayment terms stipulate ten annual payments of $7,500, each to be made on July 31, beginning 1993 and ending 2002, along with a final payment of $2,500 on December 31, 2003. Assuming interest is compounded during fractional periods, this formula calculates the interest rate at which the financing is performed:

`@AMINT(50000,10,7500,2500,1.4180,0,0.3781)` = 0.098915

A normal period is one year long, but the first period is 138 days long. Since there are 365 days between March 15, 1993 and March 15, 1994, the first period is 138/365 the length of a normal period (Odd = 0.3781). There is also a delay between the last periodic payment and the residual payment of $2,500. The length of the delay is one period (July 31, 2002 to July 31, 2003) plus 153 days (July 31, 2003 to December 31, 2003), so ResOff = 1 153/365 (1.4180).

@AMNTHS

Format

```
@AMNTHS(Date, Months, <EndMnth>)
```

Date	=	number representing the date to add number of months to
Months	=	integer representing number of months to add; can be negative
EndMnth	=	1 to indicate adherence to ends of months; 0 to indicate that ends of months are ignored (the default is 1)

@AMNTHS adds the number of months specified by Months to Date and returns a serial date number. If Months is negative, @AMNTHS subtracts the absolute value of Months from Date.

Adding one month usually means going from a day in one month to the same day in the next month. However, adding one month to March 31 can't result in April 31, since April 31 doesn't exist. In this case, the last day of the month, April 30, is returned. If Date falls on the 31st of a month, the result also falls on the last day of a month.

If Date falls on the last day of a month with less than 31 days, you can use EndMnth to specify one of two different results: to move months ahead to the same day, specify EndMnth as 0; to move months ahead to the last day of the month, omit EndMnth or specify it as 1.

Examples

`@AMNTHS(@DATE(93,4,30),3)` = 34181 (July 31, 1993), which is the last day of the month three months from April 30, 1993.

Consider a loan with 120 payments that pays on the 30th of each month starting on June 30, 1993. In February, it pays on the last day of the month. This formula calculates the date of the 43rd payment:

`@AMNTHS(@DATE(93,6,30),42,0)` = 35429 (December 30, 1996)

See Also

@ABDAYS

@ACDAYS

@YDIVS

@AMPMT

Format

```
@AMPMT(Principal, Int, Term, <Residual>, <ResOff>, <Adv>, <Odd>, <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate (for example, if Term is expressed in months, Int must be a monthly rate)
Term	=	term of loan, expressed in number of total payments
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMPMT calculates the payment (monthly, annual, and so on) for an amortized loan.

Examples

A loan for $35,000 has 48 monthly payments and a residual payment of $7,500 that is due three months after the last monthly payment. This formula calculates the monthly payment if the annual interest rate is 9% (9%/12 = 0.75% monthly rate):

`@AMPMT(35000,0.0075,48,7500,3)` = $743.48

An annuity with an investment of $250,000 makes quarterly payments starting five years from the date of investment for a period of 20 years. It also pays a lump sum of $50,000 three and a half years after the last quarterly payment. If the annualized yield from the investment is 8.4% (8.4%/4 = 2.1% quarterly rate), this formula calculates the quarterly payment:

`@AMPMT(250000,0.021,80,50000,14,0,20)` = $9,431.83

The term of the annuity is 80 quarters. A value of 14 for ResOff specifies, in quarters, the three-and-a-half year delay between the last quarterly payment and the residual payment. A value of 20 for Odd specifies the five year delay between investment and first payment, in quarters.

@AMPMTI

Format

```
@AMPMTI(Principal, Int, Term, n, <Residual>, <ResOff>, <Adv>,  <Odd>,
  <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate (for example, if Term is expressed in months, then Int must be a monthly rate)
Term	=	term of loan, expressed in number of total payments
n	=	number of payments made; must be an integer from 0 to Term
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMPTI calculates the interest portion of the nth payment of an amortized loan. Term and n should include any advance payments made at the beginning of the loan.

If Simp = 0, @AMPMTI uses this formula:

$$I_n = B_{n-1} * [(1 + Int)^{t_n} - 1]$$

If Simp = 1, @AMPMTI uses this formula:

$$I_n = B_{n-1} * [((1 + Int)^{I(t_n)} * (1 + F(t_n) * Int)) - 1]$$

```
I = interest
B = balance
```

Balance , like Principal, is the present value of an annuity. Both correspond to annuities with the same payment size but differing in number of payments. If Principal corresponds to an annuity with Term payments, Balance

corresponds to annuity with Term n + 1 payments.

equals 1 except when n equals Adv + 1, in which case

equals Odd.

Example

This formula calculates the portion of the 15th payment (the 15th after any advanced payments) of a 120 payment loan that constitutes interest, if the original principal of $100,000 is financed at a periodic rate of 4.5% and the first two payments are made in advance:

`@AMPMTI(100000,0.045,120,17,0,0,2)` = $4,106.92.

The value of 17 passed for n specifies the 15th payment after the two advance payments.

@AMPRN

Format

```
@AMPRN(Int, Term, Payment, <Residual>, <ResOff>, <Adv>, <Odd>, <Simp>)
```

Int	=	periodic interest rate (for example, if Term is expressed in half-years, Int must be a semiannual rate)
Term	=	term of loan, expressed in number of total payments
Payment	=	periodic payment
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMPRN calculates the initial principal of an amortized loan.

Examples

A loan has an annualized monthly compounded interest rate of 10.8% (10.8%/12 = 0.9% monthly rate) over a period of 48 months, and the monthly payment is $525. Each payment is due at the beginning of the month, including the first payment which coincides with the loan's start date. This formula calculates the amount financed:

`@AMPRN(0.009,48,525,0,0,1)` = $20,573.04

A savings plan requires a monthly contribution of $1,000 for a period of 25 years. If the plan pays an annual interest rate of 6.6% (6.6%/12 = 0.55% monthly rate), this formula calculates what initial deposit (not payment), if any, is needed in order to accumulate $1,000,000 two and a half years after the last monthly payment:

`@AMPRN(0.0055,300,-1000,1000000,30)` = $16,907.93.

If the annuity is viewed as a loan and the investor as the lender, the original investment can be treated as the loan principal, the monthly payments as payments from lender to borrower, and the $1,000,000 future value as a residual payment from the borrower to the lender. The interest rate is 0.55%. The negative payment means payment from lender to borrower. A value of 30 for ResOff specifies the thirty month delay between the last monthly contribution and the date on which to measure the end balance.

@AMRES

Format

```
@AMRES(Principal, Int, Term, Payment, <ResOff>, <Adv>, <Odd>, <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate (for example, if Term is expressed in months, then Int must be a monthly rate)
Term	=	term of loan, expressed in number of total payments
Payment	=	periodic payment
ResOff	=	number of periods after last periodic payment that Residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMRES calculates the residual (or balloon) payment of an amortized loan or the future value of an annuity.

Example

A $10,000,000 loan is paid back in 20 payments of $1,000,000 and a final payment. The first payment is made at the start of the loan. The remaining 19 payments are made annually, starting 9 months after the first payment. The final payment is made 20 months after the last annual payment. If the loan has an annual interest rate of 9.68%, this formula calculates the final payment (assume compounding of interest over fractional periods):

`@AMRES(10000000,0.0968,20,1000000,20/12,1,0.75)` = $1,681,942.54.

ResOff is set to 20/12 to specify the 20 month delay between the last annual payment and the residual payment. Adv is set to 1 to specify the advance payment. Odd is set to 0.75 (0.75) to specify the nine month period between the start of the loan and the second payment.

@AMRPRN

Format

```
@AMRPRN(Principal, Int, Term, n, <Part>, <Residual>, <ResOff>, <Adv>,
  <Odd>, <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate (for example, if Term is expressed in years, then Int must be a yearly rate)
Term	=	term of loan, expressed in number of total payments
n	=	number of payments made; must be an integer from 0 to Term
Part	=	part of (n+1)th period passed; 0 ≤ Part ≤ 1 (the default is 0)
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMRPRN computes the balance remaining after n payments, accounting for possible interest accrual over part of the following payment period.

Term and n should include advance payments made at the beginning of the loan. Part handles payoff situations where the payoff date (date on which the remaining balance on a loan is fully paid) doesn't coincide with a periodic payment date. For example, if 15 out of 36 monthly payments of a loan have been made and 17 days have passed since the 15th payment date, Part = ⁄ , assuming there are 30 days between the 15th and 16th monthly payments.

If Simp = 0, @AMRPRN uses this formula:

$$B_{n+P} = B_n * (1 + Int)^P$$

If Simp = 1, @AMRPRN uses this formula:

$$B_{n+P} = B_n * (1 + Int * P)$$

```
B = Balance
P = Part
```

Examples

A loan of $100,000 has an annual interest rate of 9.6% (9.6%/12 = 0.8% monthly rate). Repayment consists of monthly payments over ten years. This formula calculates the balance remaining after the 57th monthly payment:

`@AMRPRN(100000,0.008,120,57)` = $64,108.38

A loan of $2,000,000 was made on March 16, 1993, to be paid back in 40 quarterly payments and a final payment of $100,000. The annual interest rate is 9.96% (9.96%/4 = 2.49% quarterly rate). The first four payments are due at the start of the loan. The fifth payment is due July 1, 1993. Thereafter, a payment is due every three months. The final residual payment of $100,000 is due June 15, 2002. This formula calculates the remaining balance due on the loan as of September 23, 1996, assuming timely payments and simple interest accrual over fractional periods:

`@AMRPRN(2000000,0.0249,40,17,84/92,100000,75/91,4,1+16/90,1)` = $1,321,951.76.

September 23, 1994 falls in the middle of the payment period following the 17th quarterly payment. It falls 84 days into the quarter, which is 92 days long, so Part = / . The residual is paid 75 days after the 40th quarterly payment. The corresponding quarter (April 1, 2002 to July 1, 2002) contains 91 days, so ResOff =

/ . The first quarterly payment after the advance payments is paid one period and 16 days after loan inception. The 16 days correspond to the quarter containing the loan inception date, March 16, 1993. That quarter contains 90 days, so Odd = 1

/ .

@AMTERM

Format

```
@AMTERM(Principal, Int, Payment, <Residual>, <ResOff>, <Adv> <Odd>, <Simp>)
```

Principal	=	initial loan principal
Int	=	periodic interest rate (for example, if term is expressed in quarters, Int must be a quarterly rate)
Payment	=	periodic payment
Residual	=	remaining balance on loan at end of loan term (the default is 0)
ResOff	=	number of periods after last periodic payment that residual is to be paid; can have fractional component (the default is 0)
Adv	=	number of advance payments made at loan inception (the default is 0)
Odd	=	number of periods between loan inception and date of first payment (not including advance payments); can have fractional component (the default is 1)
Simp	=	0 to specify compounded interest or 1 to specify simple interest (the default is 0)

@AMTERM calculates the duration of an amortized loan, expressed in number of payments.

Example

A loan for $50,000 was made on April 1, 1993. The loan is repaid monthly, starting on May 1, 1993, except for the first three payments, which are due at the start of the loan (April 1, 1993). If the monthly interest rate is 1.15%, this formula calculates the smallest number of payments that would allow repayment with a maximum allowable monthly payment of $600:

```
@AMTERM(50000,0.0115,600,0,0,3)
```
= 228.18

The smallest number of payments to support such a loan is 229, which results in a monthly payment of $599.56. Using 228 payments results in a monthly payment of $600.10.

@ANDB

Format

```
@ANDB(Binary1, <Binary2>, <Bits>)
```

Binary1 = first binary number

Binary2 = second binary number

Bits = number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be ≤ 64

@ANDB performs a bit-by-bit logical AND of each bit in Binary1 and Binary2. Use @ANDB to set bits to 0; any bit that is 0 in either Binary1 or Binary2 causes the resulting output bit to be 0.

If only one number is given, then @ANDB performs an all-ones test, or AND reduction, on Binary1; @ANDB returns 1 if all bits in Binary1 are set to 1; otherwise, it returns 0.

Examples

`@ANDB(10,1)` = 00

`@ANDB(11,10)` = 10

`@ANDB(11)` = 1

`@ANDB(1100,111,5)` = 00100

See Also

Entering Boolean @Functions

@ANDH

Format

```
@ANDH(Hex1, <Hex2>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@ANDH performs a bit-by-bit logical AND of each bit in Hex1 and Hex2. Use @ANDH to set bits to 0; any binary bit that is 0 in either Hex1 or Hex2 causes the resulting output bit to be 0.

If only one number is given, then @ANDH performs an all-ones test, or AND reduction, on Hex1; @ANDH returns 1 if all bits in Hex1 are set to 1; otherwise, it returns 0.

Examples

`@ANDH("A","F")` = A

`@ANDH("A")` = 0

`@ANDH("C","4",8)` = 04

See Also

Entering Boolean @Functions

@ARRAY

Format

```
@ARRAY(Expression, <Columns>, <Rows>)
```

Expression	=	formula or @function using array syntax; @functions can be nested, that is, you can have more than one @function in a single statement
Columns	=	number of columns in the output range, including the column of the current cell (the default Columns depends on dimensions of input array(s) in Expression)
Row	=	number of rows in the output range, including the row of the current cell (the default Rows depends on dimensions of input array(s) in Expression)

@ARRAY returns the result of Expression, which can be either a formula with array operands or an @function with array arguments. An array is a block of values treated as a single group. You don't need to type @ARRAY to create an array formula or @function; if an Expression requires array output, Quattro Pro converts it by surrounding it with the @ARRAY @function.

Note: If an array expression returns an array, the @ARRAY @function appears only in the current cell, which is also the upper left cell of the output array; the other cells in the array contain calculated values. Also, array formulas don't recognize 3-D block syntax; if you specify a 3-D block in Expression, @ARRAY recognizes only the block on the first page of the series of consecutive pages.

Columns and Rows are optional arguments; the size of an output array is dependent on the size of the input array(s) in Expression. Specify values for Columns and Rows only if you want to truncate or replicate portions of the output array.

By using arrays in formulas and @functions, you can perform an operation on multiple values or cells. You also save time by not having to repeat the same formula or @function in multiple cells. Arrays also save memory by reducing the number of formulas in a notebook.

Tip: If you use many array formulas in a notebook, recalculation may become noticeably slower. Also, if you plan to share a notebook with other people, keep in mind that array formulas can make notebooks difficult to understand.

Examples

The next figure shows several examples using @ARRAY.

	A	B	C	D
1	B1=@ARRAY({1;2;3}*12)	12	24	36
2				
3	B3=@ARRAY({1|2|3}*12)	12		
4		24		
5		36		
6				
7	B7=@ARRAY(D7..D9*2)	16		8
8		20		10
9		24		12
10				
11	B11=@ARRAY(D7..D9/{4;5;6})	2	1.6	1.333333

12		2.5	2	1.666667
13		3	2.4	2
14				
15		1.23	-6.43	9
16	B16=@ARRAY(@ABS(B15..D15))	1.23	6.43	9
17				
18	B18=@ARRAY(@SQRT(C18..C20))	6.78233	46	
19		7.348469	54	
20		6	36	
21				
22	B22=@ARRAY(@UPPER(C22..C25))	THIS	This	
23		IS	is	
24		A	a	
25		TEST	Test	

See Also
Array Features

@ASCTOHEX

Format

```
@ASCTOHEX(ASCII, <Places>)
```

ASCII = ASCII character string to convert; can be up to 20 ASCII characters

Places = number of characters to return; can be from 1 to 40 characters

@ASCTOHEX returns the hexadecimal string equivalent of an ASCII number.

Note: If the ASCII value includes nonnumeric characters, enclose it in quotation marks.

Examples

`@ASCTOHEX("A")` = 41

`@ASCTOHEX("A",4)` = 0041

`@ASCTOHEX("01ABCDEF")` = 3031414243444546

`@ASCTOHEX("BORLAND",5)` = 14E44

See Also

Entering Number Conversion @Functions

@AVEDEV

Format

```
@AVEDEV(List)
```

List = one or more numeric or block values

@AVEDEV returns the mean absolute deviation, that is, the average of the absolute deviation of the data points in List from their mean. Use @AVEDEV to measure the variability of a data set around the mean. @AVEDEV uses this formula:

$$\frac{1}{N}\sum_{j=1}^{N}\left|x_j - \bar{x}\right|$$

Example

`@AVEDEV(10,11,12,13,12,11,10)` = 0.897959

See Also

@DEVSQ
@STD
@STDS

@BASE

Format

```
@BASE(Decimal, <Base>, <Precision>)
```

Decimal	=	any decimal value to convert
Base	=	indicates the target base in which to express Decimal; can be any integer from 2 to 36, inclusive (the default is 16)
Precision	=	indicates the number of desired digits after the decimal point; can be any integer from 0 to 15, inclusive (the default is 0)

@BASE converts a number from base-10 to a string value in a target base from 2 to 36.

Examples

`@BASE(128,8)` = 200

`@BASE(123.47,16,6)` = 7B.7851EB

See Also

Entering Number Conversion @Functions

@BDAYS

Format

```
@BDAYS(StartDate, EndDate, <Holidays>, <Saturday>, <Sunday>)
```

StartDate = number representing the start date

EndDate = number representing the end date

Holidays = block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)

Saturday = 0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)

Sunday = 0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@BDAYS returns the number of business days between StartDate and EndDate inclusive of EndDate. If EndDate is less than StartDate, the result is negative.

If neither StartDate nor EndDate falls on a weekend or holiday, @BDAYS returns the number of business days from StartDate to EndDate, including EndDate.

If StartDate and EndDate are two consecutive business days, the result is 1. If StartDate and EndDate both fall on weekends or holidays, @BDAYS returns the number of business days between the two dates, excluding EndDate.

If StartDate or EndDate (but not both) falls on a weekend or holiday, the result depends on which date falls on a business day. If StartDate falls on the weekend, the result includes EndDate. For example, if StartDate is a Saturday and EndDate is the following Thursday, the result includes the Thursday and is 4. If EndDate falls on the weekend, the result doesn't include EndDate. For example, if StartDate is a Thursday and EndDate is the following Saturday, the result is 1.

Examples

This formula calculates how many business days pass from November 30, 1993 to November 14, 1993, assuming that the dates in the block A7..C9 are holidays:

```
@BDAYS(@DATE(93,11,30),@DATE(93,11,14),A7..C9)
```
= -9

This formula calculates how many business days pass from June 2, 1993 to June 10, 1993, assuming no holidays other than weekends:

```
@BDAYS(@DATE(93,6,2),@DATE(93,6,10))
```
= 6

See Also

Setting Holidays

@HOLS

@CDAYS

@BESSELI

Format

```
@BESSELI(x, n)
```

x = numeric value at which to evaluate the function

n = number ≥ 0 representing the order of the Bessel function; if n isn't an integer, it's truncated to an integer

@BESSELI calculates the nth order modified Bessel function of the variable x with this formula:

$$I_n(x) = (i)^{-n} J_n(ix)$$

@BESSELI is equivalent to the Bessel function J , but is evaluated for purely imaginary arguments.

Example

```
@BESSELI(1.5,0)
```
= 1.646723

See Also

@BESSELJ

@BESSELK

@BESSELY

@BESSELJ

Format

```
@BESSELJ(x, n)
```

x = numeric value at which to evaluate the function

n = number ≥ 0 representing the order of the Bessel function; if n isn't an integer, it's truncated to an integer

@BESSELJ calculates the Bessel function J (x) using this formula:

$$J_n(X)\sum_{k=0}^{\infty}\frac{(-1)^k}{k!\,\Gamma(n+k+1)}\left(\frac{x}{2}\right)^{n+2k}$$, where

$$\Gamma(n+k+1)=\int_0^{\infty}e^{-x}x^{n+k}dx$$ is the Gamma function.

Example

`@BESSELJ(1.5,0)` = 0.511828

See Also

@BESSELI

@BESSELK

@BESSELY

@BESSELK

Format

```
@BESSELK(x, n)
```

x = numeric value at which to evaluate the function; must be > 0

n = integer ≥ 0 representing the order of the Bessel function; if n isn't an integer, it's truncated to an integer

@BESSELK calculates the nth order modified Bessel function of the variable x with this formula:

$$K_n(x) = \frac{\pi}{2} i^{n+1}\left[J_n(ix) + iY_n(ix)\right]$$

where and are @BESSELJ and @BESSELY, respectively.

Example

`@BESSELK(1.5,0)` = 0.213806

See Also

@BESSELJ

@BESSELI

@BESSELY

@BESSELY

Format

```
@BESSELY(x, n)
```

x = nonnegative numeric value at which to evaluate the function

n = integer ≥ 0 representing the order of the Bessel function; if n isn't an integer, it's truncated to an integer

@BESSELY calculates the Bessel function Y (x) (also called the Neumann or Weber function) using this formula:

$$Y_n(x) = \lim_{v \to n} \frac{J_v(x)\cos(v\pi) - J_{-v}(x)}{\sin(v\pi)}$$

where:

$$J_{-n}(x) = (-1)^n J_n(x)$$

Example

`@BESSELY(1.5,0)` = 0.382449

See Also

@BESSELJ

@BESSELI

@BESSELK

@BETA

Format

```
@BETA(Z, W)
```

Z = a parameter to the function; must be > 0

W = b parameter to the function; must be > 0

@BETA returns the value of the beta function, which is widely used in mathematics and statistics. @BETA uses this formula:

$$\beta(z,w) = \beta(w,z) = \int_0^1 t^{z-1}(1-t)^{w-1}\,dt$$

Examples

`@BETA(4,3)` = 0.016667

`@BETA(2,3)` = 0.083333

`@BETA(9,0.4)` = 0.93348

`@BETA(12,0.3)` = 1.432072

See Also

@BETADIST

@BETAI

@BETAINV

@BETADIST

Format

```
@BETADIST(X, Z, W, <A>, <B>)
```

X = value at which to evaluate the function over the interval A ≤ X ≤ B

Z = a distribution parameter; if W = 0, Z > 0

W = b distribution parameter; if Z = 0, W > 0

A = optional lower bound to the interval of X (the default is 0); A cannot equal B and must be ≤ X

B = optional upper bound to the interval of X (the default is 1); B cannot equal A and must be ≥ X

@BETADIST returns the cumulative beta probability density function. The cumulative beta probability density function is a bounded distribution that is useful for studying variables such as percentages that may only take on values within a restricted range. The optional arguments A and B set those bounds.

Examples

`@BETADIST(0.5,3,4,0,1)` = 0.65625

`@BETADIST(0.4,3,4,0,1)` = 0.45568

See Also

@BETA

@BETAI

@BETAINV

@BETAI

Format

```
@BETAI(Z, W, X)
```

Z = a parameter to the function; if W = 0, Z > 0

W = b parameter to the function; if Z = 0, W > 0

X = value at which to evaluate the function; cannot exceed 1

@BETAI computes the incomplete beta function, that is, the probability that a standard beta-distributed variable will be less than X. @BETAI uses this formula:

$$I_x(z,w) = 1 - I_{1-x}(w,z) = \int_0^x t^{z-1}(1-t)^{w-1}\,dt$$

Examples

`@BETAI(3,4,0.5)` = 0.65625

`@BETAI(3,4,0.1)` = 0.01585

`@BETAI(3,4,0.98)` = 0.999998

`@BETAI(7,8,0.7)` = 0.968531

See Also

@BETA

@BETADIST

@BETAINV

@BETAINV

Format

```
@BETAINV(Prob, Z, W, <A>, <B>)
```

Prob = cumulative probability value; $0 \leq Prob \leq 1$

Z = a parameter to the Beta distribution; must be > 0

W = b parameter to the Beta distribution; must be > 0

A = optional lower bound to the interval of X (the default is 0); A cannot equal B and must be $\leq X$

B = optional upper bound to the interval of X (the default is 1); B cannot equal A and must be $\geq X$

@BETAINV computes the inverse of the cumulative beta distribution function. If Prob = @BETADIST(X...), then @BETAINV(Prob...) = X.

Examples

`@BETAINV(0.65625,3,4,0,1)` = 0.5

`@BETAINV(0.45568,3,4,0,1)` = 0.4

See Also

@BETA

@BETADIST

@BETAI

@BINOMDIST

Format

```
@BINOMDIST(Successes, Trials, Prob, Cumulative)
```

Successes	=	number of successes in number of trial runs; must be ≥ 0
Trials	=	number of independent trial runs in sample; must be > Successes
Prob	=	probability of a success on each trial run; must be ≥ 0 and ≤ 1
Cumulative	=	1 to return the cumulative distribution function; 0 to return the probability that there are exactly Successes successes.

@BINOMDIST returns the binomial probability mass function, which is the probability that the number of successes in the independent trials equals Successes. Use @BINOMDIST when the outcome of experiments is success or failure, when the experiments are independent of one another, and when the probability of success doesn't change in successive trials. For example, a coin toss experiment is a binomial experiment.

Example

Using a random sample, a polling organization asks 50 voters if they favor Candidate A for reelection. Given that 55% of the city's voters favor Candidate A, this formula returns the probability that 40 people from the sample will favor her:

```
@BINOMDIST(40,50,.55,0)
```
= 0.000144

See Also

@CRITBINOM

@NEGBINOMDIST

@BINTOHEX

Format

```
@BINTOHEX(Binary)
```

Binary = binary number to convert; denote negative numbers using a minus sign

@BINTOHEX returns the hexadecimal string equivalent of a binary number.

Examples

```
@BINTOHEX("1010")
```
= A

```
@BINTOHEX("10000")
```
= 10

```
@BINTOHEX("11110")
```
= 1E

See Also

@BINTOHEX64

Entering Number Conversion @Functions

@BINTOHEX64

Format

```
@BINTOHEX64(Binary, <Places>)
```

Binary = binary number to convert; must be positive

Places = number of characters to return; must be ≤ 16

@BINTOHEX64 returns the hexadecimal string equivalent of a binary number (up to 64 bits).

Examples

`@BINTOHEX64(1001)` = 9

`@BINTOHEX64(1010,2)` = 0A

`@BINTOHEX64("11110000001111000")` = 1E078

`@BINTOHEX64("11110000001111000",2)` = 78

See Also

@BINTOHEX

Entering Number Conversion @Functions

@BINTONUM

Format

```
@BINTONUM(Binary)
```

Binary = binary number to convert; denote negative numbers using a minus sign

@BINTONUM returns the decimal equivalent of a binary number.

Examples

`@BINTONUM("1010")` = 10

`@BINTONUM("10000")` = 16

`@BINTONUM("11110")` = 30

See Also

@BINTONUM64

Entering Number Conversion @Functions

@BINTONUM64

Format

```
@BINTONUM64(Binary, <Signed>)
```

Binary = binary number to convert

Signed = 1 if the most significant bit of Binary is a sign bit; 0 (the default) if Binary is positive

@BINTONUM64 returns the decimal equivalent of a binary number (up to 64 bits).

If Signed is 1, the most significant bit of Binary is the sign bit. If the sign bit is 0, the number is positive; if it's 1, the number is negative.

Examples

`@BINTONUM64(100)` = 4

`@BINTONUM64(1010)` = 10

`@BINTONUM64("11110000001111000")` = 123000

`@BINTONUM64("11110000001111000",1)` = -8072

See Also

@BINTONUM

Entering Number Conversion @Functions

@BINTOOCT

Format

```
@BINTOOCT(Binary)
```

Binary = binary number to convert; denote negative numbers using a minus sign

@BINTOOCT returns the octal string equivalent of a binary number.

Examples

```
@BINTOOCT("1010")
```
= 12

```
@BINTOOCT("10000")
```
= 20

```
@BINTOOCT("11110")
```
= 36

See Also

@BINTOOCT64

Entering Number Conversion @Functions

@BINTOOCT64

Format

```
@BINTOOCT64(Binary, <Places>)
```

Binary = binary number to convert; must be positive

Places = number of characters to return; must be ≤ 22

@BINTOOCT returns the octal string equivalent of a binary number (up to 64 bits).

Examples

`@BINTOOCT64("0111")` = 07

`@BINTOOCT64("1000",3)` = 010

`@BINTOOCT64("11110000001111000")` = 360170

`@BINTOOCT64("1111000001111000",3)` = 170

`@BINTOOCT64("00000101001110010111011 1")` = 01234567

See Also

@BINTOOCT

Entering Number Conversion @Functions

@BITRB

Format

```
@BITRB(Binary, Position)
```

Binary = binary number

Position = bit position; must be ≥ 0 and ≤ number of bits in Binary - 1

@BITRB resets to 0 the given Position bit of a binary value.

Examples

`@BITRB(1010,1)` = 1000

`@BITRB(1010,3)` = 0010

`@BITRB(1100,2)` = 1000

See Also

Entering Boolean @Functions

@BITRH

Format

```
@BITRH(Hex, Position)
```

Hex = hexadecimal number

Position = bit position; must be ≥ 0 and ≤ number of bits in Hex - 1

@BITRH resets to 0 the given Position bit of a hexadecimal value.

Examples

`@BITRH("A",1)` = 8

`@BITRH("A",3)` = 2

`@BITRH("C",2)` = 8

See Also

Entering Boolean @Functions

@BITSB

Format

```
@BITSB(Binary, Position)
```

Binary = binary number

Position = bit position; must be ≥ 0 and ≤ number of bits in Binary - 1

@BITSB sets to 1 the given Position bit of a binary number.

Examples

`@BITSB(1010,2)` = 1110

`@BITSB(1010,0)` = 1011

`@BITSB(1100,0)` = 1101

See Also

Entering Boolean @Functions

@BITSH

Format

```
@BITSH(Hex, Position)
```

Hex = hexadecimal number

Position = bit position; must be ≥ 0 and ≤ number of bits in Hex - 1

@BITSH sets to 1 the given Position bit of a hexadecimal number.

Examples

`@BITSH("A",2)` = E

`@BITSH("C",0)` = D

See Also

Entering Boolean @Functions

@BITTB

Format

```
@BITTB(Binary, Position)
```

Binary = binary number

Position = bit position; must be ≥ 0 and ≤ number of bits in Binary - 1

@BITTB returns the value of the bit of a binary number in the given Position.

Examples

`@BITTB(1010,2)` = 0

`@BITTB(1001,0)` = 1

`@BITTB(1100,1)` = 0

See Also

Entering Boolean @Functions

@BITTH

Format

```
@BITTH(Hex, Position)
```

Hex = hexadecimal number

Position = bit position; must be ≥ 0 and ≤ number of bits in Hex - 1

@BITTH returns the value of the bit of a hexadecimal number in the given Position.

Examples

`@BITTH("A",2)` = 0

`@BITTH("9",0)` = 1

`@BITTH("C",0)` = 0

See Also

Entering Boolean @Functions

@BLOCKNAME

Format

```
@BLOCKNAME(Block)
```

Block = cell or block reference (for example, A1 or B1..B5)

@BLOCKNAME returns the name of a cell or block specified by Block. If Block doesn't contain a name, @BLOCKNAME returns ERR; if Block contains more than one name, @BLOCKNAME arbitrarily returns one of the names.

Note: If the name for a block was created in another notebook, use @BLOCKNAME2.

Example

`BLOCKNAME(D2..D15)` = SALES (block D2..D15 is named SALES)

See Also

@BLOCKNAME2
@BLOCKNAMES
@BLOCKNAMES2

@BLOCKNAME2

Format

```
@BLOCKNAME2(NotebookLink, Block)
```

NotebookLink = a reference to a page, cell, or block in another notebook (for example, [BUDGET]A:A1)

Block = cell or block reference (for example, A1 or B1..B5)

@BLOCKNAME2 returns the block name created in the notebook specified by NotebookLink that refers to Block, which can be in another notebook. If Block doesn't contain a name, @BLOCKNAME2 returns ERR; if Block contains more than one name, @BLOCKNAME2 arbitrarily returns one of the names.

Example

`@BLOCKNAME2([BUDGET]A:A1,A:D2..D15)` = SALES (block A:D2..D15 of the active notebook is named SALES in the name table of notebook BUDGET)

See Also

@BLOCKNAME

@BLOCKNAMES

@BLOCKNAMES2

@BLOCKNAMES

Format

```
@BLOCKNAMES(Block)
```

Block = cell or block reference (for example, A1 or B1..B5)

@BLOCKNAMES returns a two-column table showing the block names that intersect with Block. The left column of the table contains block names, and the right column contains corresponding block coordinates.

If Block doesn't contain a name, @BLOCKNAMES returns ERR.

Because @BLOCKNAMES returns an array, it is automatically enclosed within an @ARRAY @function.

Caution: Make sure there is enough room for a two-column table, with one row for each block name. Quattro Pro overwrites existing data in cells it uses for the table.

Note: If block names for a notebook were created in another notebook, use @BLOCKNAMES2.

Example

This example refers to cells in the next figure. Blocks B3..B7, C3..C7, D3..D7, and B3..D7 are named HOTEL, TRANS, MEALS, and TOTAL, respectively. The example is entered in cell A12.

`@ARRAY(@BLOCKNAMES(B3..D7))` = table in A12..B15 shown in the next figure.

	A	B	C	D
1	WEEKLY EXPENSE REPORT			
2	DATE	HOTEL	TRANS	MEALS
3	05/11	$99.70	$774.23	$67.34
4	05/12	$99.70	$15.00	$89.50
5	05/13	$99.70	$23.00	$97.78
6	05/14	$99.70	$13.00	$75.41
7	05/15	$99.70	$32.00	$63.20
8		$498.50	$857.23	$393.23
9				
10			TOTAL	$1,748.96
11				
12	HOTEL	[C:\QPW\EXPENSES.WB1]A:B3..B7		
13	TRANS	[C:\QPW\EXPENSES.WB1]A:C3..C7		
14	MEALS	[C:\QPW\EXPENSES.WB1]A:D3..D7		
15	TOTAL	[C:\QPW\EXPENSES.WB1]A:B3..D7		

See Also

@ARRAY
@BLOCKNAME
@BLOCKNAME2
@BLOCKNAMES2

@BLOCKNAMES2

Format

```
@BLOCKNAMES2(NotebookLink, Block)
```

NotebookLink = a reference to a page, cell, or block in another notebook (for example, [BUDGET]A:A1)

Block = cell or block reference (for example, A1 or B1..B5)

@BLOCKNAMES2 returns a two-column table showing the block names created in the notebook specified by NotebookLink that refer to blocks that intersect with Block. Use @BLOCKNAMES2 instead of @BLOCKNAMES if the block names for a notebook were created in another notebook. The left column of the output table contains block names, and the right column contains corresponding block coordinates.

If Block doesn't contain a name, @BLOCKNAMES2 returns ERR.

Because @BLOCKNAMES2 returns an array, it is automatically enclosed within an @ARRAY @function.

Caution: Make sure there is enough room for a two-column table, with one row for each block name. Quattro Pro overwrites existing data in cells it uses for the table.

Example

This example refers to cells in the next figure. Blocks B3..B7, C3..C7, D3..D7, and B3..D7 in the active notebook EXPENSES are named HOTEL, TRANS, MEALS, and TOTAL, respectively, in the notebook TRAVEL. The example is entered in cell A12.

`@ARRAY(@BLOCKNAMES2([TRAVEL]A:A1,B3..D7))` = table in A12..B15 shown in the next figure

	A	B	C	D
1	WEEKLY EXPENSE REPORT			
2	DATE	HOTEL	TRANS	MEALS
3	05/11	$99.70	$774.23	$67.34
4	05/12	$99.70	$15.00	$89.50
5	05/13	$99.70	$23.00	$97.78
6	05/14	$99.70	$13.00	$75.41
7	05/15	$99.70	$32.00	$63.20
8		$498.50	$857.23	$393.23
9				
10			TOTAL	$1,748.96
11				
12	HOTEL	[C:\QPW\EXPENSES.WB1]A:B3..B7		
13	TRANS	[C:\QPW\EXPENSES.WB1]A:C3..C7		
14	MEALS	[C:\QPW\EXPENSES.WB1]A:D3..D7		
15	TOTAL	[C:\QPW\EXPENSES.WB1]A:B3..D7		

See Also

@ARRAY

@BLOCKNAME
@BLOCKNAME2
@BLOCKNAMES

@BUSDAY

Format

```
@BUSDAY(Date, <Direction>, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing a date
Direction	=	flag specifying direction of adjustment; 0 = forward; 1 = backward; 2 = forward if in same month as Date, otherwise backward (the default is 0)
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@BUSDAY returns Date if it's a valid business day. If Date falls on a Saturday (and Saturday is set to 0 or omitted), Sunday (and Sunday is set to 0 or omitted), or holiday, @BUSDAY returns the date of the closest valid business day in the direction specified by Direction.

Example

This formula calculates the closest business day after December 25, 1993, assuming that the 25th is a holiday:

`@BUSDAY(@DATE(93,12,25),0,@DATE(93,12,25))` = 34330 (December 27, 1993)

See Also

Setting Holidays

@FBDAY

@LBDAY

@CATB

Format

```
@CATB(Binary1, <HiBit1>, <LoBit1>, <Binary2>, <HiBit2>, <LoBit2>, <Bits>)
```

Binary1	=	first binary number
HiBit1	=	highest bit of the first number to use for concatenation; the default is the most significant bit
LoBit1	=	lowest bit of the first number to use for concatenation; the default is 0
Binary2	=	second binary number
HiBit2	=	highest bit of the second number to use for concatenation; the default is the most significant bit
LoBit2	=	lowest bit of the second number to use for concatenation; the default is 0
Bits	=	number of binary digits to return; must be $\leq$ 64

@CATB joins together two given binary numbers or extracts selected bits from one binary number. Specify high bit and low bit values if you want to use only a portion of a number for concatenation. For example, if HiBit1 = 2 and LoBit1 = 0, only the first three bits of Binary1 are joined with Binary2.

Examples

`@CATB("1100",2,0,"0011",1,0)` = 10011

`@CATB("1100",2,0,"0011",1,0,3)` = 011

`@CATB("1100",3,0,"11",1,0,8)` = 00110011

`@CATB("101101",4,1)` = 0110

See Also

Entering Boolean @Functions

@CATH

Format

```
@CATH(Hex1, <HiBit1>, <LoBit1>, <Hex2>, <HiBit2>, <LoBit2>, <Bits>)
```

Hex1	=	first hexadecimal number
HiBit1	=	highest bit of the first number to use for concatenation; the default is the most significant bit
LoBit1	=	lowest bit of the first number to use for concatenation; the default is 0
Hex2	=	second hexadecimal number
HiBit2	=	highest bit of the second number to use for concatenation; the default is the most significant bit
LoBit2	=	lowest bit of the second number to use for concatenation; the default is 0
Bits	=	number of equivalent binary digits to return; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@CATH joins together two given hexadecimal numbers or extracts selected bits from one hexadecimal number. Specify high bit and low bit values if you want to use only a portion of a number for concatenation. For example, if HiBit1 = 2 and LoBit1 = 0, only the first three bits of Hex1 are joined with Hex2.

Examples

`@CATH("C",2,0,"3",1,0)` = 13

`@CATH("C",2,0,"3",1,0,3)` = 3

`@CATH("C",3,0,"3",1,0,8)` = 33

`@CATH("CA",6,2)` = 12

See Also

Entering Boolean @Functions

@CATNB

Format

```
@CATNB(n, Binary1, <Binary2>, <Binary3>, ..., <BinaryN>, <Bits>)
```

n	=	number of binary numbers being concatenated; n ≤ 64
Binary1	=	first binary number
Binary2,Binary3,...,BinaryN	=	second through the nth binary numbers
Bits	=	number of binary digits to return; must be ≤ 64

@CATNB joins together n binary numbers.

Examples

`@CATNB(3,1,0,1010)` = 101010

`@CATNB(3,1,0,1010,4)` = 1010

`@CATNB(3,11,"00",11,8)` = 00110011

See Also

Entering Boolean @Functions

@CATNH

Format

```
@CATNH(n, Hex1, <Hex2>, <Hex3>, ..., <HexN>, <Bits>)
```

n	=	number of hexadecimal numbers being concatenated; n ≤ 16
Hex1	=	first hexadecimal number
Hex2,Hex3,...,HexN	=	second through the nth hexadecimal numbers
Bits	=	number of equivalent binary digits to return; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@CATNH joins together n hexadecimal numbers.

Examples

`@CATNH(3,"1","0","A")` = 10A

`@CATNH(3,"1","0","A",4)` = A

`@CATNH(3,"A","B","C",16)` = 0ABC

See Also

Entering Boolean @Functions

@CDAYS

Format

```
@CDAYS(StartDate, EndDate, <Calendar>, <February>)
```

StartDate = number representing the start date

EndDate = number representing the end date

Calendar = flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

February = 0 to use 30-day treatment of February for 30/360 calendar; 1 to use the actual-day treatment (the default is 0)

@CDAYS returns the number of calendar days between StartDate and EndDate, including EndDate in the total. If EndDate precedes StartDate, the result is negative.

You can use Calendar to specify whether the actual or 30/360 day calendar is used. Under the actual calendar, Quattro Pro calculates the number of days by subtracting one date from the other.

To handle months with more than 30 days (and February), @CDAYS sometimes adjusts StartDate or EndDate before the sum is calculated. @CDAYS adjusts StartDate to fall on the 30th if either of the following two conditions are true: the day of the month on which StartDate falls is greater than 30, or StartDate falls on the last day of February (28th or 29th, depending on year) and February is 0.

If StartDate falls on the 30th (either because @CDAYS adjusted it or it falls on the 30th) and the day of the month on which EndDate falls is greater than 30, then EndDate also adjusts to fall on the 30th. By default, @CDAYS treats the last day of February as the 30th. To prevent this, set February to 1.

Example

`@CDAYS(@DATE(93,1,23),@DATE(95,6,28))` = 875

See Also

@BDAYS

@HOLS

@CEILING

Format

```
@CEILING(X, Y)
```

X = value to round

Y = value to make rounded x evenly divisible by

@CEILING rounds X up (away from zero) to the nearest value that's evenly divisible by Y. If X and Y have different signs, the result of @CEILING is ERR.

Examples

`@CEILING(22,5)` = 25

`@CEILING(5.7,0.2)` = 5.8

`@CEILING(-3.2,-2)` = -4

`@CEILING(-3.2,2)` = ERR

See Also

@EVEN
@FLOOR
@INT
@MROUND
@ODD
@ROUND

@CHIDIST

Format

```
@CHIDIST(X, DegFreedom)
```

X = value at which to evaluate the function; must be ≥ 0

DegFreedom = integer number of degrees of freedom in the distribution; must be ≥ 1

@CHIDIST returns the cumulative chi-square distribution, which is associated with a chi-square tests. Chi-square tests allow you to compare the differences between observed and expected frequencies.

If DegFreedom isn't an integer, @CHIDIST rounds it to the nearest integer.

Examples

`@CHIDIST(36.41503,24)` = 0.05

`@CHIDIST(17.53455,8)` = 0.025

See Also

@CHIINV

@CHIINV

Format

```
@CHIINV(Prob, DegFreedom)
```

Prob = cumulative probability value; must be ≥ 0 and ≤ 1

DegFreedom = integer number of degrees of freedom; must be ≥ 1

@CHIINV computes the inverse of the cumulative one-tailed chi-square distribution. Use @CHIINV to compute the critical value for a test involving a chi-square variable.

If DegFreedom isn't an integer, @CHIINV rounds it to the nearest integer.

Examples

`@CHIINV(0.05,24)` = 36.41503

`@CHIINV(0.025,8)` = 17.53455

See Also

@CHIDIST

@CHITEST

@CHITEST

Format

```
@CHITEST(Actual, Expected)
```

Actual = block containing actual values

Expected = block containing expected values

@CHITEST computes the probability that the actual and expected frequencies are similar by chance. @CHITEST returns the probability for a chi-square test distribution with (r - 1)(c - 1) degrees of freedom, where r = number of rows, and c = number of columns.

Actual and Expected must have the same number of values and must contain multiple rows or columns of data.

Example

This example refers to cells in the next figure. The chi-square statistic for the data in the next figure is 16.25813 and the degrees of freedom is 4.

`@CHITEST(C3..E5,C7..E9)` = 0.002692

	A	B	C	D	E
1				Soft Drink Flavors	
2		Age Ranges	Cola	Orange	Lemon-lime
3	Actual	Under 25	120	65	55
4		26-50	100	45	85
5		Over 50	75	35	70
6					
7	Expected	Under 25	108.93	53.53	77.54
8		26-50	104.38	51.31	74.31
9		Over 50	81.69	40.16	58.15

See Also

@CHIDIST

@CHIINV

@COMB

Format

```
@COMB(R, N)
```

R = number of elements in each subgroup selected from group N; must be ≥ 0

N = number of elements in the group; must be ≥ 0

@COMB calculates the number of combinations (unordered subgroups of size R) that you can form out of a group of size N. If N < R @COMB returns 0.

The formula for calculating the number of combinations, if $R \leq N$, is

$$\frac{N!}{R!\,(N-R)!}$$

Example

Given eleven marbles, this formula calculates how many ways a subset of 5 marbles can be constructed such that no two constructions contain the same 5 marbles:

`@COMB(5,11)` = 462

See Also

@PERMUT

@COMPLEX

Format

```
@COMPLEX(X, Y)
```

X = numeric value representing real coefficient of complex number

Y = numeric value representing imaginary coefficient of complex number

@COMPLEX converts X and Y into a complex number.

Example

`@COMPLEX(5,7)` = "5+7i"

See Also

@IMAGINARY

@IMREAL

@CONFIDENCE

Format

```
@CONFIDENCE(Alpha, SDev, Size)
```

Alpha = significance level; the percentage of the normal curve that is outside the confidence interval (1 - Alpha); for example, if the confidence interval is 95%, Alpha = 5%; must be > 0 and < 1

SDev = population standard deviation; must be > 0

Size = sample size; must be ≥ 1

@CONFIDENCE computes the confidence interval around the mean for a given sample size, using the normal distribution function. Given a specified degree of confidence, the confidence interval indicates that the population mean will be within that interval. Use @CONFIDENCE to apply levels of confidence to sample data and to determine margins of error.

Example

Out of 1000 people sampled, 490 said they would vote for Candidate A. If the population standard deviation is 0.5, this formula returns the 95% confidence interval for the population mean:

`@CONFIDENCE(0.05,0.5,1000)` = 0.03099

Pollsters can report that Candidate A will receive 49% of the vote with a margin of error of 3.1%.

@CONVERT

Format

```
@CONVERT(X, FromUnit, ToUnit)
```

X = numeric value in FromUnit to convert, in the units specified by FromUnit

FromUnit = unit type of the value X (must be on the list of supported unit names)

ToUnit = units to convert the value X into; must be on the list of supported unit names

@CONVERT changes X, which is expressed in FromUnit units, to the equivalent value in ToUnit units. Column Unit of the following tables lists the measurement units that you can specify in FromUnit and ToUnit. Each argument is case sensitive.

Mass measurement units

Mass	Unit
Gram	"g"
Slug	"sg"
Pound mass (avoirdupois)	"lbm"
U (atomic mass unit)	"u"
Ounce mass (avoirdupois)	"ozm"

Pressure measurement units

Pressure	Unit
Pascal	"p"
Atmosphere	"at"

Distance measurement units

Distance	Unit
Meter	"m"
Statute mile	"mi"
Nautical mile	"Nmi"
Inch	"in"
Foot	"ft"
Yard	"yd"
Angstrom	"ang"

Time measurement units

Time	Unit
Year	"yr"
Day	"day"

Hour	"hr"
Minute	"mn"
Second	"sec"

Force measurement units

Force	Unit
Newton	"N"
Dyne	"dy"
Pound force	"lbf"

Energy measurement units

Energy	Unit
Joule	"J"
Erg	"e"
Thermodynamic calorie	"c"
IT calorie	"cal"
Electron volt	"ev"
Horsepower-hour	"hh"
Watt-hour	"wh"
Foot-pound	"flb"
BTU	"btu"

Power measurement units

Power	Unit
Horsepower	"h"
Watt	"w"

Magnetic measurement units

Magnetism	Unit
Tesla	"T"
Gauss	"ga"

Temperature measurement units

Temperature	Unit
Celsius	"cel"
Fahrenheit	"fah"
Kelvin	"kel"
Rankine	"ran"

Liquid measurement units

Liquid	Unit
Teaspoon	"tsp"
Tablespoon	"tbs"
Fluid ounce	"oz"
Cup	"cup"
Pint	"pt"
Quart	"qt"
Gallon	"gal"
Liter	"lt"

If a metric unit is used (for example, Gram, Meter, or Liter), you can preface it with one of the prefixes listed in the next table. Use the metric prefixes to multiply a metric unit by a power of 10.

Metric prefixes

Metric Prefix	Multiplier	Unit Prefix
exa	1E+18	"E"
peta	1E+15	"P"
tera	1E+12	"T"
giga	1E+09	"G"
mega	1E+06	"M"
kilo	1E+03	"k"
deka	1E+01	"e"
deci	1E-01	"d"
centi	1E-02	"c"
milli	1E-03	"m"
micro	1E-06	"u"
nano	1E-09	"n"
pico	1E-12	"p"
femto	1E-15	"f"
atto	1E-18	"a"

Whenever @CONVERT is used, both FromUnit and ToUnit must come from the same table.

To determine a specific conversion factor, use 1 for X.

Examples

`@CONVERT(2,"day","hr")` = 48

`@CONVERT(3.5,"kg","lbm")` = 7.71618

`@CONVERT(2.5,"oz","mlt")` = 73.94991

See Also
@IMAGINARY
@STRING
@VALUE

@CORREL

Format

```
@CORREL(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@CORREL computes the correlation coefficient of the numeric values in Array1 and Array2. Use @CORREL to ascertain the relationship between two sets of data. If two data sets change in a related matter based on the input that generates them, they are said to be correlated.

Array1 and Array2 must have the same number of values. Also, the values in Array1 and Array2 must show some variance.

Examples

These examples refer to cells in the next figure.

`@CORREL(A2..A9,B2..B9)` = 0.994135

`@CORREL(A2..A9,C2..C9)` = 0.460718

`@CORREL(A2..A9,D2..D9)` = -0.52494

`@CORREL(B2..B9,C2..C9)` = 0.547422

	A	B	C	D
1	**X1**	**X2**	**X3**	**X4**
2	1	2	-1	-9
3	2	3	-7	-3
4	3	4	2	6
5	4	5	8	3
6	5	6	-4	-2
7	6	7	0	-21
8	7	8	-12	0
9	8	10	45	-33

See Also

@COVAR

@COUPDAYBS

Format

```
@COUPDAYBS(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPDAYBS returns the number of days from the beginning of the coupon period of a bond to the settlement date.

Example

A bond's settlement date is May 15, 1992 and its maturity date is February 15, 1996. This formula calculates the number of days from the beginning of the coupon period to the settlement date:

```
@COUPDAYBS(@DATE(92,5,15),@DATE(96,2,15))
```
= 90

@COUPDAYS

Format

```
@COUPDAYS(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPDAYS returns the number of days in the coupon period of a bond that contains the settlement date.

Example

A bond's settlement date is May 15, 1992 and its maturity date is February 15, 1996. This formula calculates the number of days in the coupon period that contains the settlement date.

`@COUPDAYS(@DATE(92,5,15),@DATE(96,2,15))` = 180

@COUPDAYSNC

Format

```
@COUPDAYSNC(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPDAYSNC returns the number of days from the settlement date of a bond to the next coupon date.

Example

A bond's settlement date is May 15, 1992 and its maturity date is February 15, 1996. This formula calculates the number of days between the settlement date and the next coupon date:

`@COUPDAYSNC(@DATE(92,5,15),@DATE(96,2,15))` = 90

@COUPNCD

Format

```
@COUPNCD(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPNCD returns the serial date number for the next coupon date after the settlement date of a bond.

Example

A bond pays a coupon semiannually and matures on August 31, 2003. This formula calculates the date of the next coupon payment after December 17, 1992:

`@COUPNCD(@DATE(92,12,17),@DATE(103,8,31))` = 34028 (February 28, 1993)

@COUPNUM

Format

```
@COUPNUM(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPNUM returns the number of coupons payable between the settlement date and maturity date of a bond.

Example

A bond's settlement date is March 15, 1994 and its maturity date is April 15, 2004. This formula calculates the number of annual coupon payments there are until the maturity date:

```
@COUPNUM(@DATE(94,3,15),@DATE(104,4,15),1)
```
= 11

@COUPPCD

Format

```
@COUPPCD(Settle, Maturity, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@COUPPCD returns the serial date number for the coupon date just before the settlement date of a bond.

Example

A bond's settlement date is December 17, 1992 and its maturity date is August 31, 1999. This formula calculates the date of the previous semiannual coupon payment before the settlement date:

`@COUPPCD(@DATE(92,12,17),@DATE(99,8,31))` = 33847 (August 31, 1992)

@COVAR

Format

```
@COVAR(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@COVAR returns the covariance by taking the deviations for each corresponding element in Array1 and Array2, computing their products, and taking the average of their average. Array1 and Array2 must have the same number of values. Use @COVAR to analyze the relationship between two data sets. The covariance is calculated using this formula:

$$c = \frac{1}{n-1}\sum_{i}(x_i - \bar{x})(y_i - \bar{y})$$

Examples

These examples refer to cells in the next figure.

`@COVAR(A2..A9,B2..B9)` = 5.6875

`@COVAR(A2..A9,C2..C9)` = 21.75

`@COVAR(A2..A9,D2..D9)` = 12.3125

`@COVAR(B2..B9,C2..C9)` = 26.21875

	A	B	C	D
1	**X1**	**X2**	**X3**	**X4**
2	1	2	1	9
3	2	3	2	3
4	3	4	2	6
5	4	5	8	3
6	5	6	4	2
7	6	7	0	21
8	7	8	12	0
9	8	10	45	33

See Also

@CORREL

@CRITBINOM

Format

```
@CRITBINOM(Trials, Prob, Alpha)
```

Trials	=	integer number of Bernoulli trials; must be ≥ 0
Prob	=	probability of success per trial; must be ≥ 0 and ≤ 1
Alpha	=	critical probability to test; must be ≥ 0 and ≤ 1

@CRITBINOM calculates the maximum number of successes that can occur before the cumulative probability expressed by Alpha is exceeded for the number of Trials. @CRITBINOM has applications in quality assurance. For example, you could use @CRITBINOM to calculate the maximum number of defects allowed in a shipment.

Example

Company A tests a sample of 100 electrical circuits received from Company B. The probability that a circuit is defective is 7%. Using an Alpha value of 7.4%, this formula calculates the maximum number of defective circuits that can be expected.

```
@CRITBINOM(100,0.07,0.074)
```
= 3

See Also

@BINOMDIST

@NEGBINOMDIST

@DELTA

Format

```
@DELTA(X, <Y>)
```

X = numeric value to check

Y = numeric value that X must equal for the function to return 1 (if omitted, assumed to be zero)

@DELTA tests whether X and Y are equal. If they are, @DELTA returns 1 (True); if not, @DELTA returns 0 (False).

Examples

`@DELTA(1,2)` = 0

`@DELTA(2,2)` = 1

`@IF(@DELTA(2,2),"Equal","Not Equal")` = Equal

See Also

@GESTEP

@DEVSQ

Format

```
@DEVSQ(List)
```

List = one or more numeric or block values

@DEVSQ returns the sum of the squares of the deviations of the numbers in List from their mean value.

@DEVSQ uses this formula:

$$\Sigma(x - \bar{x})^2$$

Example

`@DEVSQ(9,10,12,14,15)` = 26

See Also

@AVEDEV

@DFRAC

Format

```
@DFRAC(Dec, Denom)
```

Dec = number to be converted, expressed as a decimal

Denom = denominator; must be an integer

@DFRAC converts a number expressed as a decimal to a fraction using the specified denominator. @DFRAC reverses the effect of @FRACD.

The result looks like a decimal, but the portion to the right of the decimal point is actually the numerator of the fraction using the specified denominator. For example, you can use @DFRAC to convert a decimal to 32nds. Converting 99.375 to 32nds results in 99.12, representing 99 / .

Tip: Format the cell that contains the @function to show the same number of decimal places as the number of digits in the desired Denom. For example, if Denom is 32, set the cell format to display two decimal places.

Example

`@DFRAC(106.4375,32)` = 106.14; the 14 to the right of the decimal place signifies / .

See Also

@FRACD

@DISC

Format

```
@DISC(Settle, Maturity, Price, <Redemption>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Price	=	settlement price per 100 face value; must be ≥ 0 and ≤ 100
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@DISC computes the discount rate for a security, which is the percentage discount offered on a security for a 360-day or 365-day term.

@DISC computes the discount rate using this formula:

$$D = \left(1 - \frac{P}{R}\right) * \left(\frac{t_b}{M - S}\right)$$

```
D = discount rate
P = price
R = redemption
b = basis
M = maturity
S = settle
```

t_b is the number of days over which the discount rate applies (360 or 365).

Example

This formula calculates the discount rate for a bond with the following terms: Settle is May 27, 1995, Maturity is November 24, 1995, Price is 96.2492, Redemption is 100, and Calendar is 2 (actual/360).

```
@DISC(@DATE(95,5,27),@DATE(95,11,24),96.2492,100,2)
```
= 0.074602

@DURAT

Format

```
@DURAT(Discrate, Flows, <Initial>, <[Odd|Periods]>, <Simp>, <Pathdep>,
  <Filter>, <Start>, <End>)
```

Discrate	=	discount rate or a block containing discount rates that correspond to cash flows stored in Flows
Flows	=	block containing cash flows associated with the discount rates in Discrate
Initial	=	initial cash flow (the default is 0)
Odd|Periods	=	delay between initial and first cash flow, in number of periods (the default is 1) or block containing lengths of periods between cash flows (the default is 1)
Simp	=	flag specifying how to discount: 0 = compounded discounting (default) 1 = mixed compounded and simple discounting 2 = simple discounting
Pathdep	=	flag specifying whether to apply path-dependent compounding to each flow; 0 = no path (default); 1 = path
Filter	=	flag specifying filter type: 0 = no filter (default); 1 = cashflow < Start; 2 = cashflow ≤ Start; 3 = cashflow > Start; 4 = cashflow ≥ Start; 5 = Start < cashflow < End; 6 = Start ≤ cashflow ≤ End
Start	=	a starting cash flow amount to compare against individual flows
End	=	an ending cash flow amount to compare against individual flows

@DURAT calculates the duration of a given cash flow structure. Duration (also called Macaulay duration) is defined as the weighted average time to receipt of a cash flow where the present values of the cash flows are the weights. Each weight in the sum is the present value of a cash flow divided by the net present value of all the cash flows.

@DURAT computes Macaulay duration using this formula:

$$Du = \frac{\sum_{i=1}^{n} Di_i * DF_i * Fl_i}{I + \sum_{i=1}^{n} DF_i * Fl_i}$$

where

$$Di_i = \sum_{j=1}^{i} Fl_i$$

```
Di = Distance
Du = Duration
Fl = Flows
I = Initial
```

n is the number of cash flows. is the discount factor corresponding to the ith flow.

Modified (or Hicks) duration is defined as a sensitivity of present value to change in the internal rate of return. Modified duration is not defined for multiple discount rates (when Discrate is a block of discount rates).

To convert Macaulay duration to Modified (or Hicks) duration, use this formula:

```
modified duration = Macaulay duration / (1 + Discrate)
```

Example

Consider a cash flow stream comprising four flows of $5, followed by four flows of $10, followed by seven flows of $11, followed by a final flow of $110. The first flow is 0.56745 periods away. The next 11 flows occur one period apart. The last four flows are 1.5 periods apart. This formula calculates the the duration, assuming compound discounting, no initial cash flow, and the data shown in the next figure:

`@DURAT(D8,A2..B5,B8,C2..D4)` = 10.28273

	A	B	C	D	E
1	Cash Flows		Periods		
2	4	$5	1	.056745	
3	4	$10	11	1	
4	7	$11	4	1.5	
5	1	$110			
6					
7	Initial		Discount Rate		
8		0		7.85%	
9					

See Also

@DURATION

Entering Cash Flow @Functions

@DURATION

Format

```
@DURATION(Settle, Maturity, Coupon, Yield, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Coupon	=	coupon rate; must be ≥ 0
Yield	=	annual yield; must be > 0 and ≤ 1
Freq	=	frequency of coupon payments in the number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@DURATION returns the Macaulay duration for a bond with an assumed par value of 100. Macaulay duration is the weighted average maturity of a bond's cash flow stream where the present values of all future cash receipts are used as weights.

Example

The following formula calculates the Macaulay duration of a bond with these terms: Settle is August 8, 1992, Maturity is November 15, 1998, Coupon is 9%, and Yield is 8.816%.

```
@DURATION(@DATE(92,8,8),@DATE(98,11,15),0.09,0.08816)
```
= 4.836099

@EMNTH

Format

```
@EMNTH(Date)
```

Date = number representing a date

@EMNTH returns the serial date number for the date of the last day of the month in which Date falls.

Example

`@EMNTH(@DATE(96,2,14))` = 35124 (February 29, 1996), the last day of the month in which February 14, 1996 falls.

See Also

@LBDAY

@LWKDAY

@ERF

Format

```
@ERF(Lower, <Upper>)
```

Lower	=	lower bound for integrating @ERF; must be ≥ 0
Upper	=	upper bound for integrating @ERF; if omitted, @ERF integrates the error function between 0 and Lower

@ERF returns the error function integrated between Lower and Upper. The error function helps solve partial differential equations that involve convection or diffusion.

The equation for @ERF(z) is

$$\frac{2}{\sqrt{\pi}} \int_0^z e^{-t^2} dt$$

The equation for @ERF(a,b) is

$$\frac{2}{\sqrt{\pi}} \int_a^b e^{-t^2} dt$$

This is the same as @ERF(b) minus @ERF(a).

Example

`@ERF(0,1)` = 0.842701

See Also

@ERFC

@ERFC

Format

```
@ERFC(Lower)
```

Lower = lower bound for integrating @ERF; must be ≥ 0

@ERFC returns the complementary error function, which derives from the error function @ERF using this formula for @ERFC(x):

$$\frac{2}{\sqrt{\pi}} \int_{x}^{\infty} e^{-t^2} dt$$

This is the same as 1 - @ERF(x).
Therefore, @ERFC(Lower) = 1 - @ERF(Lower,Upper).

Example

`@ERFC(1)` = 0.157299

See Also

@ERF

@EVEN

Format

@EVEN(X)

X = value to round

@EVEN rounds X up (away from zero) to the nearest even integer.

Examples

`@EVEN(3.2)` = 4

`@EVEN(-3.2)` = -4

See Also

@CEILING
@FLOOR
@INT
@MROUND
@ODD
@ROUND

@EXPONDIST

Format

```
@EXPONDIST(X, Lambda, Cum)
```

X	=	value at which to evaluate function; must be ≥ 0
Lambda	=	value to indicate; Lambda = 1/Mean; must be > 0
Cum	=	1 to perform cumulative distribution function; 0 to perform the probability density function

The exponential distribution, sometimes called the waiting-time distribution, describes the amount of time or distance between the occurrence of random events (such as the time between major earthquakes or the time between no hitters pitched in major league baseball). The exponential distribution calculated by @EXPONDIST is a continuous distribution with a probability density function whose formula is:

$$f(X; \lambda) = \lambda e^{-\lambda x}$$

For the cumulative distribution function, the formula is:

$$F(X; \lambda) = 1 - e^{-\lambda x}$$

Use this distribution in connection with estimating the length of material life, or the length of time a process might take.

Examples

On average, customers at a certain bank must wait 2 minutes before being served by a teller. This formula calculates the probability that someone would have to wait 3 minutes:

`@EXPONDIST(3,1/2,0)` = 0.111565

This formula calculates the probability that someone would wait only 1 minute for a teller:

`@EXPONDIST(1,1/2,1)` = 0.393469

@FACT

Format

`@FACT(N)`

N = integer ≥ 0 specifying the factorial to calculate

@FACT calculates the factorial of a number, N!, defined as follows: if N ≥ 0,

`N! = N × (N-1) × (N-2) × (N-3) ×...× (2) × (1)`

@FACT(0) returns 1. If N is a non-integer or negative number, @FACT returns ERR.

Examples

`@FACT(10)` = 3628800

`@FACT(128)` = 3.9E+215

See Also

@FACTDOUBLE

@FACTDOUBLE

Format

`@FACTDOUBLE(N)`

N = value ≥ 0 to calculate factorial of

@FACTDOUBLE returns the double factorial of N, N!!, defined as follows:

If N is even, N!! = N(N-2)(N-4)...(4)(2)

If N is odd, N!! = N(N-2)(N-4)...(3)(1)

If N is negative, @FACTDOUBLE returns ERR.

Examples

`@FACTDOUBLE(12)` = 46080

`@FACTDOUBLE(13)` = 135135

See Also

@FACT

@FBDAY

Format

```
@FBDAY(Date, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing a date
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@FBDAY returns the serial date number of the first business day of the month in which Date falls. If the first of the month isn't a business day, @FBDAY returns the business day closest to it within the same month.

Example

This formula calculates the first business day in January 1993, assuming that Saturdays and some dates are holidays:

`@FBDAY(@DATE(93,1,1),A7..C9,0,1)` = 33972 (January 3, 1993)

See Also

Setting Holidays

@BUSDAY

@LBDAY

@FDIST

Format

```
@FDIST(X, DegFreedom1, DegFreedom2)
```

X	=	positive value at which to evaluate the function DegFreedom1
DegFreedom1	=	numerator degrees of freedom; must be ≥ 1
DegFreedom2	=	denominator degrees of freedom; must be ≥ 1

@FDIST returns the cumulative F-distribution function, which is the probability that a random variable will be less than X. Use @FDIST to compare two population variances.

Example

`@FDIST(6.256057,5,4)` = 0.05

See Also

@FINV
@FTEST

@FEETBL

Format

```
@FEETBL(Tu, Ppu, [StdTbl|Val], <[MinTbl|Val]>, <[MaxTbl|Val]>, <RndPlcs>)
```

Tu	=	total units; if Tu is negative, @FEETBL uses its absolute value
Ppu	=	price per unit
StdTbl|Val	=	fee table or a single value that defines the standard fee calculation
MinTbl|Val	=	fee table or a single value that defines the minimum fee calculation (if omitted, MinTbl equals StdTbl)
MaxTbl|Val	=	fee table or a single value that defines the maximum fee calculation (if omitted, MaxTbl equals StdTbl)
RndPlcs	=	number of places to which the final result is rounded; can be from 0 to 10 places (the default is no rounding)

@FEETBL returns fee calculations from tables. You can use @FEETBL to calculate fees or commissions for many types of stock transactions, taxes, sales commissions, and other types of fees and charges. To use @FEETBL, you need to create a table (or tables) that describe the fees.

@FEETBL is more powerful than other table lookup @functions such as @HLOOKUP and @VLOOKUP because it allows you to


Compare the standard fee with minimum and maximum values


Multiply the lookup value by the number of units or total price


Add a fixed value to the fee


Round the result to a specified number of decimal places

If the fee table is indexed by values of total units or price per unit, Tu or Ppu must be greater than the smallest value in the index; otherwise, @FEETBL cannot find a lookup value. If either Tu or Ppu is zero, @FEETBL returns zero.

If you specify an optional argument, such as RndPlcs, you must also specify all preceding optional arguments. If MinTbl and MaxTbl are not pertinent to the fee calculation, use StdTbl again for MinTbl and MaxTbl, or enter values that have no effect on the final result. For example, enter 0 for MinTbl and 1E+99 for MaxTbl.

Note: For valid comparison, values for StdTbl, MinTbl, and MaxTbl arguments must have the same units.

The upper left cell of a fee table must contain a table header string that identifies the row index, column index, and cell contents of the table, and also specifies if the table contains an additive factor for the fee calculation. The table header string consists of three or four parameters separated by a space; each parameter has several possible values.

Table header parameters

Parameter	Description	Values
1	Row index	tu, ppu, tp, na

2	Column index	tu, ppu, tp, na
3	Cell contents	fpu, fpct, luo
4	Additive factor	fa

Description Values

tu = total units

fpu = fee per unit

ppu = price per unit

fpct = fee percentage

tp = total price

luo = lookup only

na = not applicable

fa = fixed adder

The first parameter of the table header identifies the contents of the row index, which appears in the first column of the table below the table header. The second parameter identifies the contents of the column index, which appears in the first row of the table to the right of the table header.

The third parameter of the table header determines if @FEETBL multiplies the lookup value from the table by another value. For example, "fpu" (fee per unit) indicates that @FEETBL multiplies the lookup value by the number of units; "fpct" (fee percentage) indicates that the lookup value is a percentage that @FEETBL multiplies by the total price; "luo" (lookup only) indicates that @FEETBL uses the lookup value without modification.

The fourth parameter of the table header is an optional additive factor; specify "fa" (fixed adder) to add a value to the result of the operation specified by the third parameter. If the fee table has no additive factor, omit the fourth parameter.

In the next figure, the table header in cell A3 is "tp na fpct fa"; "tp" indicates that A4..A9 represents the row index values for total price; "na" indicates that B3..C3 has no column index values; "fpct" indicates that the lookup values in B4..B9 are percentages that must be multiplied by the total price; "fa" indicates that the values in C4..C9 are "fixed adders", that is, one of these values must be added to the product of the fee percentage and the total price.

	A	B	C	D
1	Standard Commission Rate Table			
2	Principal	%Fee +	Fixed Adder	
3	**tp na fpct fa**			
4	**$0**	1.60%	$26.00	
5	**$2,500**	0.60%	$51.00	
6	**$6,000**	0.30%	$69.00	
7	**$22,000**	0.20%	$91.00	
8	**$50,000**	0.10%	$141.00	
9	**$500,000**	0.08%	$241.00	
10				

In the next figure, the table header in cell A2 is "tu tp luo"; "tu" indicates that A3..A7 represents the row index values for total units; "tp" indicates that B2..E2 represents the column index values for total price; "luo" indicates that @FEETBL uses the lookup values in B3..E7.

	A	B	C	D	E	F
1	Total Units			Total Price		
2	**tu tp luo**	**$0**	**$10,000**	**$15,000**	**$20,000**	
3	0	$250	$500	$750	$1,000	
4	2	$200	$400	$600	$800	
5	5	$175	$350	$525	$700	
6	10	$150	$300	$450	$600	
7	20	$125	$250	$375	$500	
8						

@FEETBL treats all string values (other than the table header) or empty cells in fee tables as zero.

Examples

A furniture manufacturer sells 100 bookcases at a price of $150 each to a retailer. This formula calculates the handling fee for the order based on the fee table in the next figure.

```
@FEETBL(100,150,A1..D5) = $300
```

Cell A4 is the row index value for 100 total units. Cell C1 is the column index value for $150 price per unit. Cell C4 is the lookup value, which is multiplied by the total units: $3 * 100 = $300.

This formula calculates the handling fee based on the fee table in the next figure for a sale of 5 end tables at a price of $75 each:

```
@FEETBL(5,75,A1..D5)
```
= $25

Cell A2 is the row index value for 5 units (between 0 and 9). Cell B1 is the column index value for $75 price per unit. Cell B2 is the lookup value, which is multiplied by the total units: $5 * 5 = $25.

	A	B	C	D	E
1	**tu ppu fpu**	**$0**	**$100**	**$500**	
2	0	$5	$7	$8	
3	10	$4	$5	$6	
4	100	$2	$3	$4	
5	1000	$2	$1	$2	
6					

See Also

@HLOOKUP

@VLOOKUP

@FIB

Format

`@FIB(N)`

N = integer ≥ 0 specifying the desired term of a Fibonacci sequence

@FIB calculates the Nth term of a Fibonacci sequence (1, 1, 2, 3, 5, 8, 13, 21...), in which each number, after the first two, is the sum of the two numbers immediately preceding it. @FIB(0) is defined to be 0.

Examples

`@FIB(4)` = 3

`@FIB(9)` = 34

`@FIB(15)` = 610

@FINV

Format

```
@FINV(Prob, DegFreedom1, DegFreedom2)
```

Prob = cumulative probability value; must be ≥ 0 and ≤ 1

DegFreedom1 = numerator degrees of freedom; must be ≥ 1

DegFreedom2 = denominator degrees of freedom; must be ≥ 1

@FINV returns the inverse of the cumulative F-distribution function. Use this function to measure the degree of variability in two data sets.

Example

`@FINV(0.05,5,4)` = 6.256057

See Also

@FDIST

@FTEST

@FIRSTBLANKPAGE

Format

```
@FIRSTBLANKPAGE(Block)
```

Block = a cell or block reference; can be a link to another opened notebook (for example, [BUDGET]A:A1)

@FIRSTBLANKPAGE returns a string that contains the letters for the first unnamed blank page in a notebook that isn't part of a group.

Quattro Pro searches for the first unnamed blank page (that isn't in a group) starting at page A and continuing toward page IV. If there are no unnamed blank pages (or they're all in groups), @FIRSTBLANKPAGE returns ERR.

Example

`@FIRSTBLANKPAGE(B17)` = AA (if it is the first page that is blank and unnamed)

See Also

@LASTBLANKPAGE

@FIRSTINGROUP

Format

```
@FIRSTINGROUP(Block, GroupName)
```

Cell = a cell or block of the notebook to check

GroupName = a string value representing a group name

@FIRSTINGROUP returns a string that contains the letters for the first page in the group named GroupName. @FIRSTGROUP searches the notebook referenced by Block for the group. If the group doesn't exist, @FIRSTINGROUP returns ERR.

Example

`@FIRSTINGROUP([REPORTQ4]A:C12,"Totals")` = "A" (if the notebook REPORTQ4 contains a group named Totals that starts with page A)

See Also

@LASTINGROUP

@FISHER

Format

```
@FISHER(X)
```

X = numeric value; -1 < X < 1

@FISHER returns the Fisher transformation at the value X. Fisher's z-transformation is used to produce an approximately normally distributed variable (rather than skewed) from the correlation coefficient. The formula @FISHER uses is

$$z' = \frac{1}{2} \ln\left(\frac{1 + x}{1 - x}\right)$$

Example

```
@FISHER(0.25)
```
= 0.255413

See Also

@FISHERINV

@PEARSON

@FISHERINV

Format

```
@FISHERINV(Y)
```

Y = numeric value ≤ 354 for which you want the inverse of the Fisher transformation

@FISHERINV returns the inverse of the Fisher transformation. Use @FISHERINV to determine the confidence limits for a correlation coefficient.

Example

`@FISHERINV(0.255413)` = 0.25

See Also

@FISHER

@PEARSON

@FLOOR

Format

```
@FLOOR(X, Y)
```

X = value to round

Y = value to make rounded X evenly divisible by

@FLOOR rounds X down (toward zero) to the nearest value that's evenly divisible by Y. If X and Y have different signs, the result of @FLOOR is ERR.

Examples

`@FLOOR(3.2,3)` = 3

`@FLOOR(-3.2,-3)` = -3

See Also

@CEILING
@EVEN
@MROUND
@ODD
@INT
@ROUND

@FORECAST

Format

```
@FORECAST(X, KnownY, KnownX)
```

X = numeric value at which to evaluate the function

KnownY = dependent range of values

KnownX = independent range of values

@FORECAST returns a predicted Y value corresponding to X based upon a linear regression of KnownY and KnownX.

KnownY and KnownX must contain the same number of values. The variance of KnownX must not be 0.

Example

This example refers to cells in the figure below.

`@FORECAST(1000,C2..C16,B2..B16)` = $15,868.50

	A	B	C
1	Date	Advertising	Sales
2	04/30/93	$435	$7,000
3	05/07/93	$400	$6,000
4	05/14/93	$505	$7,767
5	05/21/93	$470	$7,800
6	05/28/93	$610	$9,534
7	06/04/93	$540	$7,750
8	06/11/93	$575	$8,945
9	06/18/93	$715	$11,301
10	06/25/93	$645	$9,465
11	07/02/93	$680	$10,760
12	07/09/93	$785	$13,000
13	07/16/93	$750	$11,890
14	07/23/93	$855	$12,980
15	07/30/93	$820	$13,068
16	08/06/93	$890	$14,246

@FRACD

Format

```
@FRACD(Frac, Denom)
```

Frac = number to be converted

Denom = denominator; must be an integer

@FRACD converts the fraction Frac to a decimal number. For example, you can use this @function to convert a number with a fractional portion in 32nds to a decimal number. @FRACD reverses the effect of @DFRAC.

Frac looks like a decimal, but @FRACD doesn't use it that way. The portion to the right of the decimal point is the numerator of the fraction using the denominator specified by Denom. For example, if Denom is 32 and you want to find the decimal equivalent of 99 / , set Frac to 99.12. If Denom were 100, setting Frac to 99.12 represents 99 / .

Example

This formula finds the decimal equivalent of 106 / .

```
@FRACD(106.14,32)
```
= 106.4375

See Also

@DFRAC

@FTEST

Format

```
@FTEST(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@FTEST returns the results of an F-test run against the samples in Array1 and Array2. An F-test is a one-tailed probability that the differences in the sample variances in Array1 and Array2 are different. Use @FTEST to determine if two samples have significantly different variances (that is, if data sets were drawn from different parent populations).

Array1 and Array2 must have more than two values. The variance of Array1 or Array2 must not be zero.

Example

`@FTEST({75,82,83,85,85,90},{80,86,92,93,95,96})` = 0.637248

See Also

@FDIST

@FINV

@FUTV

Format

```
@FUTV(Intrate, Flows, <[Odd|Periods]>, <Simp>, <Pathdep>, <Filter>,
  <Start>, <End>)
```

Intrate	=	interest rate or block containing interest (discount) rates
Flows	=	block containing cash flows
Odd|Periods	=	delay after last cash flow in number of periods (the default is 0) or block containing lengths of periods between cash flows (the default is 1)
Simp	=	flag specifying how to discount: 0 = compounded discounting (default) 1 = mixed compounded and simple discounting 2 = simple discounting
Pathdep	=	flag specifying whether to apply path-dependent compounding to each flow; 0 = no path (default); 1 = path
Filter	=	flag specifying filter type: 0 = no filter (default); 1 = cashflow < Start; 2 = cashflow ≤ Start; 3 = cashflow > Start; 4 = cashflow ≥ Start; 5 = Start < cashflow < End; 6 = Start ≤ cashflow ≤ End
Start	=	a starting cash flow amount to compare against individual flows
End	=	an ending cash flow amount to compare against individual flows

@FUTV calculates the future value of a given cash flow structure. The future value of a stream of cash flows is the sum of the future values of each cash flow.

By default, @FUTV computes the future value at the time of the last cash flow. If you specify Periods, @FUTV calculates the future value at a time one period after the last cash flow. If you specify Odd, @FUTV calculates the future value at a time Odd periods after the last cash flow.

@FUTV computes future value using this formula:

$$FV = \sum_{i=1}^{n} Fl_i * IF_i$$

where n is the number of cash flows, and is the interest factor associated with the ith cash flow, Flows . is the @FUTV counterpart of the discount factor, , used in @NETPV. Unlike , which reduces the value of a flow, increases the value.

```
FV = Future Value
Fl = Flows
```

Example

Suppose a portfolio has a bond that will make 15 annual interest payments of $1,500, and pay $20,000 in principal along with the last interest payment. If the interest earned on investing the annual interest payments (the reinvestment rate) is 8.5%, this formula calculates the amount in the portfolio at the end of 15 years, using the data shown in the next figure:

`@FUTV(D2,A2..B3)` = $62,348.40

	A	B	C	D	E
1	Cash Flows		Interest Rate		
2	14	$1,500		8.5%	
3	1	$21,500			
4					

See Also
Entering Cash Flow @Functions

@GAMMADIST

Format

```
@GAMMDIST(X, Alpha, Beta, Cum)
```

X	=	value at which to evaluate the function; must be ≥ 0
Alpha	=	parameter to the gamma distribution; must be > 0
Beta	=	parameter to the gamma distribution; must be > 0
Cum	=	1 to return the cumulative gamma distribution function; 0 to return the probability density function

@GAMMDIST returns the gamma distribution function, which is the probability that a random variable will be less than X. Use @GAMMADIST to study random variables characterized by skewed and asymmetric distributions.

When Alpha = 1, @GAMMADIST returns the exponential distribution; see @EXPONDIST.

Examples

`@GAMMADIST(18,8,2,1)` = 0.676103

`@GAMMADIST(18,8,2,0)` = 0.058558

See Also

@GAMMAINV

@GAMMALN

@GAMMAP

@GAMMAQ

@GAMMAINV

Format

```
@GAMMAINV(Prob, Alpha, Beta)
```

Prob = probability associated with the gamma cumulative function; must be ≥ 0 and ≤ 1

Alpha = a parameter to the gamma distribution; must be > 0

Beta = a parameter to the gamma distribution; must be > 0

@GAMMAINV returns the inverse of the cumulative gamma distribution function.

Example

`@GAMMAINV(0.676103,8,2)` = 18

See Also

@GAMMADIST

@GAMMALN

@GAMMAP

@GAMMAQ

@GAMMALN

Format

```
@GAMMALN(X)
```

X = value for which you want to calculate @GAMMALN; must be > 0

Returns the natural logarithm of the gamma function. Use @GAMMALN to build other common statistical functions such as the beta function (see @BETA) and the factorial function (see @FACT).

Example

```
@GAMMALN(6)
```
= 4.787492

See Also

@GAMMADIST
@GAMMAINV
@GAMMAP
@GAMMAQ

@GAMMAP

Format

```
@GAMMAP(A, X)
```

A = parameter to the function; must be > 0

X = value at which to evaluate the function; must be ≥ 0

@GAMMAP returns the incomplete gamma function, also known as the standard cumulative gamma distribution. @GAMMAP is equal to the cumulative gamma distribution when b = 1.

Example

`@GAMMAP(3,4)` = 0.761897

See Also

@GAMMADIST

@GAMMAINV

@GAMMALN

@GAMMAQ

@GAMMAQ

Format

```
@GAMMAQ(A, X)
```

A = parameter to the function; must be > 0

X = value at which to evaluate the function; must be ≥ 0

@GAMMAQ is a complement to the incomplete gamma function and equals (1 - @GAMMAP).

Example

`@GAMMAQ(3,4)` = 0.238103

See Also

@GAMMADIST
@GAMMAINV
@GAMMALN
@GAMMAP

@GCD

Format

```
@GCD(X, Y)
```

X = positive integer to find greatest common divisor of

Y = positive integer to find greatest common divisor of

@GCD returns the greatest common divisor of X and Y (the largest integer that both numbers can be divided by without a remainder; it's also called the greatest common denominator).

Examples

`@GCD(96,78)` = 6

`@GCD(112,42)` = 14

See Also

@LCM

@GEOMEAN

Format

```
@GEOMEAN(List)
```

List = one or more numeric or block values; values in List must be positive

@GEOMEAN returns the geometric mean of a positive range of values. The geometric mean is the nth root of the product of a series of numbers. Use @GEOMEAN when you're interested in an average rate of change of values in a data set given a varying rate of change.

@GEOMEAN uses this formula:

$$\sqrt[n]{x_1 x_2 \ldots x_n}$$

Example

`@GEOMEAN(3,4,5,6,7)` = 4.789389

See Also

@HARMEAN
@MEDIAN
@MODE
@TRIMMEAN
@AVG

@GESTEP

Format

```
@GESTEP(X, <Y>)
```

X = numeric value to check

Y = numeric value that X must exceed for function to return 1 (if omitted, assumed to be 0)

@GESTEP tests whether X is greater than or equal to Y. If it is, @GESTEP returns 1 (true); if not, @GESTEP returns 0 (false).

Examples

`@GESTEP(1,2)` = 0

`@GESTEP(2,1)` = 1

`@GESTEP(1)` = 1

`@GESTEP(-2)` = 0

You can sum several @GESTEP functions to count the number of values that exceed a certain threshold (Y).

See Also

@DELTA

@GETGROUP

Format

```
@GETGROUP(Block, <PageName>)
```

Block	=	a cell or block of the notebook to check
PageName	=	a string value representing a page name or an address specifying the page name to check (optional)

@GETGROUP returns a string that is the name of the group that contains the page specified by Block (unless PageName is used, as discussed next).

If Block is used in conjunction with the optional argument PageName, @GETGROUP searches the notebook referenced by Block for the group that contains the page specified by PageName. PageName is a string or cell address.

If the page isn't part of a group, @GETGROUP returns ERR.

Example

`@GETGROUP([REPORTQ4]A:C12,"April")` searches the notebook REPORTQ4 for the name of the group that contains the page named April

`@GETGROUP([REPORTQ4]A:C12)` searches the notebook REPORTQ4 for the name of the group that contains the page named A

`@GETGROUP([REPORTQ4]A:C12,"Totals:A12")` searches the notebook REPORTQ4 for the name of the group that contains the page named Totals.

See Also

@FIRSTINGROUP

@LASTINGROUP

@HARMEAN

Format

```
@HARMEAN(List)
```

List = one or more numeric or block values; none of the values in List can equal 0

@HARMEAN returns the harmonic mean of a data set. The harmonic mean is the reciprocal of the arithmetic mean of the reciprocals of a set of numbers (see @MEAN).

@HARMEAN uses this formula:

$$\frac{1}{\frac{1}{n}\left(\frac{1}{x_1}+\frac{1}{x_2}+\ldots+\frac{1}{x_n}\right)}$$

Example

`@HARMEAN(3,4,5,6,7)` = 4.575163

See Also

@GEOMEAN
@MEDIAN
@MODE
@TRIMMEAN
@AVG

@HEXTOASC

Format

```
@HEXTOASC(Hex)
```

Hex = hexadecimal number to convert; can be up to 40 hexadecimal digits

@HEXTOASC returns the ASCII equivalent of a hexadecimal number.

Note: If the hexadecimal number includes nonnumeric characters, enclose it in quotation marks.

Examples

```
@HEXTOASC("2B") = +
@HEXTOASC("3031414243444546") = 01ABCDEF
@HEXTOASC("424F524C414E44") = BORLAND
```

See Also

Entering Number Conversion @Functions

@HEXTOBIN

Format

```
@HEXTOBIN(Hex)
```

Hex = hexadecimal number to convert

@HEXTOBIN returns the binary string equivalent of a hexadecimal number. To convert a negative number, precede Hex with a minus sign.

Examples

`@HEXTOBIN("A")` = 1010

`@HEXTOBIN("10")` = 10000

`@HEXTOBIN("1E")` = 11110

See Also

@HEXTOBIN64

Entering Number Conversion @Functions

@HEXTOBIN64

Format

```
@HEXTOBIN64(Hex, <Places>)
```

Hex = hexadecimal number to convert

Places = number of characters to return; must be ≤ 64

@HEXTOBIN64 returns the binary string equivalent of a hexadecimal number (up to 64 bits).

Note: If the hexadecimal number includes nonnumeric characters, enclose it in quotation marks.

Examples

`@HEXTOBIN64("A",2)` = 10

`@HEXTOBIN64("A",6)` = 001010

`@HEXTOBIN64("1E078")` = 00011110000001111000

`@HEXTOBIN64("1E078",7)` = 1111000

See Also

@HEXTOBIN

Entering Number Conversion @Functions

@HEXTONUM64

Format

```
@HEXTONUM64(Hex, <Signed>)
```

Hex = hexadecimal number to convert

Signed = 1 if the most significant bit of Hex is a sign bit; 0 if Hex is positive (the default is 0)

@HEXTONUM64 returns the decimal equivalent of a hexadecimal number (up to 64 bits).

If Signed is 1, the most significant bit of Hex is the sign bit. If the sign bit is 0, the number is positive; if it's 1, the number is negative.

Note: If the hexadecimal number includes nonnumeric characters, enclose it in quotation marks.

Examples

`@HEXTONUM64("A")` = 10

`@HEXTONUM64("123456789ABCDEF0")` = 1311768467463790320

`@HEXTONUM64("FE4FA1",1)` = -110687

See Also

@HEXTONUM

Entering Number Conversion @Functions

@HEXTOOCT

Format

```
@HEXTOOCT(Hex)
```

Hex = hexadecimal number to convert

@HEXTOOCT returns the octal string equivalent of a hexadecimal number. To convert a negative number, precede Hex with a minus sign.

Examples

`@HEXTOOCT("A")` = 12

`@HEXTOOCT("10")` = 20

`@HEXTOOCT("1E")` = 36

See Also

@HEXTOOCT64

Entering Number Conversion @Functions

@HEXTOOCT64

Format

```
@HEXTOOCT64(Hex, <Places>)
```

Hex = hexadecimal number to convert

Places = number of characters to return; must be ≤ 22

@HEXTOOCT64 returns the octal string equivalent of a hexadecimal number (up to 64 bits).

Note: If the hexadecimal number includes nonnumeric characters, enclose it in quotation marks.

Examples

`@HEXTOOCT64("A")` = 12

`@HEXTOOCT64("7",2)` = 07

`@HEXTOOCT64("1E078",6)` = 360170

`@HEXTOOCT64("0123456789ABCDEF")` = 0004432126361152746757

See Also

@HEXTOOCT

Entering Number Conversion @Functions

@HOLS

Format

```
@HOLS(StartDate, EndDate, Holidays, <Saturday>, <Sunday>)
```

StartDate = number representing the start date

EndDate = number representing the end date

Holidays = block containing dates that are holidays; to indicate no holidays, enter an empty cell or block

Saturday = 0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)

Sunday = 0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@HOLS returns the number of holidays between StartDate and EndDate, including the given dates (if they appear in Holidays).

By default, @HOLS doesn't include holidays that fall on a Saturday or Sunday; if either Saturday or Sunday is passed as 1, the count also includes holidays falling on that day.

Example

This formula calculates the number of holidays between April 1, 1993 and December 14, 1993, assuming that the dates contained in block A7..C9 are holidays.

```
@HOLS(@DATE(93,4,1),@DATE(93,12,14),A7..C9)
```
= 5

See Also

@BDAYS

@CDAYS

@HYPGEOMDIST

Format

```
@HYPGEOMDIST(SampleSuccess, SampleSize, PopSuccess, PopSize)
```

SampleSuccess	=	successes in the sample; must be ≥ 0
SampleSize	=	sample size; must be ≥ 0 and ≤ PopSize
PopSuccess	=	successes in the population; must be ≥ 0 and ≤ PopSize
PopSize	=	population size; must be ≥ 0

@HYPGEOMDIST returns the hypergeometric distribution of a sample, which gives the probability of successes in a sample given the sample's size, the total population, and the number of successful trials in that population. Use @HYPGEOMDIST to determine the probability that a distribution contains exactly SampleSuccess items of a particular type.

SampleSuccess must be greater than or equal to 0, greater than the lesser of SampleSize or PopSuccess, and greater than the larger of 0 or (SampleSize-PopSize+PopSuccess)

@HYPGEOMDIST uses this formula:

$$P(X = d) = h(d; n, D, N) = \frac{\binom{D}{d}\binom{N-D}{n-d}}{\binom{N}{n}}$$

where:

```
d = SampleSuccess
n = SampleSize
D = PopSuccess
N = PopSize
```

Examples

Five cards are drawn from a deck of 52 playing cards. This formula calculates the probability that one of the five cards drawn is an ace (assuming there are only four aces in the deck):

`@HYPGEOMDIST(1,5,4,52)` = 0.299474

See Also

@BINOMDIST
@COMB
@FACT
@NEGBINOMDIST
@PERMUT

@IMABS

Format

```
@IMABS(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want the absolute (modulus) value

@IMABS returns the absolute value (modulus) of a complex number with this formula:

$$|C| = \sqrt{x^2 + y^2}$$

```
C = Complex
```

Example

```
@IMABS("-10+25.6j")
```
= 27.48381

@IMAGINARY

Format

```
@IMAGINARY(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy from which you want to extract the imaginary coefficient

@IMAGINARY returns the imaginary coefficient of a complex number.

Examples

```
@IMAGINARY("2+8i")
```
= 8

```
@IMAGINARY("-i")
```
= -1

See Also

@COMPLEX

@IMREAL

@IMARGUMENT

Format

```
@IMARGUMENT(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want to calculate the angle in the complex plane

@IMARGUMENT returns the angle Θ, in radians, of a number in the complex plane, such that

$$x + yi = |x + yi|e^{i\Theta} = |x + yi|(\cos\Theta + i\sin\Theta)$$

Example

```
@IMARGUMENT("5+12i")
```
= 1.176005

@IMCONJUGATE

Format

```
@IMCONJUGATE(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want to calculate the complex conjugate

@IMCONJUGATE returns the complex conjugate of a complex number.

Example

```
@IMCONJUGATE("5+12i")
```
= "5-12i"

@IMCOS

Format

```
@IMCOS(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want to calculate the cosine

@IMCOS returns the cosine of the complex number Complex. @IMCOS uses this formula:

$$\cos(C) = \frac{e^{iC} - e^{-iC}}{2}$$

```
C = Complex
```

Example

```
@IMCOS("5+12i")
```
= "23083.7+78034.8i"

See Also

@COS

@IMSIN

@IMDIV

Format

```
IMDIV(Complex1, Complex2)
```

Complex1 = complex numerator or dividend in the format x + yi, x + iy, x + yj or x + jy

Complex2 = complex denominator or divisor in the format x + yi, x + iy, x + yj or x + jy

@IMDIV returns the quotient of two complex numbers (Complex1 and Complex2) using this formula:

```
Given: Complex1 = a + bi and Complex2 = c + di
```

$$\frac{C1}{C2} = \frac{a + bi}{c + di} = \frac{(ac + bd) + (bc - ad)i}{c^2 + d^2}$$

```
C = Complex
```

Example

```
@IMDIV("5+6i","3+4i")
```
= "1.56-0.08i"

See Also

@IMPOWER

@IMPRODUCT

@IMSUB

@IMEXP

Format

```
@IMEXP(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy for which you want to calculate the exponential

@IMEXP returns the exponential of a complex number using this formula:

Given Complex = *x* + *yi*,

C = Complex

Example

`@IMEXP("5+12i")` = "125.239-79.6345i"

See Also

@IMPOWER
@IMLN
@IMLOG10
@IMLOG2

@IMLN

Format

```
@IMLN(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want to calculate the natural logarithm

@IMLN returns the natural logarithm of the complex number Complex using this formula:

$$\ln(x + yi) = \ln\sqrt{x^2 + y^2} + i\tan^{-1}\left(\frac{y}{x}\right)$$

Example

```
@IMLN("5+12i")
```
= "2.56495+1.17601i"

See Also

@IMLOG2

@IMLOG10

@IMPOWER

@IMLOG10

Format

```
@IMLOG10(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy for which you want to calculate the base 10 log

@IMLOG10 returns the base 10 (common) logarithm of the complex number Complex using this formula:

```
Given Complex = x + yi
```

$$\log 10(C) = \frac{\log(C)}{\log(10)}$$

```
C = Complex
```

Example

```
@IMLOG10("5+12i")
```
= "1.11394+0.510733i"

See Also

@IMLOG2

@IMLN

@IMPOWER

@IMLOG2

Format

```
@IMLOG2(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy for which you want to calculate the base 2 log

@IMLOG2 returns the base 2 logarithm of the complex number Complex using this formula:

```
Given Complex = x + yi
```

$$\log 2(C) = \frac{\log(C)}{\log(2)}$$

```
C = Complex
```

Example

```
@IMLOG2("5+12i")
```
= "3.70044+1.69662i"

See Also

@IMLOG10
@IMLN
@IMPOWER

@IMPOWER

Format

```
@IMPOWER(Complex, Power)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy

Power = the power to which you want to raise Complex; can be a complex number in the format x + yi, x + iy, x + yj, or x + jy

@IMPOWER returns the complex number Complex raised to the power Power. Power can be a value or a complex number.

Examples

`@IMPOWER("5+12i",3)` = "-2035-828i"

`@IMPOWER("5+12i","3+2i")` = "-150.575+145.094i"

See Also

@IMEXP
@IMLN
@IMLOG2
@IMLOG10

@IMPRODUCT

Format

```
@IMPRODUCT(Complex1, Complex2)
```

Complex1 = complex number in the format x + yi, x + iy, x + yj or x + jy

Complex2 = complex number in the format x + yi, x + iy, x + yj or x + jy

@IMPRODUCT returns the product of two complex numbers (Complex1 and Complex2) using this formula:

```
Given Complex1 = a + bi and Complex2 = c + di

(Complex1)(Complex2) = (ac - bd) + (ad + bc)i
```

Example

`@IMPRODUCT("5+12i","2-i")` = "22+19i"

`@IMPRODUCT("10+2i",5)` = "50+10i"

See Also

@IMDIV

@IMPOWER

@IMSUB

@IMSUM

@IMREAL

Format

```
@IMREAL(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy from which you want to extract the real coefficients

@IMREAL returns the real coefficient of a complex number.

Examples

`@IMREAL("2+8i")` = 2

`@IMREAL("-i")` = 0

See Also

@COMPLEX

@IMAGINARY

@IMSIN

Format

```
@IMSIN(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj, or x + jy for which you want to calculate the sine

@IMSIN returns the sine of the complex number Complex using the formula:

```
Given Complex = x + yi,
```

$$\sin(C) = \frac{e^{iC} - e^{-iC}}{2i}$$

```
C = Complex
```

Examples

```
@IMSIN("5+12i")
```
= "-78034.8+23083.7i"

```
@IMSIN("1+i")
```
= "1.29846+0.634964i"

See Also

@IMCOS

@IMSQRT

Format

```
@IMSQRT(Complex)
```

Complex = complex number in the format x + yi, x + iy, x + yj or x + jy to calculate square root of

@IMSQRT returns the square root of a complex number using this formula:

$$\sqrt{C} = \sqrt{|C|}\left(\cos\left(\frac{\arg(C)}{2}\right) + \sin\left(\frac{\arg(C)}{2}\right)\right)$$

```
C = Complex
```

Example

```
@IMSQRT("5+12i")
```
= "3+2i"

@IMSUB

Format

```
@IMSUB(Complex1, Complex2)
```

Complex1 = complex number in the format x + yi, x + iy, x + yj or x + jy from which to subtract Complex2

Complex2 = complex number in the format x + yi, x + iy, x + yj or x + jy to subtract from Complex1

@IMSUB returns the difference of two complex numbers (Complex1 and Complex2) with this formula:

```
Given Complex1 = (a + bi) and Complex2 = (c + di)

(a + bi) - (c + di) = (a - c) + (b - d)i
```

Example

```
@IMSUB("5+12i","2-i")
```
= "3+13i"

See Also

@IMDIV
@IMPOWER
@IMPRODUCT
@IMSUM

@IMSUM

Format

```
@IMSUM(List)
```

List = one or more complex numbers in the format x + yi, x + iy, x + yj or x + jy, separated by commas

@IMSUM returns the sum of a list of complex numbers using this formula:

```
Given Complex1 = (a + bi) and Complex2 = (c + di)

Complex1 + Complex2 = (a + c) + (b + d)i
```

Example

```
@IMSUM("5+12i","7+14i")
```
= "12+26i"

See Also

@IMDIV
@IMPOWER
@IMPRODUCT
@IMSUB

@INDEXTOLETTER

Format

```
@INDEXTOLETTER(Index)
```

Index = an integer number from 0 to 255 inclusive

@INDEXTOLETTER returns a one- or two-character string equivalent (for example, A, B, AA, AB, and IV) for the index number of a page or column.

If Index is < 0 or > 255, @INDEXTOLETTER returns ERR. If Index isn't an integer, it's rounded to the nearest integer.

Examples

`@INDEXTOLETTER(0)` = A

`@INDEXTOLETTER(1)` = B

`@INDEXTOLETTER(255)` = IV

See Also

@LETTERTOINDEX

@INTERCEPT

Format

```
@INTERCEPT(KnownY, KnownX)
```

KnownY = dependent range of values

KnownX = independent range of values

@INTERCEPT returns the y-intercept of the linear regression line through two data sets. KnownY and KnownX must contain the same number of values. The formula @INTERCEPT uses is

$$q = \overline{Y} - a_1\overline{X}$$

, the slope, is calculated using this formula:

$$a_1 = \frac{n\sum xy - (\sum x)(\sum y)}{n\sum x^2 - (\sum x)^2}$$

Example

`@INTERCEPT({16,28,30,35,52,65},{11,15,18,22,35,43})` = 3.56304

See Also

@FORECAST

@PEARSON

@RSQ

@SLOPE

@STEYX

@INTRATE

Format

```
@INTRATE(Settle, Maturity, Investment, Redemption, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Investment	=	amount invested; must be > 0
Redemption	=	redemption value; must be > 0
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@INTRATE returns the simple annualized yield for a fully invested security. @INTRATE computes yield using this formula:

$$Y = \left(\frac{R - I}{I}\right) * \left(\frac{t_b}{M - S}\right)$$

```
Y = yield
R = redemption
I = investment
b = basis
M = maturity
S = settle
```

t_b is the number of days over which the discount rate applies (360 or 365).

Example

This formula calculates the interest rate for a bond with the following terms: Settle is November 11, 1995, Maturity is May 27, 1996, Investment is $10,000, Redemption is $10,397.50, and Calendar is 1 (actual/actual).

```
@INTRATE(@DATE(95,11,11),@DATE(96,5,27),10000,10397.50,1)
```
= 0.073277

@INVB

Format

```
@INVB(Binary, <Bits>)
```

Binary	=	binary number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary; must be ≤ 64

@INVB inverts the bits of a binary number. All bits that are 1 change to 0, and all bits that are 0 change to 1.

Examples

`@INVB(0)` = 1

`@INVB(1010,5)` = 10101

`@INVB(1100,5)` = 10011

See Also

Entering Boolean @Functions

@INVH

Format

```
@INVH(Hex, <Bits>)
```

Hex	=	hexadecimal number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@INVH inverts the binary bits of a hexadecimal number. All bits that are 1 change to 0, and all bits that are 0 change to 1.

Example

`@INVH("A")` = 5

`@INVH("C",8)` = F3

See Also

Entering Boolean @Functions

@ISBDAY

Format

```
@ISBDAY(Date, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing a date
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@ISBDAY tests whether Date is a business day. To qualify as a business day, Date can't fall on a Saturday or Sunday (unless Saturday and Sunday are designated as business days by Saturday and Sunday), and can't appear in the block specified by Holidays. If Date is a business day, @ISBDAY returns 1; otherwise, @ISBDAY returns 0.

Example

Given the block of holidays, A7..C9, and treating Saturday as a business day (except when it's a date included in A7..C9), this formula tests whether May 31, 1993 is a business day:

`@ISBDAY(@DATE(93,5,31),A7..C9,1)` = 0, since May 31, 1993 is Memorial Day.

See Also

Setting Holidays

@ISLEGALPAGENAME

Format

```
@ISLEGALPAGENAME(PageName)
```

PageName = a string value

@ISLEGALPAGENAME returns 1 if PageName is a valid page name (even if the page name doesn't currently exist). Otherwise, it returns 0.

Example

`@ISLEGALPAGENAME("A")` = 1

`@ISLEGALPAGENAME("A Report")` = 0 (name contains a space)

@KURT

Format

```
@KURT(List)
```

List = one or more numeric or block values

@KURT returns the kurtosis of List. The kurtosis of a data set measures a distribution's closeness to normality, indicating relative peakedness or flatness. A kurtosis greater than zero is referred to as leptokurtic. A kurtosis less than zero is referred to as platykurtic.

List must have four or more values. The standard deviation of List must not be 0.

@KURT uses this formula:

$$\left\{\frac{n(n+1)}{(n-1)(n-2)(n-3)}\sum\left(\frac{x_i-\bar{x}}{s}\right)^4\right\}-\frac{3(n-1)^2}{(n-2)(n-3)}$$

where s is the sample standard deviation.

Examples

`@KURT(5,7,9,12,14,15,4,9,5,6)` = -1.11117

`@KURT(9.7,10,9.5,9.3,10.2,10,9.5,11)` = 1.780277

`@KURT(20,25,27,22,35,28)` = 0.876754

See Also

@SKEW

@LARGEST

Format

```
@LARGEST(Array, N)
```

Array	=	a numeric array or a block of values
N	=	number that indicates the rank in size from the data set Array; must be greater than 0 and less than or equal to the number of values in Array

@LARGEST returns the Nth largest number in Array. Use @LARGEST to determine a value's rank in a data set from the top of that set.

Note: If there are duplicates in Array, @LARGEST treats them as separate numbers.

Examples

`@LARGEST({1,2,3,4,5,6,7,8,9,10},2)` = 9

`@LARGEST({1,2,3,4,5,6,7,8,9,10},4)` = 7

`@LARGEST({1,2,3,4,5,6,7,8,9,10},6)` = 5

See Also

@PERCENTILE

@PERCENTRANK

@QUARTILE

@SMALLEST

@LASTBLANKPAGE

Format

`@LASTBLANKPAGE(Block)`

Block = a cell or block reference; can be a link to another opened notebook (for example, [BUDGET]A:A1)

@LASTBLANKPAGE returns a string that contains the letters for the last unnamed blank page in a notebook that isn't part of a group.

Quattro Pro searches for the last unnamed blank page (that isn't in a group) starting at page IV and continuing toward page A. If there are no unnamed blank pages (or if they're all in groups), @LASTBLANKPAGE returns ERR.

Example

`@LASTBLANKPAGE(B17)` = IG (if it is the last page that is blank and unnamed)

See Also

@FIRSTBLANKPAGE

@LASTINGROUP

Format

```
@LASTINGROUP(Block, GroupName)
```

Block = a cell or block of the notebook to check

GroupName = a string value representing a group name

@LASTINGROUP returns a string that contains the letters for the last page in the group named GroupName. @LASTINGROUP searches the notebook referenced by Block for the group. If the group doesn't exist, @LASTINGROUP returns ERR.

Example

`@LASTINGROUP([REPORTQ4]A:C12,"Totals")` = "C" (if the notebook named REPORTQ4 contains a group named Totals that ends with page C)

See Also

@FIRSTINGROUP

@LBDAY

Format

```
@LBDAY(Date, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing a date
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@LBDAY returns the serial date number for the date of the last business day of the month in which Date falls.

Example

This formula calculates the last business day in November 1993, assuming that Sundays and the dates contained in block A7..C9 are holidays.

`@LBDAY(@DATE(93,11,1),A7..C9,1)` = 34303 (November 30, 1993)

See Also

Setting Holidays

@NBDAY

@PBDAY

@LCM

Format

```
@LCM(X, Y)
```

X = positive integer to find least common multiple of

Y = positive integer to find least common multiple of

@LCM returns the least common multiple of X and Y (the smallest positive integer into which both X and Y can divide without leaving a remainder).

Examples

`@LCM(9,6)` = 18

`@LCM(24,12)` = 24

See Also

@GCD

@LETTERTOINDEX

Format

```
@LETTERTOINDEX(Letters)
```

Letters = a one- or two-character string enclosed in quotation marks; column and page letters run in sequence from A to Z, and continue from AA to AZ, up to IV

@LETTERTOINDEX returns the index number (from 0 to 255) for column letters or page letters.

If Letters is a character string outside the range of page and column letters (for example, "IW"), @LETTERINDEX returns ERR.

Examples

`@LETTERTOINDEX("A")` = 0

`@LETTERTOINDEX("B")` = 1

`@LETTERTOINDEX("IV")` = 255

See Also

@INDEXTOLETTER

@LINTERP

Format

```
@LINTERP(KnownX, KnownY, X)
```

KnownX = one-dimensional block containing X values in increasing order

KnownY = one-dimensional block containing Y values corresponding to the X values in KnownX

X = number for which the corresponding Y value is desired

@LINTERP interpolates a Y value corresponding to X using the XY pairs specified by KnownX (which contains the X coordinates) and KnownY (which contains the Y coordinates). If X lies between two values in KnownX, @LINTERP interpolates using those two values and their respective KnownY counterparts. If X is outside the range of KnownX, the Y value is extrapolated based on the slope of the line between the two closest points.

KnownX and KnownY don't have to be the same size. If KnownY is smaller than KnownX, the last value in KnownY is used as the corresponding Y value for extra KnownX values. If KnownY is larger than KnownX, its extra values are ignored.

Example

This formula calculates the Y value for the X value 6.7 if the data in the next figure is used.

`@LINTERP(A3..A9,B3..B9,6.7)` = 17.5976

	A	B	C
1	x values	y values	
2			
3	-28.345	-9.7821	
4	-17.89	-5.6667	
5	0.9232	2.891	
6	1.212	2.9978	
7	4.552	13.67	
8	10.75	25.003	
9	30.8	33.33	
10			

@LOGINV

Format

```
@LOGINV(Prob, Mean, SDev)
```

Prob = probability associated with the cumulative lognormal distribution function; 0 ≤ Prob < 1

Mean = mean of ln(x)

SDev = standard deviation of ln(x); must be > 0

@LOGINV returns the inverse of the cumulative lognormal distribution.

Example

`@LOGINV(0.027985,2.5,0.8)` = 2.640543

See Also

@LOGNORMDIST

@LOGNORMDIST

Format

```
@LOGNORMDIST(X, Mean, SDev)
```

X	=	value to evaluate the function; must be > 0
Mean	=	mean of ln(x)
SDev	=	standard deviation of ln(x); must be > 0

@LOGNORMDIST returns the cumulative lognormal distribution.

Example

`@LOGNORMDIST(3,2.5,0.8)` = 0.03991

See Also

@LOGINV

@LWKDAY

Format

```
@LWKDAY(Wkday, Month, Year, <AuxWkday>)
```

Wkday	=	number from 1 (Saturday) to 7 (Friday)
Month	=	number from 1 (January) to 12 (December)
Year	=	number from 0 (1900) to 199 (2099) or a standard year like 1993
AuxWkday	=	auxiliary day of the week that must fall in the same week as Wkday; 0 for no auxiliary day or a number from 1 (Saturday) to 7 (Friday) indicating the auxiliary day (the default is 0)

@LWKDAY returns the serial date number for the date of the last occurrence of Wkday in Month of Year (for example, the last Tuesday in November 1994).

You can use AuxWkday to specify that both Wkday and AuxWkday must fall in the same week of the same month. (See the second example.)

Examples

`@LWKDAY(3,6,115)` = 42184 (June 29, 2015), the date of the last Monday in June 2015.

`@LWKDAY(4,11,94,7)` = 34660 (November 22, 1994), the last Tuesday on which both the last Tuesday and a Friday fall on the same week of November 1994.

See Also

@LBDAY

@EMNTH

@MDAYS

Format

```
@MDAYS(Month, Year)
```

Month = number from 1 (January) to 12 (December)

Year = number from 0 (1900) to 199 (2099) or a standard year like 1993

@MDAYS returns the number of calendar days in Month of Year.

Example

`@MDAYS(2,1996)` = 29, the number of days in February 1996.

See Also

@BDAYS
@HOLS
@CDAYS
@YDAYS

@MDET

Format

```
@MDET(Array)
```

Array = a numeric array or a block of values specifying a square matrix; must have an equal number of rows and columns, and cannot contain blank cells

@MDET calculates the determinant of a matrix (Array). The determinant is obtained by taking any row or column of the matrix, forming the products of each element and its cofactor, and taking the sum of the products; @MDET uses this formula:

where is the element in the ith row and jth column of A and the cofactor

is the product of the determinant of the minor matrix

, formed by deleting row i and column j of A, and a power of -1:

$$A_{ij} = (-1)^{i+j}\left|M_{ij}\right|$$

If Array does not contain the same number of rows and columns, or if Array contains any blank cells, @MDET returns ERR. If any two rows or columns in Array are equal or have proportional elements, @MDET returns 0.

Examples

```
@MDET({12,15,21|8,13,17|16,32,44})
```
= 144

This formula calculates the determinant of the data shown in the next figure:

```
@MDET(C3..F6)
```
= -2869.95

	C	D	E	F	G
3	2.908	-2.253	6.775	3.97	
4	1.212	1.995	2.266	8.008	
5	4.552	5.681	8.85	1.302	
6	5.809	-5.03	0.099	7.832	
7					

@MDURATION

Format

```
@MDURATION(Settle, Maturity, Coupon, Yield, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Coupon	=	coupon rate; 0 ≤ Coupon ≤ 1
Yield	=	annual yield; 0 < Yield ≤ 1
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@MDURATION returns the modified Macaulay duration for a bond with assumed par value of 100. Modified duration is calculated using this formula:

$$\text{modified duration} = \frac{D}{1 + \frac{Y}{F}}$$

D = Duration
Y = Yield
F = Frequency

Example

This formula calculates the modified duration of a bond with these terms: Settle is August 8, 1992, Maturity is November 15, 1998, Coupon is 9%, and Yield is 8.816%.

```
@MDURATION(@DATE(92,8,8),@DATE(98,11,15),0.09,0.08816)
```
= 4.631923

See Also

@DURATION

@MEDIAN

Format

```
@MEDIAN(List)
```

List = one or more numeric or block values

@MEDIAN returns the middle value in a range of values in a data set arranged in ascending or descending order. If the number of values in the data set is even, the median is the mean of the two middle values. Use @MEDIAN when you want a more robust estimation of the central value in a distribution than you obtain with @AVG.

Examples

`@MEDIAN(10,12,15,25,30)` = 15

`@MEDIAN(2,4,5,5,6,8,9,9)` = 5.5

See Also

@AVG
@COUNT
@DAVG
@MODE
@SUM

@MNTHS

Format

```
@MNTHS(StartDate, EndDate, <EndMnth>)
```

StartDate = number representing the start date

EndDate = number representing the end date

EndMnth = 1 to indicate adherence to ends of months; 0 to indicate that ends of months are ignored; the default is 1

@MNTHS calculates the number of whole months between StartDate and EndDate. A whole month is the day of the month on which a given date falls to that same day in the next month, such as March 11 to April 11.

If the day of the month on which StartDate falls doesn't exist in the month in which EndDate falls, and EndDate falls on the last day of that month, the number of months returned includes that month. For example, the number of months returned for March 31, 1993 to June 30, 1993 is 3 and not 2.

If StartDate falls on the last day of a month with less than 31 days, and EndDate precedes StartDate, the result depends on the value of EndMnth; for example, when evaluating the period from February 29, 1992, to January 31, 1992, if EndMnth is 1, -1 is returned; if EndMnth is 0, 0 is returned.

Examples

`@MNTHS(@DATE(93,4,9),@DATE(94,9,15))` = 17, the number of whole months between April 9, 1993 and September 15, 1994.

`@MNTHS(@DATE(93,4,30),@DATE(93,1,31))` = -3

See Also

@BDAYS
@HOLS
@CDAYS
@MDAYS
@YDAYS

@MODE

Format

```
@MODE(List)
```

List = one or more numeric or block values

The mode is a measure of central tendency and represents the value in a sample or population that appears more frequently than any other value. The mode emphasizes data concentration and is best used to describe large data sets. It is commonly used to decide which resulting value is correct when the same measuring or computing process is repeated several times.

If the data set contains no duplicate data points, @MODE returns NA.

Examples

`@MODE(2,2,5,7,9,9,9,10,10,11,12,18)` = 9

`@MODE(91,87,83,80,86,55,83,68,79,83)` = 83

`@MODE(1,2,3,4,5)` = NA

See Also

@MEDIAN

@AVG

@MROUND

Format

```
@MROUND(X, Y)
```

X = value to round

Y = value to make rounded X divisible by

@MROUND rounds X to the nearest value that's evenly divisible by Y. If Y is zero, @MROUND returns zero. If X and Y do not have the same sign, @MROUND returns ERR.

Examples

`@MROUND(2.36,0.25)` = 2.25

`@MROUND(2.47,0.25)` = 2.5

See Also

@CEILING
@EVEN
@FLOOR
@ODD
@INT
@ROUND

@MTGACC

Format

```
@MTGACC(Int, TtlPer, Principal, Residual, ExtraPrin, <Fper>, <Lper>,
  <Rper>, <Option>)
```

Int	=	number ≥ 0 representing the periodic interest rate
TtlPer	=	total periods in the loan from start to finish, or the total periods remaining from the chosen starting period forward
Principal	=	original loan balance; also can be any starting point in the loan
Residual	=	remaining balance on loan at end of loan term; enter 0 if the loan will be paid in full
ExtraPrin	=	extra principal amount to be paid each period (must be positive)
Fper	=	number of the first period, relative to the starting point, in which extra principal is paid; the default is 1 (the first period)
Lper	=	number of the last period, relative to the starting point, in which extra principal is paid; the default is until the end of the loan; you can set Lper to any number greater than or equal to the last period number when extra principal payments last the life of the loan (for example, Lper can be 400 for a loan which lasts 360 periods)
Rper	=	period for which the loan status is reported; the default is at loan end (any number greater than the end of the loan defaults to loan end); Rper does not affect the value @MTGACC returns if Option is 0 or 10
Option	=	specifies the output value type (the default is 0):

0 = number of periods to loan end, when balance equals Residual

1 = balance of loan at the Rper

2 = cumulative interest paid at Rper

3 = cumulative principal paid at Rper

10 = number of fewer periods in loan life, due to payment of extra principal

11 = balance reduction at Rper due to payment of extra principal

12 = reduction in cumulative interest paid at Rper due to payment of extra principal

13 = increase in cumulative principal paid at Rper due to payment of extra principal

@MTGACC calculates the effects of paying extra monthly principal for amortized loans. The value that @MTGACC returns depends on the Option you specify. For the specified Rper, you can find the loan balance, cumulative interest, or cumulative principal. You can also find how the number of periods, loan balance, and cumulative interest have been reduced, and how cumulative principal has increased, as a result of paying extra principal.

@MTGACC returns values for the end of the loan or for the end of any payment period. During calculation, @MTGACC rounds currency values to 2 decimal places, giving currency answers in whole cents, as is common with mortgage institutions.

You can use @MTGACC to return information about a loan on which you make no extra principal payments (ExtraPrin = 0); for example, @MTGACC returns exact values of the loan balance, cumulative interest paid, and cumulative principal paid.

Examples

For a mortgage with a yearly interest rate of 9%, monthly payments for 30 years, an original balance of $150,000, and a future value of $80,000, the following formulas calculate the effects of paying $300

extra principal per month starting at the beginning of the second year and continuing through the fourth year. The reporting period is the tenth year (when the home will be sold and the mortgage paid-off).

Number of periods to loan end:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,0)` = 357

Balance at Rper:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,1)` = $128,635.26

Cumulative interest paid at Rper:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,2)` = $125,724.06

Cumulative principal paid at Rper:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,3)` = $21,364.74

Number of fewer periods in loan life:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,10)` = 3

Reduction in balance at Rper due to ExtraPrin:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,11)` = $13,964.70

Reduction in cumulative interest paid at Rper due to ExtraPrin:

`@MTGACC(.09/12,30*12,150000,80000,300,2*12,4*12,10*12,12)` = $6,464.70

@MULT

Format

`@MULT(List)`

List = one or more numbers or blocks of numbers, separated by commas

@MULT calculates the cumulative product of a set of numbers (List). List is a comma separated list of numbers, blocks containing numbers, or both. The numbers in this list are multiplied by each other. Blank cells and cells containing strings in any of the blocks passed as arguments are ignored.

Example

This formula calculates the cumulative product 3.4, 5.7, -1.2, and the numbers shown in the next figure.

`@MULT(A6..C10,3.4,5.7,-1.2)` = 30037.32

	A	B	C	D
6	3.467	0.123		
7	134.23	0.034	1.238	
8	87.65	6.54%	0,987	
9	-2.35		79.11	
10	101.93		0.005	
11				

@MULTINOMIAL

Format

`@MULTINOMIAL(List)`

List = one or more numbers to calculate multinomial of; each number in List must be ≥ 0

@MULTINOMIAL returns the multinomial of the values in List using this formula for @MULTINOMIAL(a,b,c):

$$\frac{(a+b+c)!}{a!b!c!}$$

If any value in List is negative, @MULTINOMIAL returns ERR.

Example

`@MULTINOMIAL(3,4,5)` = 27720

@NBDAY

Format

```
@NBDAY(Date, <Holidays>, <Saturday>, <Sunday>)
```

Date	=	number representing a date
Holidays	=	block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)
Saturday	=	0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)
Sunday	=	0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@NBDAY returns the serial date number of the next business day after Date.

Examples

`@NBDAY(@DATE(93,2,26))` = 34029 (March 1, 1993)

`@NBDAY(@DATE(93,12,24),A7..C9,0,1)` = 34329 (December 26, 1993), assuming that Saturdays and the dates in block A7..C9 are holidays.

See Also

Setting Holidays

@LBDAY

@PBDAY

@NEGBINOMDIST

Format

```
@NEGBINOMDIST(Failures, Successes, Prob)
```

Failures	=	number of failures
Successes	=	threshold of successes
Prob	=	probability of a success; $0 \leq$ Prob ≤ 1

@NEGBINOMDIST returns the negative binomial distribution. Use @NEGBINOMDIST to determine the distribution of the number of failures you experience before achieving a given number of successes.

Example

A polling organization asks a sampling of voters if they favor Candidate A for reelection. Given that 55% of the city's voters favor Candidate A, this formula calculates the probability that the polling organization will contact 10 voters who do not favor her for reelection before contacting 1 voter who does favor her:

`@NEGBINOMDIST(10,1,0.55)` = 0.000187

@NETPV

Format

```
@NETPV(Discrate, Flows, <Initial>, <[Odd|Periods]>, <Simp>, <Pathdep>,
  <Filter>, <Start>, <End>)
```

Discrate	=	discount rate or block containing discount rates corresponding to block of cash flows
Flows	=	block containing cash flows
Initial	=	initial cash flow (the default is 0)
Odd|Periods	=	delay between initial and first cash flow in number of periods (the default is 1) or block containing lengths of periods between cash flows (the default is 1)
Simp	=	flag specifying how to discount: 0 = compounded discounting (default) 1 = mixed compounded and simple discounting 2 = simple discounting
Pathdep	=	flag specifying whether to apply path-dependent compounding to each flow; 0 = no path (default); 1 = path
Filter	=	flag specifying filter type: 0 = no filter (default); 1 = cashflow < Start; 2 = cashflow ≤ Start; 3 = cashflow > Start; 4 = cashflow ≥ Start; 5 = Start < cashflow < End; 6 = Start ≤ cashflow ≤ End
Start	=	a starting cash flow amount to compare against individual flows
End	=	an ending cash flow amount to compare against individual flows

@NETPV computes the net present value of a stream of cash flows.

Example

A firm is considering the purchase of two machines. Machine A requires an initial outlay of $40,000 and produces a cash flow of $23,000 for three years. Machine B requires an outlay of $50,000 and produces a cash flow of $22,000 for four years. The following formulas calculate the net present values of both machines, using the data shown in the next figure and a discount rate of 12%. The results are useful for determining which machine is a better purchase:

Machine A: `@NETPV(0.12,A13..B14,B12)` = $15,242.12

Machine B: `@NETPV(0.12,C13..D14,D12)` = $16,821.69

	A	B	C	D	E
11	Machine A Income		Machine B Income		
12	Initial	($40,000)	Initial	($50,000)	
13	3	$23,000	4	$22,000	
14					

The net present value of the flows associated with Machine B is greater, so it should be purchased.

See Also

Entering Cash Flow @Functions

@NORMDIST

Format

```
@NORMDIST(X, Mean, SDev, Cum)
```

X	=	value at which to evaluate function
Mean	=	mean of the normal distribution
SDev	=	standard deviation of the normal distribution; must be > 0
Cum	=	1 to return the cumulative normal distribution function; 0 (the default) to return the probability density function

@NORMDIST computes the normal distribution function. A normal distribution is one that is perfectly symmetrical about its mean, and its spread is determined by the value of the standard deviation. The normal distribution describes many statistical phenomena, including the distribution of population means.

@NORMDIST uses this formula to calculate the cumulative normal distribution function:

$$\int_{-\infty}^{x} f(t)dt, f(t) = \frac{e^{-(t-\mu)/2\sigma^2}}{\sqrt{2\pi\sigma}}$$

To calculate the probability mass function for a normal distribution, @NORMDIST uses this formula:

$$f(t) = \frac{e^{-(t-\mu)/2\sigma^2}}{\sqrt{2\pi\sigma}}$$

Examples

`@NORMDIST(50,48,1.2,1)` = 0.95221

`@NORMDIST(50,48,1.2,0)` = 0.082898

See Also

@NORMINV

@NORMSDIST

@NORMSINV

@STANDARDIZE

@NORMINV

Format

```
@NORMINV(Prob, Mean, SDev)
```

Prob = probability corresponding to the normal distribution; 0 < Prob < 1

Mean = mean of the normal distribution

SDev = standard deviation of the normal distribution; must be > 0

@NORMINV returns the inverse of the cumulative normal distribution function.

Example

`@NORMINV(0.95221,48,1.2)` = 50

See Also

@NORMDIST

@NORMSDIST

@NORMSINV

@STANDARDIZE

@NORMSDIST

Format

```
@NORMSDIST(X)
```

X = value at which to evaluate the function

@NORMSDIST returns the standard normal cumulative distribution function. The standard normal cumulative distribution function has a mean of 0 and a standard deviation of 1.

@NORMSDIST uses this formula:

$$\int_{-\infty}^{x} f(t)dt, f(t) = \frac{e^{-t^2} / 2}{\sqrt{2\pi}}$$

Example

```
@NORMSDIST(1.66667)
```
= 0.95221

See Also

@NORMDIST
@NORMINV
@NORMSINV
@STANDARDIZE

@NORMSINV

Format

```
@NORMSINV(Prob)
```

Prob = probability corresponding to the normal distribution; must be > 0 and < 1

@NORMSINV returns the inverse of the standard normal cumulative distribution function. The standard normal cumulative distribution function has a mean of 0 and a standard deviation of 1.

Example

`@NORMSINV(0.95221)` = 1.66667

See Also

@NORMDIST
@NORMINV
@NORMSDIST
@STANDARDIZE

@NUMTOBIN

Format

```
@NUMTOBIN(Decimal)
```

Decimal = decimal number to convert

@NUMTOBIN returns the binary string equivalent of a decimal number. To convert a negative number, precede Decimal with a minus sign.

Examples

`@NUMTOBIN(10)` = 1010

`@NUMTOBIN(16)` = 10000

`@NUMTOBIN(30)` = 11110

See Also

@NUMTOBIN64

Entering Number Conversion @Functions

@NUMTOBIN64

Format

```
@NUMTOBIN64(Decimal, <Places>)
```

Decimal = decimal number to convert

Places = number of characters to return; must be ≤ 64

@NUMTOBIN64 returns the binary string equivalent of a decimal number (up to 64 bits).

Examples

`@NUMTOBIN64(10)` = 1010

`@NUMTOBIN64(10,5)` = 01010

`@NUMTOBIN64(123000)` = 11110000001111000

`@NUMTOBIN64(123000,7)` = 1111000

See Also

@NUMTOBIN

Entering Number Conversion @Functions

@NUMTOHEX64

Format

```
@NUMTOHEX64(Decimal, <Places>)
```

Decimal = decimal number to convert

Places = number of characters to return; must be ≤ 16

@NUMTOHEX64 returns the hexadecimal string equivalent of a decimal number (up to 64 bits).

Examples

`@NUMTOHEX64(10)` = A

`@NUMTOHEX64(10,2)` = 0A

`@NUMTOHEX64(123000)` = 1E078

`@NUMTOHEX64(1000000000)` = 3B9ACA00

`@NUMTOHEX64(1311768467463790340)` = 123456789ABCDF04

See Also

Entering Number Conversion @Functions

@NUMTOOCT

Format

```
@NUMTOOCT(Decimal)
```

Decimal = decimal number to convert

@NUMTOOCT returns the octal string equivalent of a decimal number. To convert a negative number, precede Decimal with a minus sign.

Examples

`@NUMTOOCT(10)` = 12

`@NUMTOOCT(16)` = 20

`@NUMTOOCT(30)` = 36

See Also

@NUMTOOCT64

Entering Number Conversion @Functions

@NUMTOOCT64

Format

```
@NUMTOOCT64(Decimal, <Places>)
```

Decimal = decimal number to convert

Places = number of characters to return; must be ≤ 22

@NUMTOOCT64 returns the octal string equivalent of a decimal number (up to 64 bits).

Examples

`@NUMTOOCT64(8)` = 10

`@NUMTOOCT64(10,3)` = 012

`@NUMTOOCT64(123000)` = 360170

`@NUMTOOCT64(123000,3)` = 170

`@NUMTOOCT64(2^63)` = 10000000000000000002

See Also

@NUMTOOCT

Entering Number Conversion @Functions

@NWKDAY

Format

```
@NWKDAY(N, Wkday, Month, Year, <AuxWkday>)
```

N	=	number from 1 to 5
Wkday	=	number from 1 (Saturday) to 7 (Friday)
Month	=	number from 1 (January) to 12 (December)
Year	=	number from 0 (1900) to 199 (2099) or a standard year like 1993
AuxWkday	=	auxiliary day of the week that must fall in the same week as Wkday; 0 for no auxiliary day or a number from 1 (Saturday) to 7 (Friday) indicating the auxiliary day (the default is 0)

@NWKDAY returns the serial date number for the date of the Nth occurrence of Wkday in Month. If there isn't an Nth occurrence, @NWKDAY returns ERR.

You can use AuxWkday to specify another day that must fall in the same week and month as Wkdday; see the second example.

Examples

`@NWKDAY(2,3,4,99)` = 36262 (April 12, 1999), the date of the second Monday in April 1999.

`@NWKDAY(1,7,12,93,3)` = 34313 (December 10, 1993), the first Friday on which both the first Friday and a Monday fall in the same week of December 1993.

See Also

@LWKDAY

@WKDAY

@OCTTOBIN

Format

```
@OCTTOBIN(Oct)
```

Oct = octal number to convert; denote negative numbers using a minus sign

@OCTTOBIN returns the binary string equivalent of an octal number.

Examples

`@OCTTOBIN("12")` = 1010

`@OCTTOBIN("20")` = 10000

`@OCTTOBIN("36")` = 11110

See Also

Entering Number Conversion @Functions

@OCTTOHEX

Format

```
@OCTTOHEX(Oct)
```

Oct = octal number to convert; denote negative numbers using a minus sign

@OCTTOHEX returns the hexadecimal string equivalent of an octal number.

Examples

```
@OCTTOHEX("12")
```
= A

```
@OCTTOHEX("20")
```
= 10

```
@OCTTOHEX("36")
```
= 1E

See Also

Entering Number Conversion @Functions

@OCTTONUM

Format

```
@OCTTONUM(Oct)
```

Oct = octal number to convert; denote negative numbers using a minus sign

@OCTTONUM returns the decimal equivalent of an octal number.

Examples

```
@OCTTONUM("12")
```
= 10

```
@OCTTONUM("20")
```
= 16

```
@OCTTONUM("36")
```
= 30

See Also

Entering Number Conversion @Functions

@ODD

Format

```
@ODD(X)
```

X = value to round

@ODD rounds X up (away from zero) to the nearest odd integer. If X is already an odd integer, @ODD returns X.

Examples

`@ODD(3.2)` = 5

`@ODD(3)` = 3

`@ODD(-3.2)` = -5

See Also

@CEILING
@EVEN
@FLOOR
@MROUND
@INT
@ROUND

@ODDFPRICE

Format

```
@ODDFPRICE(Settle, Maturity, Issue, FirstCpn, Coupon, Yield, <Redemption>,
  <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date
Maturity	=	number representing the maturity date
Issue	=	number representing the issue date
FirstCpn	=	number representing the first coupon date
Coupon	=	coupon rate; must be ≥ 0
Yield	=	annual yield; $0 < \text{Yield} \leq 1$
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ODDFPRICE returns the price per 100 face value of a bond having an odd (short or long) first period. In an odd first coupon period, the first coupon payment is a prorated multiple of a normal coupon payment.

Dates for @ODDFPRICE must follow this pattern:

Issue < Settle < Maturity

Issue < FirstCpn < Maturity

Example

This formula returns the price per 100 face value of a bond with the following terms: Settle is March 15, 1993, Maturity is is November 15, 1995, Issue is January 4, 1992, FirstCpn is May 15, 1993, Coupon is 8.5%, Yield is 8.7%, Redemption is 100, Freq 2, and Calendar is 0 (30/360).

```
@ODDFPRICE(@DATE(93,3,15),@DATE(95,11,15),@DATE(92,1,4),@DATE(93,5,15),0.08
5, 0.087,100,2,0)
```
= 99.40933

@ODDFYIELD

Format

```
@ODDFYIELD(Settle, Maturity, Issue, FirstCpn, Coupon, Price, <Redemption>,
  <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date
Maturity	=	number representing the maturity date
Issue	=	number representing the issue date
FirstCpn	=	number representing the first coupon date
Coupon	=	coupon rate; must be ≥ 0
Price	=	price of the security; must be > 0
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ODDFYIELD returns the yield of a security having an odd (short or long) first period. In an odd first coupon period, the first coupon payment is a prorated multiple of a normal coupon payment.

Dates for @ODDFYIELD must follow this pattern:

Issue < Settle < Maturity

Issue < FirstCpn < Maturity

Example

This formula calculates the yield for a security with the following terms: Settle is March 15, 1993, Maturity is November 15, 1995, Issue is January 4, 1992, FirstCpn is May 15, 1993, Coupon is 8.5%, Price is 100, Redemption is 100, Freq is 2, and Calendar is 0 (30/360).

```
@ODDFYIELD(@DATE(93,3,15),@DATE(95,11,15),@DATE(92,1,4),@DATE(93,5,15),0.08
5, 100,100,2,0)
```
= 0.084487

@ODDLPRICE

Format

```
@ODDLPRICE(Settle, Maturity, LastCpn, Coupon, Yield, <Redemption>, <Freq>,
  <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
LastCpn	=	number representing the last coupon date; must be < Maturity
Coupon	=	coupon rate; must be ≥ 0
Yield	=	annual yield; $0 <$ Yield ≤ 1
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ODDLPRICE returns the price per 100 face value of a bond having an odd (short or long) last period. In an odd last coupon period, the last coupon payment is a prorated multiple of a normal coupon payment.

Example

This formula calculates the price per 100 face value of a bond with the following terms: Settle is June 1, 1992, Maturity is December 15, 2012, LastCpn is September 15, 2012, Coupon is 7.5%, and Yield is 5.25%.

```
@ODDLPRICE(@DATE(92,6,1),@DATE(112,12,15),@DATE(112,9,15), 0.075,0.0525) =
```
128.0663

@ODDLYIELD

Format

```
@ODDLYIELD(Settle, Maturity, LastCpn, Coupon, Price, <Redemption>, <Freq>,
  <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
LastCpn	=	number representing the last coupon date; must be < Maturity
Coupon	=	coupon rate; must be ≥ 0
Price	=	price; must be > 0
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@ODDLYIELD returns the yield of a bond having an odd (short or long) last period. In an odd last coupon period, the last coupon payment is a prorated multiple of a normal coupon payment.

Example

This formula calculates the yield of a bond with the following terms: Settle is June 1, 1992, Maturity is December 15, 2012, LastCpn is September 15, 2012, Coupon is 7.5%, and Price is 128.0663.

```
@ODDLYIELD(@DATE(92,6,1),@DATE(112,12,15),@DATE(112,9,15),0.075, 128.0663)
```

= 0.0525

@ORB

Format

@ORB(Binary1, <Binary2>, <Bits>)

Binary1	=	first binary number
Binary2	=	second binary number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be ≤ 64

@ORB performs a bit-by-bit logical OR of each bit in Binary1 and Binary2. Any bit that is set to 1 in either Binary1 or Binary2 causes the resulting output bit to be set to 1.

If only one number is given, then @ORB performs an any-ones test, or OR reduction, on Binary1; @ORB returns 1 if any bits in Binary1 are set to 1; otherwise, it returns 0.

Examples

`@ORB(10,1)` = 11

`@ORB(10,10)` = 10

`@ORB(10)` = 1

`@ORB(1100,1,5)` = 01101

See Also

Entering Boolean @Functions

@ORH

Format

```
@ORH(Hex1, <Hex2>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@ORH performs a bit-by-bit logical OR of each bit in Hex1 and Hex2. Any binary bit that is set to 1 in either Hex1 or Hex2 causes the resulting output bit to be set to 1.

If only one number is given, then @ORH performs an any-ones test, or OR reduction, on Hex1; @ORH returns 1 if any bits in Hex1 are set to 1; otherwise, it returns 0.

Examples

`@ORH("A","F")` = F

`@ORH("A")` = 1

`@ORH("C","1",8)` = 0D

See Also

Entering Boolean @Functions

@PAGEINDEX

Format

```
@PAGEINDEX(Name)
```

Name = a string corresponding to the name of a page; must be enclosed in quotation marks

@PAGEINDEX returns the index number (from 0 to 255) for a specified page name. If no page names match the string Name, @PAGEINDEX returns ERR.

Note: To return the index number for a page in another notebook, use @PAGEINDEX2.

Example

`@PAGEINDEX("EXPENSES")` = 0 (the index number for the page named EXPENSES is 0)

@PAGEINDEX2

Format

```
@PAGEINDEX2(NotebookLink, Name)
```

NotebookLink = a reference to a page, cell, or block in another notebook (for example, [BUDGET]A:A1)

Name = a string corresponding to the name of a page; must be enclosed in quotation marks

@PAGEINDEX2 returns the index number (from 0 to 255) for a specified page name in a notebook specified by NotebookLink. If no page names match the string Name, @PAGEINDEX2 returns ERR.

Example

`@PAGEINDEX2([BUDGET]A:A1,"EXPENSES")` = 0 (the index number for the page named EXPENSES in notebook BUDGET is 0)

See Also

@PAGEINDEX

@PAGENAME

Format

```
@PAGENAME(Index)
```

Index = a number from 0 to 255 inclusive

@PAGENAME returns the name of a page specified by Index. If there isn't a name for the specified page, @PAGENAME returns ERR.

Note: To return a page name from another notebook, use @PAGENAME2.

Example

`@PAGENAME(3)` = EXPENSES (the page with index number 3 is named EXPENSES)

See Also

@PAGENAMES

@PAGENAMES2

@PAGENAME2

Format

```
@PAGENAME2(NotebookLink, Index)
```

NotebookLink = a reference to a page, cell, or block in another notebook (for example, [BUDGET]A:A1)

Index = a number from 0 to 255 inclusive

@PAGENAME2 returns the name of a page specified by Index in a notebook specified by NotebookLink. If there isn't a name for the specified page, @PAGENAME2 returns ERR.

Example

`@PAGENAME2([BUDGET]A:A1,3)` = EXPENSES (the page with index number 3 in notebook BUDGET is named EXPENSES)

See Also

@PAGENAME

@PAGENAMES

@PAGENAMES2

@PAGENAMES

Format
`@PAGENAMES`

@PAGENAMES returns a two-column table showing the page letters and corresponding page names for the active notebook. The left column of the table contains page letters (from A to IV), and the right column contains corresponding page names.

Because @PAGENAMES returns an array, it is automatically enclosed within an @ARRAY @function.

If the active notebook doesn't have any named pages, @PAGENAMES returns ERR.

Caution: Make sure there is enough room for a two-column table, with one row for each page name. Quattro Pro overwrites existing data in cells it uses for the table.

Note: To return page names for another notebook, use @PAGENAMES2.

Example
The active notebook consists of five named pages, Qtr1, Qtr2, Qtr3, Qtr4, and Totals, whose page letters are A, B, C, D, and E, respectively.

`@ARRAY(@PAGENAMES)` = table in A1..B5 shown in the next figure

	A	B
1	A	Qtr1
2	B	Qtr2
3	C	Qtr3
4	D	Qtr4
5	E	Totals

See Also
@ARRAY
@PAGENAME
@PAGENAME2

@PAGENAMES2

Format

```
@PAGENAMES2(NotebookLink)
```

NotebookLink = a reference to a page, cell, or block in another notebook (for example, [BUDGET]A:A1)

@PAGENAMES2 returns a two-column table showing the page letters and corresponding page names for the notebook specified by NotebookLink. The left column of the table contains page letters (from A to IV), and the right column contains corresponding page names.

Because @PAGENAMES2 returns an array, it is automatically enclosed within an @ARRAY @function.

If the active notebook doesn't have any named pages, @PAGENAMES2 returns ERR.

Caution: Make sure there is enough room for a two-column table, with one row for each page name. Quattro Pro overwrites existing data in cells it uses for the table.

Example

A notebook named BUDGET consists of five named pages, Qtr1, Qtr2, Qtr3, Qtr4, and Totals, whose page letters are A, B, C, D, and E, respectively.

`@ARRAY(@PAGENAMES2([BUDGET]F:A1))` = table in A1..B5 shown in the next figure

	A	B
1	A	Qtr1
2	B	Qtr2
3	C	Qtr3
4	D	Qtr4
5	E	Totals

See Also

@ARRAY
@PAGENAME
@PAGENAME2
@PAGENAMES

@PBDAY

Format

```
@PBDAY(Date, <Holidays>, <Saturday>, <Sunday>)
```

Date = number representing a date

Holidays = block containing dates that are holidays or the date of a single holiday or 0 to indicate no holidays (the default is 0)

Saturday = 0 to specify that Saturday is not a business day; 1 to specify that Saturday is a business day (the default is 0)

Sunday = 0 to specify that Sunday is not a business day; 1 to specify that Sunday is a business day (the default is 0)

@PBDAY returns the serial date number of the first business day before Date.

Examples

`@PBDAY(@DATE(93,3,1))` = 34026 (February 26, 1993)

`@PBDAY(@DATE(93,12,26),A7..C9,0,1)` = 34325 (December 22, 1993), assuming that Saturdays and the dates in the block A7..C9 are holidays.

See Also

Setting Holidays

@LBDAY

@NBDAY

@PEARSON

Format

```
@PEARSON(Array1, Array2)
```

Array1 = array of independent values

Array2 = array of dependent values

@PEARSON returns the Pearson product moment correlation coefficient, which measures the linear association of two data sets. Array1 and Array2 must have the same number of values. @PEARSON uses this formula:

$$r = \frac{n(\Sigma XY) - (\Sigma X)(\Sigma Y)}{\sqrt{\left[n\Sigma X^2 - (\Sigma X)^2\right] - \left[n\Sigma Y^2 - (\Sigma Y)^2\right]}}$$

A value of r near or equal to 0 implies little or no linear relationship exists between the two lists of numbers. A value of r near or equal to 1 or -1 indicates a very strong linear relationship.

Example

This example refers to cells in the next figure.

`@PEARSON(B2..B16,C2..C16)` = 0.989324

	A	B	C
1	Date	Advertising	Sales
2	04/30/93	$435	$7,000
3	05/07/93	$400	$6,000
4	05/14/93	$505	$7,767
5	05/21/93	$470	$7,800
6	05/28/93	$610	$9,534
7	06/04/93	$540	$7,750
8	06/11/93	$575	$8,945
9	06/18/93	$715	$11,301
10	06/25/93	$645	$9,465
11	07/02/93	$680	$10,760
12	07/09/93	$785	$13,000
13	07/16/93	$750	$11,890
14	07/23/93	$855	$12,980
15	07/30/93	$820	$13,068
16	08/06/93	$890	$14,246

See Also

@INTERCEPT

@RSQ

@SLOPE

@STEC

@STEYX

@PERCENTILE

Format

```
@PERCENTILE(Array, X)
```

Array = a numeric array or a block of values

X = a percentile value between 0 and 1, inclusive

@PERCENTILE returns a number from Array at the percentile indicated by X.

Examples

`@PERCENTILE({4,5,7,9,10,12,13,16},0)` = 4

`@PERCENTILE({4,5,7,9,10,12,13,16},0.25)` = 6.5

`@PERCENTILE({4,5,7,9,10,12,13,16},0.50)` = 9.5

`@PERCENTILE({4,5,7,9,10,12,13,16},0.75)` = 12.25

`@PERCENTILE({4,5,7,9,10,12,13,16},1)` = 16

The examples above return values from percentile increments of 0.25, which are equal to quartiles. See @QUARTILE.

@PERCENTRANK

Format

```
@PERCENTRANK(Array, X, <Digits>)
```

Array	=	a numeric array or a block of values
X	=	number to rank in Array; if X doesn't match a value in Array, @PERCENTRANK interpolates to return a percentage rank
Digits	=	number of significant digits for returned percentage value; must be ≥ 1 (the default is 3)

@PERCENTRANK returns the percentage rank of X in Array. Use @PERCENTRANK to see where a value stands within a list of values based on a percentage.

Example

This example refers to cells in the next figure. This formula returns the rank of a student's score among the scores of all test takers in the block A2..A11, where the student's score is in A4:

`@PERCENTRANK(A2..A11,A4,3)` = 0.222

Figure 19: @PERCENTRANK sample data

	A	B
1	Test Scores	
2	78	
3	80	
4	85	
5	85	
6	86	
7	87	
8	91	
9	92	
10	95	
11	98	
12		

See Also

@LARGEST
@MEDIAN
@PERCENTILE
@QUARTILE
@SMALLEST
@MAX
@MIN

@PERMUT

Format

```
@PERMUT(N, R)
```

N = number of different objects; $n \geq 0$

R = number of objects taken at a time; $R \leq N$

@PERMUT returns the total number of arrangements of objects taken R at a time from a set of N objects. The formula @PERMUT uses is:

$$\frac{N!}{(N-R)!}$$

@PERMUT is similar to @COMB except that it takes into account the order that objects are selected.

Example

Given 11 different colored marbles, this formula calculates how many different ways an ordered subset of five marbles can be constructed such that no two constructions contain the same five marbles in the same order. (Different constructions can contain the same five marbles, but they can't share the same ordering.)

`@PERMUT(11,5)` = 55,440

See Also

@COMB

@PIRATE

Format

```
@PIRATE(Npv, Flows, <Initial>, <[Odd|Periods]>, <Simp>, <Pathdep>, <Guess>,
  <Precision>, <Maxiter>, <Filter>, <Start>, <End>)
```

Npv	=	net present value
Flows	=	block containing cash flows
Initial	=	initial cash flow (the default is 0)
Odd|Periods	=	delay between initial and first cash flow, in number of periods (the default is 1) or block containing lengths of periods between cash flows (the default is 1)
Simp	=	flag specifying how to discount: 0 = compounded discounting (default) 1 = mixed compounded and simple discounting 2 = simple discounting
Pathdep	=	flag specifying whether to apply path-dependent compounding to each flow; 0 = no path (default); 1 = path
Guess	=	IRR guess for numerical search (useful for locating multiple roots); must be > -100%; the default is 0.10
Precision	=	minimum required precision; Precision > 0; the default is 0.000001
Maxiter	=	maximum number of iterations for search; Maxiter > 0; the default is 50
Filter	=	flag specifying filter type: 0 = no filter (default); 1 = cashflow < Start; 2 = cashflow ≤ Start; 3 = cashflow > Start; 4 = cashflow ≥ Start; 5 = Start < cashflow < End; 6 = Start ≤ cashflow ≤ End
Start	=	a starting cash flow amount to compare against individual flows
End	=	an ending cash flow amount to compare against individual flows

@PIRATE computes the internal rate of return for a stream of cash flows. It's similar to @IRR, but @PIRATE accommodates more complex cash flow structures.

The initial Guess for the discount rate is 10%. For some cash flow streams, particularly those with both positive and negative flows, multiple solutions are possible. By specifying a different Guess, it's possible to locate other solutions. If no solution exists, @PIRATE returns ERR.

The default value of Precision is 0.000001; smaller values need more search iterations and may require a larger value for Maxiter, which specifies the maximum number of iterations to use when attempting to find a solution. If the net present value of the cash flows doesn't converge within Precision to the target value specified by Npv (within Maxiter iterations), @PIRATE returns ERR.

Example

In the next figure, the stream consists of four flows, specified in block A12..A15. The time lengths of the periods preceding each flow are specified in block C12..C15. The Npv is $98.34. This formula calculates the internal rate of return, assuming compounded interest:

`@PIRATE(B17,A12..A15,0,C12..C15)` = 0.050041

	A	B	C	D
11	Cash Flows		Periods	

12	$4.50		0.3455
13	$4.50		1.2
14	$4.50		1
15	$104.50		1.5
16			
17	npv	$98.34	
18			

See Also
Entering Cash Flow @Functions

@POISSON

Format

```
@POISSON(N, Mean, Cum)
```

N	=	number of events; must be > 0
Mean	=	expected numeric value for the mean over the distribution; must be > 0
Cum	=	1 to return the cumulative Poisson probability distribution that the number of random events will be in the range from zero to N; 0 to return the Poisson probability mass function that the number of events will be N

@POISSON returns the Poisson probability distribution, that is, the probability that N number of events will occur over a given time period. The Poisson distribution function uses this formula:

$$p(r) = e^{-\mu}\frac{\mu^r}{r!}$$

with

$$r > 0$$

Example

On average, Company Z receives 30 customer service phone calls per hour. What is the probability that Company Z will receive 35 calls in one hour?

```
@POISSON(35,30,0)
```
= 0.045308

See Also

@EXPONDIST

@PRICE

Format

```
@PRICE(Settle, Maturity, Coupon, Yield, <Redemption>, <Freq>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Coupon	=	coupon rate; must be ≥ 0
Yield	=	annual yield; $0 \leq$ Yield ≤ 1
Redemption	=	redemption value per 100 face value; must be > 0
Freq	=	frequency of coupon payments in number of payments per year (can be 1, 2, 3, 4, 6, or 12; the default is 2)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@PRICE returns the price per 100 face value of a security that pays periodic interest.

Example

An 8.85% bond that matures August 17, 2017 has a yield-to-maturity of 7.5% for September 22, 1993 settlement. This formula calculates the price of the bond:

```
@PRICE(@DATE(93,9,22),@DATE(117,8,17),0.0885,0.075)
```
= 114.8902

@PRICEDISC

Format

```
@PRICEDISC(Settle, Maturity, Discount, <Redemption>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Discount	=	rate of discount; 0 ≤ Discount ≤ 1
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@PRICEDISC returns the dollar price per 100 face value of a discounted security. The dollar price of a bill is its value less the applicable dollar discount. The discount is the product of the redemption value and the quoted discount rate, prorated for the number of days between settlement and maturity. @PRICEDISC uses this formula:

$$P = R * \left(1 - D * \left(\frac{M - S}{t_b}\right)\right)$$

```
P = price
R = redemption
D = discount
M = maturity
S = settle
b = basis
```

t_b is the number of days over which the discount rate applies (360 or 365).

Example

This formula calculates the dollar price of a bill with the following terms: Settle is January 17, 1993, Maturity is August 15, 1993, Discount is 8.897%, Redemption is 100, and Calendar is 2 (actual/360).

```
@PRICEDISC(@DATE(93,1,17),@DATE(93,8,15),0.08897,100,2)
```
= 94.81008

@PRICEMAT

Format

```
@PRICEMAT(Settle, Maturity, Issue, Coupon, Yield, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Issue	=	number representing the issue date; must be < Settle
Coupon	=	coupon rate; 0 ≤ Coupon ≤ 1
Yield	=	annual yield; 0 ≤ Yield ≤ 1
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@PRICEMAT returns the price per 100 face value (not including accrued interest) of a security that pays interest at maturity. The invoice price of a security that pays interest at maturity is the sum of the quoted price and accrued interest between issue and settlement dates.

Example

This formula calculates the price per 100 face value of a security with the following terms: Settle is February 1, 1993, Maturity is April 1, 1993, Issue is January 2, 1993, Coupon is 10%, Yield is 10%, and Calendar is 2 (actual/360).

```
@PRICEMAT(@DATE(93,2,1),@DATE(93,4,1),@DATE(93,1,2),0.10,0.10,2)
```
= 99.986

@PROB

Format

```
@PROB(XData, ProbRange, LowerLimit, UpperLimit)
```

XData	=	values of X associated with the probabilities
ProbRange	=	a block or array of probability values associated with XData; each value in ProbRange must be ≥ 0 and ≤ 1; the sum of ProbRange values must equal 1
LowerLimit	=	lower limit on the value for the desired probability
UpperLimit	=	upper limit on the value for the desired probability

@PROB determines the probability that XData values are between two limits. If XData and ProbRange don't have the same number of values, @PROB returns ERR.

Example

`@PROB({10,13,15,18,25},{0.1,0.2,0.4,0.2,0.1},10,13)` = 0.3

@QUARTILE

Format

```
@QUARTILE(Array, X)
```

Array = a numeric array or a block of values

X = number signifying what quartile value to return:

0 = minimum value in Array

1 = 25th percentile

2 = 50th percentile (median)

3 = 75th percentile

4 = maximum value in Array

@QUARTILE returns a number from Array at the quartile indicated by X. You create quartiles of a data set by partitioning the values into four groups containing an equal number of values.

If the quartile falls between two discrete values in the list, a fractional value is determined using linear interpolation.

Examples

`@QUARTILE({4,5,7,9,10,12,13,16},0)` = 4

`@QUARTILE({4,5,7,9,10,12,13,16},1)` = 6.5

`@QUARTILE({4,5,7,9,10,12,13,16},2)` = 9.5

`@QUARTILE({4,5,7,9,10,12,13,16},3)` = 12.25

`@QUARTILE({4,5,7,9,10,12,13,16},4)` = 16

See Also

@LARGEST
@MEDIAN
@PERCENTILE
@PERCENTRANK
@SMALLEST
@MAX
@MIN

@QUOTIENT

Format

```
@QUOTIENT(X, Y)
```

X = value to divide

Y = nonzero value to divide x by

@QUOTIENT is similar to @INT; it returns the integer portion of X/Y. If Y is zero, @QUOTIENT returns ERR.

Example

`@QUOTIENT(7,2)` = 3

See Also

@INT

@RANDBETWEEN

Format

```
@RANDBETWEEN(N, M)
```

N = integer value that random number must be greater than or equal to

M = integer value that random number must be less than or equal to

@RANDBETWEEN returns a random number between N and M using a uniform distribution.
@RANDBETWEEN returns a new random number each time you recalculate a notebook.

See Also

@Rand

@RANK

Format

```
@RANK(Number, Array, Order)
```

Number	=	a number from Array
Array	=	one or more numeric or block values
Order	=	flag indicating how to sort the list of numbers: any nonzero value = ascending order; 0 = descending order

@RANK returns the rank of Number in Array in either ascending or descending order.

Examples

Theses examples refer to the next figure.

`@RANK(B6,B2..B10,0)` = 1

`@RANK(B2,B2..B10,1)` = 3

`@RANK(B9,B2..B10,1)` = 1

	A	B	C	D
1				
2	HJ	117		
3	MW	122		
4	KM	125		
5	WC	109		
6	AD	137		
7	MS	119		
8	DP	125		
9	MS	106		
10	DM	121		
11				

See Also

@PERCENTRANK

@RECEIVED

Format

```
@RECEIVED(Settle, Maturity, Investment, Discount, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date; must be > Settle
Investment	=	amount invested; must be > 0
Discount	=	rate of discount; 0 ≤ Discount ≤ 1
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@RECEIVED returns the redemption value (the amount received after maturity) of a discount security. @RECEIVED uses this formula:

$$R = \frac{I}{\left(1 - \frac{D(M - S)}{t_b}\right)}$$

```
R = redemption
I = investment
D = discount
M = maturity
S = settle
b = basis
```

t_b is the number of days over which the discount rate quote applies (360 or 365).

Example

This formula calculates the redemption value of a bill with the following terms: Settle is January 17, 1995, Maturity is June 15, 1995, Investment is $1,000,000, Discount is 7.922%, and Calendar is 2 (actual/360).

```
@RECEIVED(@DATE(95,1,17),@DATE(95,6,15),1000000,0.07922,2)
```
= $1,033,899.79

@RSQ

Format

```
@RSQ(KnownX, KnownY)
```

KnownX = independent range of values

KnownY = dependent range of values

@RSQ returns r , the square of the coefficient of correlation of the linear regression line through the data points in the arrays KnownX and KnownY. KnownX and KnownY must have the same number of values. Use @RSQ to test the linear relationship of KnownX and KnownY and to show the proportion of the variance of KnownY that can be attributed to a variance in KnownX. The r statistic measures the fraction of the variance explained by the regression equation.

Example

`@RSQ({7,8,5,9,7,2},{10,5,4,9,6,7})` = 0.063962

See Also

@CORREL
@COVAR
@INTERCEPT
@PEARSON
@SLOPE
@STEC
@STEYX

@SCMARG

Format

```
@SCMARG(Npv, Discrate, Flows, <Initial>, <[Odd|Periods]>, <Simp>,
  <Pathdep>, <Guess>, <Precision>, <Maxiter>, <Filter>, <Start>, <End>)
```

Npv	=	net present value
Discrate	=	discount rate or block containing discount rates corresponding to block of cash flows
Flows	=	block containing cash flows
Initial	=	initial cash flow (the default is 0)
Odd|Periods	=	delay between initial and first cash flow, in number of periods (the default is 1) or block containing lengths of periods between cash flows (the default is 1)
Simp	=	flag specifying how to discount: 0 = compounded discounting (default) 1 = mixed compounded and simple discounting 2 = simple discounting
Pathdep	=	flag specifying whether to apply path-dependent compounding to each flow; 0 = no path (default); 1 = path
Guess	=	initial margin for numerical search (useful for locating multiple roots) (must be > -100%; the default is 0)
Precision	=	minimum required precision (must be > 0; the default is 0.000001)
Maxiter	=	maximum number of iterations for search (must be > 0; the default is 50)
Filter	=	flag specifying filter type: 0 = no filter (default); 1 = cashflow < Start; 2 = cashflow ≤ Start; 3 = cashflow > Start; 4 = cashflow ≥ Start; 5 = Start < cashflow < End; 6 = Start ≤ cashflow ≤ End
Start	=	a starting cash flow amount to compare against individual flows
End	=	an ending cash flow amount to compare against individual flows

@SCMARG computes the discount scenario margin, a value that must be added to the Discrate series so that the net present value of the cash flow series equals Npv. An initial guess is made for the margin and is refined until the difference between the computed net present value and Npv is less than Precision. If @SCMARG doesn't find a solution within Maxiter search iterations, it returns ERR.

If Discrate is a number, @SCMARG returns the difference between the internal rate of return and Discrate. If Discrate is a block of discount rates, the margin is the amount to add to each discount rate to make the net present value equal Npv.

Example

A stream of cash flows is indexed off the current London InterBank Offer Rate (LIBOR). The time distances between flows is specified in block C11..C15. The first flow is 1 period away, the second is 1.5 periods after that, and so on. If the current LIBOR is 8.5%, this formula calculates the margin to LIBOR if the net present value is $167,000 and compounded discounting is used:

`@SCMARG(167000,0.085,A11..A15,0,C11..C15)` = 0.02599

	A	B	C	D	E

	Cash Flows	Time Periods
11	25000	1
12	35000	1.5
13	23000	1.2
14	50000	1
15	134500	1.3
16		

See Also
Entering Cash Flow @Functions

@SERIESSUM

Format

```
@SERIESSUM(X, N, M, Coefficients)
```

X	=	value in the power series
N	=	initial power to raise X to
M	=	increment of the power N for each successive term in the power series
Coefficients	=	a block or array of one or more numeric values by which each power of X is multiplied; the number of values in Coefficients sets the number of terms in the power series

@SERIESSUM returns the sum of a power series based on this formula for @SERIESSUM(X,N,M,A):

$$A_1X^N + A_2X^{(N+M)} + A_3X^{(N+2M)} + \ldots + A_iX^{(N+(i-1)M)}$$

Example

If the block A1..A3 contains the values 1, 2, and 3, then `@SERIESSUM(0.5,1,2,A1..A3)` = 0.84375

@SHLB

Format

```
@SHLB(Binary, <ShiftBits>, <BitIn>, <Bits>)
```

Binary	=	binary number
ShiftBits	=	number of bits to shift; 0 ≤ ShiftBits ≤ 64; the default is 1
BitIn	=	binary bit inserted during the shift (can be 0, 1, "S" or "I"; "S" = same as the least significant bit before shifting; "I" = inverse of the least significant bit before shifting; the default is 0)
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary; must be ≤ 64

@SHLB shifts the given binary number left by ShiftBits bits. @SHLB inserts the BitIn bit at the least significant bit (LSB).

Examples

`@SHLB(1000,1,0,5)` = 10000

`@SHLB(1000,1,1,6)` = 010001

`@SHLB(1000,2,1,6)` = 100011

See Also

Entering Boolean @Functions

@SHLBO

Format

```
@SHLBO(Binary, <Bits>)
```

Binary	=	binary number
Bits	=	number of input binary digits used during the shift operation; if omitted, Bits = the number of bits in Binary; must be ≤ 64

@SHLBO returns the overflow bit (either 0 or 1) of the given binary number after it has been shifted left by one bit.

An overflow occurs when a bit is shifted out of the word size specified by Bits. For example, if Binary = 1000 and Bits = 4, shifting the number left one bit results in 0000 with an overflow of 1 bit shifted to the fifth place not shown.

Examples

`@SHLBO(1000)` = 1

`@SHLBO(100,4)` = 0

`@SHLBO(1100,4)` = 1

See Also

Entering Boolean @Functions

@SHLH

Format

```
@SHLH(Hex, <ShiftBits>, <BitIn>, <Bits>)
```

Hex	=	hexadecimal number
ShiftBits	=	number of bits to shift; 0 ≤ ShiftBits ≤ 64; the default is 1
BitIn	=	binary bit inserted during the shift (can be 0, 1, "S" or "I"; "S" = same as the least significant bit before shifting; "I" = inverse of the least significant bit before shifting; the default is 0)
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@SHLH shifts the given hexadecimal number left by ShiftBits bits. @SHLH inserts the BitIn bit at the least significant bit (LSB).

Examples

`@SHLH("41",1)` = 82

`@SHLH("41",1,0,12)` = 082

`@SHLH("C",1,1,12)` = 019

See Also

Entering Boolean @Functions

@SHLHO

Format

```
@SHLHO(Hex, <Bits>)
```

Hex	=	hexadecimal number
Bits	=	number of equivalent input binary digits used during the shift operation; if omitted, Bits = the number of bits in Hex; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@SHLHO returns the overflow bit (either 0 or 1) of the given hexadecimal number after it has been shifted left by one bit.

An overflow occurs when a bit is shifted out of the word size specified by Bits. For example, if the binary equivalent of Hex = 1000 and Bits = 4, shifting the number left one bit results in 0000 with an overflow of 1 bit shifted to the fifth place not shown.

Examples

`@SHLHO("A")` = 1

`@SHLHO("A",5)` = 0

`@SHLHO("C",4)` = 1

See Also

Entering Boolean @Functions

@SHRB

Format

```
@SHRB(Binary, <ShiftBits>, <BitIn>, <Bits>)
```

Binary	=	binary number
ShiftBits	=	number of bits to shift; 0 ≤ ShiftBits ≤ 64; the default is 1
BitIn	=	binary bit inserted during the shift (can be 0, 1, "S" or "I"; "S" = same as the most significant bit before shifting; "I" = inverse of the most significant bit before shifting; the default is "S")
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary; must be ≤ 64

@SHRB returns the result of shifting the given binary number right by ShiftBits bits. @SHRB inserts the BitIn bit at the most significant bit (MSB).

Examples

`@SHRB(1000,1)` = 1100

`@SHRB(1000,1,1,5)` = 10100

`@SHRB(1100,1,0,6)` = 000110

See Also

Entering Boolean @Functions

@SHRBO

Format

```
@SHRBO(Binary, <Bits>)
```

Binary	=	binary number
Bits	=	number of input binary digits used during the shift operation; if omitted, Bits = the number of bits in Binary; must be ≤ 64

@SHRBO returns the overflow bit (either 0 or 1) of the given binary number after it has been shifted right by one bit.

An overflow occurs when a bit is shifted out of the word size specified by Bits. For example, if Binary = 1001 and Bits = 4, shifting the number right one bit results in 0100 with an overflow of 1 bit shifted off the right side.

Examples

`@SHRBO(1001)` = 1

`@SHRBO(10010,4)` = 0

`@SHRBO(1100,4)` = 0

See Also

Entering Boolean @Functions

@SHRH

Format

```
@SHRH(Hex, <ShiftBits>, <BitIn>, <Bits>)
```

Hex	=	hexadecimal number
ShiftBits	=	number of bits to shift; 0 ≤ ShiftBits ≤ 64; the default is 1
BitIn	=	binary bit inserted during the shift (can be 0, 1, "S" or "I"; "S" = same as the most significant bit before shifting; "I" = inverse of the most significant bit before shifting; the default is "S")
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@SHRH returns the result of shifting the given hexadecimal number right by ShiftBits bits. @SHRH inserts the BitIn bit at the most significant bit (MSB).

Examples

`@SHRH("41",1,1)` = A0

`@SHRH("41",1,1,4)` = 8

`@SHRH("C",1,0,12)` = 006

See Also

Entering Boolean @Functions

@SHRHO

Format

```
@SHRHO(Hex, <Bits>)
```

Hex = hexadecimal number

Bits = number of equivalent input binary digits used during the shift operation; if omitted, Bits = the number of bits in Hex; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@SHRHO returns the overflow bit (either 0 or 1) of the given hexadecimal number after it has been shifted right by one bit.

An overflow occurs when a bit is shifted out of the word size specified by Bits. For example, if the binary equivalent of Hex = 1001 and Bits = 4, shifting the number right one bit results in 0100 with an overflow of 1 bit shifted off the right side.

Examples

`@SHRHO("A")` = 0

`@SHRHO("B",5)` = 1

`@SHRHO("C",4)` = 0

See Also

Entering Boolean @Functions

@SKEW

Format

```
@SKEW(List)
```

List = one or more numeric or block values

@SKEW returns the skewness of a distribution. Skewness characterizes the degree of asymmetry of a distribution around its mean value. Use @SKEW when you want a non-dimensional quantity to measure the "shape" of a distribution rather than its moment, which is a measure in the same units as the elements of the distribution. @SKEW finds the skewness coefficient using this formula:

$$\frac{n}{(n-1)(n-2)} \sum \left(\frac{x_1 - \bar{x}}{s} \right)^3$$

A positive result means that the distribution is skewed to the right (the median is less than the mean). A negative result means that the distribution is skewed to the left (the median is greater than the mean). @SKEW returns 0 when the distribution is symmetrical around its mean.
If there are less than three data points in List, or if the standard deviation is zero, @SKEW returns ERR.

Example

`@SKEW(4,5,8,5,7,12,6,9,2,5)` = 0.685055

See Also

@KURT
@STD
@STDS
@VAR
@VARS

@SLOPE

Format

```
@SLOPE(KnownX, KnownY)
```

KnownX = independent range of values

KnownY = dependent range of values

@SLOPE returns the slope of the linear regression line through data points in x and y. The slope (the rate of change along the regression line) is the distance between the y values of two points, divided by the distance between their respective x values.

@SLOPE uses this formula:

$$b = \frac{n\sum xy - (\sum x)(\sum y)}{n\sum x^2 - (\sum x)^2}$$

Example

`@SLOPE({1,2,3,4,5,6},{4,8,12,16,20,24})` = 0.25

See Also

@INTERCEPT

@PEARSON

@RSQ

@STEYX

@SMALLEST

Format

```
@SMALLEST(Array, N)
```

Array	=	a numeric array or a block of values
N	=	number that indicates the rank in size from the data set Array; must be greater than 0 and less than or equal to the number of values in Array

@SMALLEST finds the Nth smallest number in Array. Use @SMALLEST to determine a value's rank in a data set from the bottom of that set.

Note: If there are duplicates in Array, @SMALLEST treats them as separate numbers.

Example

```
@SMALLEST({3,45,8,4,7,6,13,2,87,23,58,14,17,21},5)
```
= 7

See Also

@LARGEST
@MAX
@MEDIAN
@MIN
@PERCENTILE
@PERCENTRANK
@QUARTILE

@SQRTPI

Format

```
@SQRTPI(X)
```

X = value ≥ 0 to multiply by pi

@SQRTPI returns the square root of (@PI * X). If X is negative, @SQRTPI returns ERR.

Example

```
@SQRTPI(2)
```
= 2.506628

See Also

@PI
@SQRT

@STANDARDIZE

Format

```
@STANDARDIZE(X, Mean, SDev)
```

X	=	number to normalize
Mean	=	arithmetic mean of a distribution
SDev	=	standard deviation of a distribution

@STANDARDIZE normalizes the values from a distribution. A standard normal cumulative distribution assumes a mean of 0 and a standard deviation of 1. Use @STANDARDIZE to normalize values for use with other statistical @functions that require normally distributed variables (such as @ZTEST).

@STANDARDIZE uses this formula:

$$z = \frac{x - \mu}{\sigma}$$

Example

`@STANDARDIZE(2.6,1.6,0.5)` = 2

See Also

@NORMDIST
@NORMINV
@NORMSDIST
@NORMSINV

@STEC

Format

`@STEC(KnownY, KnownX)`

KnownY = dependent range of 3 or more values

KnownX = independent range of 3 or more values

@STEC computes the standard error of the regression coefficients.

@STEC is equal to (StdError divided by Std(x)) times

$$\sqrt{n - 1}$$

where StdError is:

$$\sqrt{(n - 1)(n - 2)} * \left(Var(yv) - \sigma^2 * Var(xv)\right)$$

and

$$\sigma = \frac{n * \Sigma(xv) - \Sigma(xv)\Sigma(yv)}{n * \Sigma\left(xv^2\right) - \Sigma(xv)^2}$$

`v` = values

@STEC returns ERR if KnownY and KnownX do not have the same number of values, or if KnownY and KnownX have less than 3 values each.

Example

This formula calculates the standard error of regression coefficients for range y (3,7,4,5) in block A1..A4 and range x (3.4,5.3,6,8) in block B1..B4:

`@STEC(A1..A4,B1..B4)` = 0.593769

See Also

@INTERCEPT

@PEARSON

@RSQ

@SLOPE

@STEYX

@STEYX

Format

```
@STEYX(KnownY, KnownX)
```

KnownY = dependent range of 3 or more values

KnownX = independent range of 3 or more values

@STEYX returns the standard error of a linear regression. The standard error is the deviation of the observed y values from the linear combinations. @STEYX uses this formula:

$$S_{yx} = \sqrt{\left[\frac{1}{n(n-2)}\right]\left[n\sum y^2 - (\sum y)^2 - \frac{[n\sum xy - (\sum x)(\sum y)]^2}{n\sum x^2 - (\sum x)^2}\right]}$$

If KnownY or KnownX have a different number of data points, or if the variance of KnownX = 0, @STEYX returns ERR.

Example

```
@STEYX({4,6,7,9,8},{7,9,12,9,5})
```
= 2.215697

See Also

@INTERCEPT

@PEARSON

@RSQ

@SLOPE

@STEC

@STKOPT

Format

```
@STKOPT(OptCode, OptPrem, UndStkValue, Date, Load, CmdString)
```

OptCode	=	option code string with expiration month, strike-price, and put or call symbol enclosed in quotation marks (for example, "MAY 22.5 C"); the strike price can be a decimal or fractional number, but not both (for example, it can be 11/32 or 1.625, but not 2 3/8); valid month codes are JAN, FEB, MAR, APR, MAY, JUN, JUL, AUG, SEP, OCT, NOV, and DEC; you can add spaces between month, strike-price, and Put or Call symbol; the total string (not including quotation marks) must be 20 characters or less.
OptPrem	=	option premium or price
UndStkValue	=	value or price of the underlying stock
Date	=	serial date number between 2 (January 1, 1900) and 73050 (December 31, 2099) representing the date on which to evaluate the stock option
Load	=	load or commission involved in sale or purchase
CmdString	=	command string enclosed in quotation marks specifying the operations to perform; cannot exceed 20 characters (not including quotes); see the table below for a list of valid CmdString characters

@STKOPT calculates values useful in evaluating stock options. It takes as input the basic information for the option, such as the option code (expiration month, strike price, and call or put indicator), the premium, the underlying value, the date at which the option is evaluated, and the associated "load" or fee. It then calculates a value such as the expiration date of the option, the number of calendar days remaining until the option expires, or the intrinsic value of the option.

If the expiration month specified in OptCode is earlier in the year than the month specified by Date, the expiration date of the option occurs in the following year.

Note: @STKOPT is valid only for options whose duration is 1 year or less

Load should be in the same units as OptPrem. You can use the Load argument as a general purpose value to incorporate any factor in the calculated result. For example, if you prefer a different annualization, such as 360/D, set Load to 360 and replace "A" in the CmdString formula with "L/D".

@STKOPT calculates output values according to a formula that you specify in the CmdString argument. The formula may include the following characters.

Character	Description
A	Annualization ratio (365.25/D); can be used to adjust percentages to equivalent yearly percentages
D	Days from DateTimeNum until expiration of option specified
E	Expiration date code of option specified
I	Intrinsic value of option
L	Load or fee to purchase or sell the option; can also be a general-purpose numeric constant
P	Premium or market value of the option
S	Strike or exercise price of the option
T	Time value of the option

U	Underlying value or price of the stock
+	Addition operator
-	Subtraction operator
*	Multiplication operator
/	Division operator

@STKOPT evaluates the command string formula from left to right with no operator precedence or associativity. The command string can be upper- or lowercase. The next table shows some examples of command strings.

String	Result
"D"	Outputs the number of days until option expiration
"I"	Outputs the intrinsic value
"L/P"	Outputs the ratio of the Load divided by the Premium
"T/U*A"	Time Value / Underlying value * Annualization ratio
"P-L/U*A"	((Premium - Load) / Underlying value) * Annualization ratio

For a call, the intrinsic value is the underlying value minus the strike price or zero, whichever is greater. For a put, the intrinsic value is the strike price minus the underlying value or zero, whichever is greater. The time value is equal to the premium minus the intrinsic value. The load is an expense for either the buyer or the seller. A positive time value is considered to be an expense for the buyer and income for the seller of the option.

Examples

A November put-option on stock of company XYZ has a strike price of $65 and a premium of $2.625, with an average load of $.12 per share. XYZ's stock is currently trading at $67 per share, and the date of evaluation is June 8, 1993.

Expiration date:

`@STKOPT("NOV65P",2.625,67,@DATE(93,6,8),.12,"E")` = 34293 (November 20, 1993)

Number of days to expiration:

`@STKOPT("NOV65P",2.625,67,@DATE(93,6,8),.12,"D")` = 165

Time value/stock price:

`@STKOPT("NOV65P",2.625,67,@DATE(93,6,8),.12,"T/U")` = 0.039179

Percentage load:

`@STKOPT("NOV65P",2.625,67,@DATE(93,6,8),.12,"L/P")` = 0.045714

See Also

@FEETBL

@SUBB

Format

```
@SUBB(Binary1, Binary2, <BitIn>, <Bits>)
```

Binary1	=	first binary number
Binary2	=	second binary number
BitIn	=	input borrow bit (either 0 or 1); the default is 0
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be ≤ 64

@SUBB returns the difference of two binary numbers.

Use two's complement notation to represent negative numbers. If BitIn is 1, @SUBB subtracts one extra bit from the result.

Examples

`@SUBB(100,100,0,1)` = 0

`@SUBB(1000,100,1,3)` = 011

`@SUBB(1100,1,1,5)` = 01010

See Also

Entering Boolean @Functions

@SUBBO

Format

```
@SUBBO(Binary1, Binary2, <BitIn>, <Bits>)
```

Binary1	=	first binary number
Binary2	=	second binary number
BitIn	=	input borrow bit (either 0 or 1); the default is 0
Bits	=	number of input binary digits used for subtraction; if omitted, Bits = the number of bits in Binary1 or Binary2, whichever is greater; must be ≤ 64

@SUBBO returns the overflow bit (either 0 or 1) of the difference of two binary numbers. An overflow occurs when a bit is borrowed from outside the word size specified by Bits. For example, if Binary1 = 10 and Binary2 = 110, the difference of the two numbers is 100 with 1 borrow bit in the fourth place not shown.

Use two's complement notation to represent negative numbers. If BitIn is 1, @SUBBO subtracts one extra bit from the difference of the two numbers before returning the overflow.

Examples

`@SUBBO(1000,1111)` = 1

`@SUBBO(1000,111,1,5)` = 0

`@SUBBO(1100,1101,1,4)` = 1

See Also

Entering Boolean @Functions

@SUBH

Format

```
@SUBH(Hex1, Hex2, <BitIn>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
BitIn	=	input borrow bit (either 0 or 1); the default is 0
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@SUBH returns the difference of two hexadecimal numbers.

Use two's complement notation to represent negative numbers. If BitIn is 1, @SUBH subtracts one extra bit from the result.

Examples

`@SUBH("1000","100",1)` = 0EFF

`@SUBH("1000","100",0)` = 0F00

`@SUBH("C","1",1,8)` = 0A

See Also

Entering Boolean @Functions

@SUBHO

Format

```
@SUBHO(Hex1, Hex2, <BitIn>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
BitIn	=	input borrow bit (either 0 or 1); the default is 0
Bits	=	number of input binary digits used for subtraction; if omitted, Bits = the number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be $\leq$ 64

@SUBHO returns the overflow bit (either 0 or 1) of the difference of two hexadecimal numbers. An overflow occurs when a bit is borrowed from outside the word size specified by Bits. For example, if the binary equivalent for Hex1 and Hex2 are 10 and 110, respectively, the difference of the two numbers is 100 with 1 borrow bit in the fourth place not shown.

Use two's complement notation to represent negative numbers. If BitIn is 1, @SUBHO subtracts one extra bit from the difference of the two numbers before returning the overflow.

Examples

`@SUBHO(8,"F")` = 1

`@SUBHO(8,7,1,5)` = 0

`@SUBHO("C","D",1,4)` = 1

See Also

Entering Boolean @Functions

@SUMSQ

Format

```
@SUMSQ(List)
```

List = one or more numeric or block values

@SUMSQ returns the sum of the squares of the values in List.

Example

```
@SUMSQ(2,3)
```

See Also

@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXPY2
@SUMXY
@SUMXY2
@SUMPRODUCT

@SUMX2MY2

Format

```
@SUMX2MY2(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMX2MY2 returns the sum of the difference of the squares of the corresponding values in Array1 and Array2.

@SUMX2MY2 uses this formula:

$$\Sigma\left(x^2 - y^2\right)$$

If Array1 and Array2 have a different number of values, @SUMX2MY2 ignores extra values in the larger array.

Example

`@SUMX2MY2({3,4,5},{2,3,4})` = (9 - 4) + (16 - 9) + (25 - 16)) = 21

See Also

@SUMSQ
@SUMX2PY2
@SUMXMY2
@SUMXPY2
@SUMXY
@SUMXY2
@SUMPRODUCT

@SUMX2PY2

Format

```
@SUMX2PY2(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMX2PY2 returns the sum of the sum of squares of the corresponding values in Array1 and Array2.

@SUMX2PY2 uses this formula:

$$\Sigma\left(x^2 + y^2\right)$$

If Array1 and Array2 have a different number of values, @SUMX2PY2 ignores extra values in the larger array.

Example

`@SUMX2PY2({3,4,5},{2,3,4})` = (9 + 4) + (16 + 9) + (25 + 16) = 79

See Also

@SUMSQ
@SUMX2MY2
@SUMXMY2
@SUMXPY2
@SUMXY
@SUMXY2
@SUMPRODUCT

@SUMXMY2

Format

```
@SUMXMY2(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMXMY2 returns the sum of the squares of the differences of corresponding values in Array1 and Array2.

@SUMXMY2 uses this formula:

$$\Sigma(x - y)^2$$

If Array1 and Array2 have a different number of values, @SUMXMY2 ignores extra values in the larger array.

Example

`@SUMXMY2({3,4,5},{2,3,4})` = (3 - 2) + (4 - 3) + (5 - 4) = 3

See Also

@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXPY2
@SUMXY
@SUMXY2
@SUMPRODUCT

@SUMXPY2

Format

```
@SUMXPY2(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMXPY2 returns the sum of the squares of the corresponding values in Array1 and Array2.

@SUMXPY2 uses this formula:

$$\Sigma(x + y)^2$$

If Array1 and Array2 have a different number of values, @SUMXPY2 ignores extra values in the larger array.

Example

`@SUMXPY2({3,4,5},{2,3,4})` = (3 + 2) + (4 + 3) + (5 + 4) = 155

See Also

@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXY
@SUMXY2
@SUMPRODUCT

@SUMXY

Format

```
@SUMXY(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMXY returns the sum of the products of the corresponding numbers in Array1 and Array2.

@SUMXY uses this formula:

$$\Sigma(xy)$$

If Array1 and Array2 have a different number of values, @SUMXY ignores extra values in the larger array.

Example

`@SUMXY({3,4,5},{2,3,4})` = (3 * 2) + (4 * 3) + (5 * 4) = 38

See Also

@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXPY2
@SUMXY2
@SUMPRODUCT

@SUMXY2

Format

```
@SUMXY2(Array1, Array2)
```

Array1 = first array of numeric values

Array2 = second array of numeric values

@SUMXY2 returns the sum of the squares of the products of the corresponding values in Array1 and Array2.

@SUMXY2 uses this formula:

$$\Sigma(xy)^2$$

If Array1 and Array2 have a different number of values, @SUMXY2 ignores extra values in the larger array.

Example

`@SUMXY2({3,4,5},{2,3,4})` = (3 * 2) + (4 * 3) + (5 * 4) = 580

See Also

@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXPY2
@SUMXY
@SUMPRODUCT

@TBILLEQ

Format

```
@TBILLEQ(Settle, Maturity, Discount)
```

Settle = number representing the settlement date; must be < Maturity

Maturity = number representing the maturity date

Discount = rate of discount expressed as a decimal fraction; must be ≥ 0 and ≤ 1

@TBILLEQ returns the bond equivalent yield (also called coupon equivalent yield) of a Treasury bill. Use @TBILLEQ to compare the yields of Treasury bills and bonds.

Example

This formula calculates the bond equivalent yield of a bill quoted at 9.35% with a maturity date of October 16, 1996 for settlement on April 9, 1996:

```
@TBILLEQ(@DATE(96,4,9),@DATE(96,10,16),0.0935)
```
= 0.099524

@TBILLPRICE

Format

```
@TBILLPRICE(Settle, Maturity, Discount)
```

Settle = number representing the settlement date; must be < Maturity

Maturity = number representing the maturity date

Discount = rate of discount expressed as a decimal fraction; must be ≥ 0 and ≤ 1

@TBILLPRICE calculates the price per 100 face value of a Treasury bill. The dollar price of a Treasury bill is its face value less the applicable dollar discount. @TBILLPRICE uses this formula:

$$P = 100 * \left(1 - D * \left(\frac{M - S}{t_b}\right)\right)$$

```
P = price
D = discount
M = Maturity
S = Settle
b = basis
```

Example

This formula calculates the price per 100 face value for a bill maturing August 1, 1996 and trading at 9.14% for January 3, 1996 settlement:

```
@TBILLPRICE(@DATE(96,1,3),@DATE(96,8,1),0.0914)
```
= 94.64294

@TBILLYIELD

Format

```
@TBILLYIELD(Settle, Maturity, Price)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Price	=	the Treasury bill's price per 100 face value

@TBILLYIELD returns the yield of a Treasury bill.

Example

This formula calculates the yield of a bill maturing August 1, 1996 and trading at a price of 94.64294 for January 3, 1996 settlement:

```
@TBILLYIELD(@DATE(96,1,3),@DATE(96,8,1),94.64294)
```
= 0.096574

@TDIST

Format

```
@TDIST(X, DegFreedom, Tails)
```

X	=	value at which to evaluate the distribution
DegFreedom	=	integer number of degrees of freedom; must be ≥ 1
Tails	=	to return a one-tailed distribution; 2 to return a two-tailed distribution

@TDIST returns the one-tailed or two-tailed Student's t-distribution, which is the probability that a given realization will be greater than X. Use the t-distribution when comparing the means of small samples. The single-tailed t-distribution yields a probability of deviation in a single direction from the mean; a two-tailed t-distribution yields a probability of deviation in two directions.

If DegFreedom isn't an integer, @TDIST rounds it to the nearest integer.

Example

`@TDIST(2.228139,10,2)` = 0.05

See Also

@TINV
@TTEST

@TINV

Format

```
@TINV(Prob, DegFreedom)
```

Prob cumulative probability value; $0 \leq Prob \leq 1$

DegFreedom = number of degrees of freedom

@TINV returns the inverse of the t-distribution. Use the t-distribution when comparing the means of small samples.

Example

`@TINV(0.05,10)` = 2.228139

See Also

@TDIST

@TTEST

@TRIMMEAN

Format

```
@TRIMMEAN(Array, Fraction)
```

Array = numeric array or a block of values

Fraction = decimal fraction of data points to exclude; $0 \leq$ Fraction < 1

@TRIMMEAN computes the mean of a data set, with a fraction of the points excluded. @TRIMMEAN is helpful when you want to exclude outlying values from your analysis. Because the function excludes an equal number of data points from the top and bottom of the data set, it rounds the number of excluded data points down to the nearest multiple of 2.

Example

`@TRIMMEAN({3,6,7,8,10,11,11,14,15,20},0.2)` = 10.25

@TTEST

Format

```
@TTEST(Array1, Array2, Tails, Type)
```

Array1	=	first array of numeric values
Array2	=	second array of numeric values
Tails	=	1 to return a one-tailed test; 2 to return a two-tailed test
Type	=	a discrete variable specifying the type of test to conduct; 1 = a paired test; 2 = a two-sample equal variance test; 3 = a two-sample unequal variance test

@TTEST returns the probability associated with the Student's t-Test. Use @TTEST to test the means of two small samples.

If Array1 and Array2 have a different number of data points, @TTEST returns ERR.

Example

`@TTEST({62,77,73,69,54,67,59,76},{64,80,72,53,69,63,76,74},2,2)` = 0.68502

See Also

@TDIST

@TINV

@WEIBULL

Format

```
@WEIBULL(X, Alpha, Beta, Cum)
```

X	=	function parameter to evaluate
Alpha	=	parameter to the distribution; must be > 0
Beta	=	parameter to the distribution; must be > 0
Cum	=	a numeric value (0 or 1) indicating whether to use the cumulative distribution function (1) or the probability density function (0)

@WEIBULL returns the Weibull distribution, which is used to calculate the mean time to failure of a device. If Alpha = 1, @WEIBULL returns the same value as @EXPONDIST with Lambda = 1/Beta.

Examples

`@WEIBULL(20,2,15,1)` = 0.830987

`@WEIBULL(20,2,15,0)` = 0.030047

See Also

@EXPONDIST

@WKDAY

Format
`@WKDAY(Date)`

Date = number representing a date to check

@WKDAY returns a number (from 1 for Saturday to 7 for Friday) representing the day Date falls on.

Example
`@WKDAY(@DATE(95,6,22))` = 6 (Thursday), since June 22, 1995 falls on a Thursday.

See Also
@LWKDAY
@NWKDAY

@XORB

Format

```
@XORB(Binary1, <Binary2>, <Bits>)
```

Binary1 = first binary number

Binary2 = second binary number

Bits = number of binary bits used for both input and output; if omitted, Bits = number of bits in Binary1 or Binary2, whichever is greater; must be ≤ 64

@XORB performs a bit-by-bit logical exclusive OR of each bit in Binary1 and Binary2. Use @XORB to set bits to 1 if the bits being compared in Binary1 and Binary2 are different.

If only one number is given, then @XORB performs an exclusive OR operation on the bits in Binary1; this parity test, or XOR reduction, returns 1 or 0 based on successive bit comparisons.

Examples

`@XORB(10,1)` = 11

`@XORB(11,10)` = 01

`@XORB(11)` = 0

`@XORB(1100,110,5)` = 01010

See Also

Entering Boolean @Functions

@XORH

Format

```
@XORH(Hex1, <Hex2>, <Bits>)
```

Hex1	=	first hexadecimal number
Hex2	=	second hexadecimal number
Bits	=	number of binary bits used for both input and output; if omitted, Bits = number of bits in Hex1 or Hex2, whichever is greater; 4 binary digits = 1 hexadecimal digit; must be ≤ 64

@XORH performs a bit-by-bit logical exclusive OR of each bit in Hex1 and Hex2. Use @XORH to set bits to 1 if the bits being compared in Hex1 and Hex2 are different.

If only one number is given, then @XORH performs the exclusive OR operation on the bits in Hex1; this parity test, or XOR reduction, returns 1 or 0 based on successive bit comparisons.

Examples

`@XORH("A","A")` = 0

`@XORH("E")` = 1

`@XORH("C","6",8)` = 0A

See Also

Entering Boolean @Functions

@YDAYS

Format

```
@YDAYS(Year)
```

Year = number from 0 (1900) to 199 (2099) or a standard year like 1993

@YDAYS returns the number of calendar days in a specified year.

Example

`@YDAYS(1996)` = 366, the number of days in 1996, since it's a leap year.

See Also

@BDAYS
@HOLS
@CDAYS
@MDAYS
@MNTHS

@YDIV

Format

```
@YDIV(Date, <Numdiv>, <Months>, <Anchor>, <Endmnth>)
```

Date	=	number representing the date about which to calculate year division
NumDiv	=	integer representing number of divisions away from beginning of division in which date falls; can be negative
Months	=	number of months per division; doesn't have to be a divisor of 12; can be longer than 1 year; must be an integer ≥ 0 (the default is 3)
Anchor	=	date to anchor division boundaries on (the default is January 1, 1900)
EndMnth	=	1 to indicate adherence to ends of months; 0 to indicate that ends of months are ignored (the default is 1)

@YDIV returns the date of the beginning of the year division (quarter, half-year, or some other fixed period of months, specified by Months) in which Date falls.

If NumDiv is used, @YDIV returns the date that's NumDiv divisions away from the beginning of the division in which Date falls. For example, @YDIV can calculate the beginning of the second quarter after the quarter containing March 21, 1993.

By default, divisions start at January 1, 1900; all other boundaries are some number of divisions away from January 1, 1900. For example, the default quarter boundaries are January 1, April 1, July 1, and October 1 of each year. Use Anchor to specify an alternate start date.

Division cycles can be changed to coincide with any date specified by Anchor. For example, you can change the quarter boundaries to February 23, May 23, August 23, and November 23 by setting Anchor to one of these dates in any year. When the division length is an integral divisor of 12, the year component of Anchor is ignored since the boundary dates repeat each year.

Examples

`@YDIV(@DATE(94,8,19))` = 34516 (July 1, 1994), the date of the beginning of the quarter in which August 19, 1994, falls (assuming that quarters begin January 1, April 1, July 1 and October 1).

A bond pays interest every six months until the bond's maturity, at which time it pays the final interest. The next formula calculates the date of the next coupon payment after September 12, 1994 if the bond matures July 15, 1999:

`@YDIV(@DATE(94,9,12),1,6,@DATE(99,7,15))` = 34714 (January 15, 1995)

The value 1 (NumDiv), passed as the second argument, designates the beginning of the next division (next coupon payment date). The value 6 (Months), passed as the third argument, designates that divisions are six months long. The fourth argument (Anchor) designates that year divisions must coincide with that date; in this case, that coupon dates coincide with the maturity date.

@YEARFRAC

Format

```
@YEARFRAC(StartDate, EndDate, <Calendar>)
```

StartDate = number representing the start date

EndDate = number representing the end date

Calendar = flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@YEARFRAC returns the fraction of the year covered between StartDate and EndDate.

Example

This formula calculates the fraction of a year between August 15, 1993 and November 4, 1993:

`@YEARFRAC(@DATE(93,8,15),@DATE(93,11,4))` = 0.222222

@YIELD

Format

```
@YIELD(Settle, Maturity, Issue, Coupon, Price, <Calendar>)
```

Settle	=	number representing the settlement date
Maturity	=	number representing the maturity date
Issue	=	number representing the issue date
Coupon	=	coupon rate; must be ≥ 0
Price	=	price; must be > 0
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@YIELD returns the annual yield of a security that pays periodic interest.

Dates for @YIELD must follow this pattern:

```
Issue < Settle < Maturity
```

Example

This formula calculates the annual yield of an 11.75% bond that was issued January 15, 1993 and matures November 15, 2014. The bond was priced at 126.1875 for settlement on January 23, 1993.

```
@YIELD(@DATE(93,1,23),@DATE(114,11,15),@DATE(93,1,15),0.1175,126.1875) =
```
0.089877

@YIELDDISC

Format

```
@YIELDDISC(Settle, Maturity, Price, <Redemption>, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Price	=	settlement price; must be > 0
Redemption	=	redemption value per 100 face value (must be > 0; the default is 100)
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@YIELDDISC returns the annualized yield for a discount security. @YIELDDISC uses this formula:

$$Y = \left(\frac{R}{P} - 1\right) * \left(\frac{t_b}{M - S}\right)$$

```
Y = yield
R = redemption
P = price
b = basis
M = maturity
S = settle
```

t_b is the number of days over which the discount rate applies (360 or 365).

Example

This formula calculates the yield on a bill maturing October 15, 1993 and trading at a price of 97.8 for July 15, 1993 settlement, using an actual/360 calendar:

```
@YIELDDISC(@DATE(93,7,15),@DATE(93,10,15),97.8,100,2)
```
= 0.088023

@YIELDMAT

Format

```
@YIELDMAT(Settle, Maturity, Issue, Coupon, Price, <Calendar>)
```

Settle	=	number representing the settlement date; must be < Maturity
Maturity	=	number representing the maturity date
Issue	=	number representing the issue date; must be < Settle
Coupon	=	coupon rate; 0 ≤ Coupon ≤ 1
Price	=	price per 100 face value
Calendar	=	flag specifying which calendar to observe (0 = 30/360, 1 = actual/actual, 2 = actual/360, 3 = actual/365; the default is 0)

@YIELDMAT returns the annual yield of a security that pays interest at maturity.

Example

This formula calculates the annual yield of a security with the following terms: Settle is March 7, 1993, Maturity is November 3, 1993, Issue is November 8, 1992, Coupon is 6.247%, Price is 100, and Calendar is 3 (actual/365).

```
@YIELDMAT(@DATE(93,3,7),@DATE(93,11,3),@DATE(92,11,8),0.06247,100,3) =
```
0.06122

@YLD2YLD

Format

```
@YLD2YLD(Y1, F1, F2, <Q1>, <Q2>)
```

Y1	=	given yield to be converted to another compounding frequency
F1	=	number of times given yield is compounded per year
F2	=	number of times target yield is compounded per year
Q1	=	number of periods for quoted yields (F1 is the default)
Q2	=	number of periods for quoted yields (F2 is the default)

@YLD2YLD converts a yield expressed in one compounding frequency to another. For example, since bond yields are compounded twice per year while mortgage yields are compounded 12 times per year, some conversion is necessary to compare the two different yield figures.

To use @YLD2YLD, you supply Y1, F1, F2 and the optional arguments Q1 and Q2. The result is Y2. @YLD2YLD solves this equation for Y2:

$$\left(1 + \frac{Y1}{Q1}\right)^{F1} = \left(1 + \frac{Y2}{Q2}\right)^{F2}$$

Examples

This formula converts an annual yield of 6.5% into a semiannual yield:

`@YLD2YLD(0.065,1,2)` = 0.063977

@ZTEST

Format

```
@ZTEST(Array, X, <S>)
```

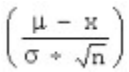
Array = a numeric array or a block of values

X = a value to test against the mean of the values in Array

S = population standard deviation; if omitted, @ZTEST uses the sample standard deviation

@ZTEST returns the two-tailed probability of a z-test. @ZTEST calculates a z-score, which is the distance between X and the mean for Array, and then returns the two-tailed probability of the z-score for a normal distribution. Use @ZTEST to test whether a value is drawn from a large sample population.

@ZTEST(Array,x) is equal to 1 minus @NORMDIST times



Example

```
@ZTEST({10,12,14,17,19,21,22,25},15)
```
= 0.087352

See Also

@NORMDIST
@NORMINV
@NORMSDIST
@NORMSINV
@STANDARDIZE

Using @Functions

Quattro Pro provides many built-in functions that perform calculations and return values. These functions, called @functions because they're preceded by an at-sign (@), are special commands entered in cells, either alone or in formulas.

Entering @Functions

It is important to enter @functions in the proper format, or syntax:

@FUNCTION(Argument1, Argument2,...)

Syntax Rules lists the rules to follow when entering @functions.

The type of information entered as an argument depends on the specific @function. For more information on arguments, see Arguments in @Functions.

@Function formulas

An @function formula is the complete command string, including arguments and punctuation. You enter an @function formula in a cell just as you would any other value; select the cell and type the entry. The @function displays on the input line. When you press Enter, the result of the @function displays in the cell. If the result is a numeric value, it becomes a value entry. If the result is a string, it becomes a label entry.

These features can help you construct @function formulas:

When you type anything, the **@Functions button** appears in the SpeedBar. Choose it to display a list of @functions types. Double-click on an @function type to display the @functions (or subtypes, if applicable) for that type. Press F1 for help on the highlighted @function. Choose an @function and press OK to enter it into a formula. The **Functions key** (Alt+F3) also displays the @function list.

Block names make it easier to reference blocks of cells. Once you've assigned a name to a block, you can reference it by name instead of coordinates. For details, see Using Blocks in Formulas.

Pointing lets you insert cell references in a function formula by pointing to them, either with the selector or the mouse. See Pointing Out Links for information on pointing out blocks in other open notebooks.

Input-line parenthesis matching makes it easier to enter @functions with proper syntax. For details, see Input-Line Parenthesis Matching.

@Function Help is available from the input line; type any @function, then press F1 to display help for it.

See Also

Nesting @Functions

Comments in @Functions

@Functions and Formulas

Entering Formulas

Using Arrays in Formulas

Syntax Rules

An @function formula's syntax must meet these rules:

- There must be a leading at-sign (@), plus sign (+), or equal sign (=) at the beginning of the formula.
- You can enter the @function in either uppercase or lowercase letters.
- If there are arguments, enclose them in parentheses.
- When there are multiple arguments, separate them by a semicolon or the argument separator specified by the punctuation setting of the International property in the application Object Inspector. (In the @function examples in Help, we use a comma to separate multiple arguments.)
- Enter arguments in the specified order.
- If you specify an optional argument for an @function, you must also specify all *preceding* optional arguments for the @function.
- There must be no spaces between @ and the @function name. There can be spaces between arguments, parentheses, and the @function name, but Quattro Pro deletes them.

See Also

@Functions and Formulas

Entering Formulas

Nesting @Functions

You can use more than one @function in a single formula. For example,

```
@ROUND(@AVG(C14..F24),2)
```

This @function formula calculates the average of block C14 to F24, then rounds the average to two decimal places.

This is called nesting @functions. You can nest as many @functions as you like in a formula (up to 1022 characters), but it is more difficult to debug complex @function formulas if there's a problem. It is best to break them down into several smaller formulas, which you can reference with a final formula. The smaller pieces make it easier to pinpoint an error, and are available for use with other formulas, so they use memory more efficiently.

See Also

@Functions and Formulas
Entering Formulas

Comments in @Functions

Regardless which @function you're using, you can include an onscreen comment along with it. Typically, comments serve as memory jogs, asides, or annotations that you don't want to appear in the notebook itself.

To enter a comment, type a semicolon (;) after the @function, then type in the text. The @function and comment together can be up to 1022 characters long.

Examples

```
@DMAX(A2..E10,4,A13..A14); computes the top sale in July
```

```
@MOD(@NOW,7); 0=Sun,1=Mon,2=Tues,3=Wed,4=Thurs,5=Fri,6=Sat
```

See Also

@Functions and Formulas

Entering Formulas

Arguments in @Functions

"Argument" refers to the information required by an @function. Most @functions need at least one argument. There are three general types of arguments:

Numeric values

Block values

String values

Most arguments require one of these types of values. Some @functions accept a combination or choice of types for a single argument. For example, @SUM accepts a block in combination with numeric values:

```
@SUM(B10..C25,50) totals numeric entries in block B10..C25 plus 50
```

The specific types of arguments for each @function are listed in the description for that @function.

See Also

@Functions and Formulas

Entering Formulas

Numeric @Function Arguments

A numeric value can be used as an @function argument in any of these forms:

- an actual value: `@SIN(1.571)`
- the coordinates of a cell containing a numeric value: `@SIN(B5)`
- the name of a cell block containing a single numeric value: `@INT(TOTAL)`
- a formula resulting in a numeric value: `@INT(B4*10)`
- another @function resulting in a numeric value: `@INT(@PI)`
- a combination of these: `@ABS(@INT(C4)+35-TOTAL)`

See Also

@Functions and Formulas

Entering Formulas

Block Arguments

Block @Function Arguments

A block of cells can be entered as an @function argument in a number of ways:

the coordinates of a block of cells: `@SUM(A1..B3)`

the address of a single cell: `@SUM(B3)`

a block name: `@SUM(JANUARY)`

You can use a combination of these in a list: `@SUM(JANUARY,C15..D25,F10)`

See Also

@Functions and Formulas

Entering Formulas

String Arguments

String @Function Arguments

A string value can be used as an @function argument in these ways:

an actual string, in double quotes: `@PROPER("ACME Company")`

the address of a cell containing a label: `@PROPER(G13)`

the name of a single-cell block containing a label: `@PROPER(COMPANY NAME)`

a formula resulting in a string: `@LOWER(+MONTH&"Sales")`

another @function resulting in a string: `@LENGTH(@PROPER("ACME Industries"))`

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Recalculation

By default, Quattro Pro calculates formulas in a spreadsheet when you enter them, and recalculates them each time you change the data involved. Quattro Pro offers three options for formula recalculation:

F9 CalcBackground (the default) recalculates between keystrokes.

Automatic pauses operation while Quattro Pro recalculates formulas.

Manual recalculates the formulas only when you press F9. (Quattro Pro still calculates formulas when you enter or edit them, but not when values in the cells they reference change.)

You change the recalculation mode in the Notebook property list.

The CALC indicator on the status line tells you that data has been added or changed, and Quattro Pro has not recalculated the spreadsheet. To recalculate a single cell in Manual recalculation mode, select the cell, press F2, then press Enter.

See Also

@Functions and Formulas

Entering Formulas

Database @Functions

The database @functions are similar to the statistical @functions. Instead of taking a simple list of values, these @functions all take three arguments, as in the following example:

@DMAX(Block,Column,Criteria).

@DAVG	The average (mean) of all numeric values in a given block.
@DCOUNT	The number of nonblank cells in a given block.
@DMAX	The largest numeric or latest date value in a given block.
@DMIN	The smallest numeric or earliest date value in a given block.
@DSTD	The population standard deviation of all values in a given block.
@DSTDS	The sample standard deviation of all values in a given block.
@DSUM	The total of all numeric values in a given block.
@DVAR	The population variance of all values in a given block.
@DVARS	The sample variance of all values in a given block.

See Also

@Functions and Formulas

Entering Formulas

Statistical @Functions

Date and Time @Functions

The date and time @functions calculate dates and times, or perform calculations involving business days.

Quattro Pro stores and calculates dates and time in "serial numbers." The integer part of the number represents the date, and the decimal fraction represents the time.

@ABDAYS	Adds (or subtracts) a given number of business days to a given date.
@ACDAYS	Adds a given number of calendar days to a given date.
@AMNTHS	Adds a given number of months to a given date.
@BDAYS	Returns the number of business days between two dates, inclusive of the second date.
@BUSDAY	Returns a given date if it is a business day, or the closest business day before (or after) the date.
@CDAYS	Returns the number of calendar days between two dates, inclusive of the second date.
@DATE	Returns the date number for a given year, month, and day; 0 = December 30, 1899.
@DATEVALUE	Returns the date number for a given formatted date string.
@DAY	Returns the day of the month (1-31) represented by a given date/time serial number.
@EMNTH	Returns the date of the last day of the month in which a specified date falls.
@FBDAY	Returns the date of the first business day of a month in which a specified date falls.
@HOLS	Returns the number of holidays between two dates, excluding holidays that fall on weekends.
@HOUR	Returns the number of hours past midnight (0-23) represented by a given date/time serial number.
@ISBDAY	Returns 1 if the specified date is a business day, or 0 if it is not.
@LBDAY	Returns the date of the last business day of a month in which a specified date falls.
@LWKDAY	Returns the date of the last given weekday in a given month.
@MDAYS	Returns the number of calendar days in a given month of a given year.
@MINUTE	Returns the number of minutes past the hour (0-59) represented by a given date/time serial number.
@MONTH	Returns the month in number form (1-12) represented by a given date/time serial number.
@MNTHS	Returns the number of whole months between two dates.
@NBDAY	Returns the date of the first valid business day after a given date.
@NOW	Returns the date and time serial number for the current system date and time.
@NWKDAY	Returns the date of the nth occurence of a given weekday in a given month.
@PBDAY	Returns the date of the first valid business day before a given date.
@SECOND	Returns the number of seconds past the minute (0-59) represented by a given

	date/time serial number.
@TIME	Returns the time number for a given hour, minute, and second. The hour is a numeric value between 0 and 23.
@TIMEVALUE	Returns the time number for a given formatted time string.
@TODAY	Returns the date number for the current system date.
@WKDAY	Returns the day of the week that a given date falls on.
@YDAYS	Returns the number of calendar days in a given year.
@YDIV	Returns the date of the beginning of the year division in which a given date or given number of divisions always falls.
@YEAR	Returns the year number (-300 to 1299; 0 = 1900) represented by a given date/time serial number.
@YEARFRAC	Returns the year fraction representing the number of whole days between a given starting and ending date.

See Also

Setting Holidays

@Functions and Formulas

Entering Formulas

@Function Arguments

Engineering @Functions

Bessel	These @functions return values that satisfy the Bessel equation or the modified Bessel equation. Bessel functions have various applications in physics and engineering.
Boolean	These @functions handle applications of digital logic and bitwise operations, which involve the testing, setting, or shifting of actual bits in a number.
Complex number	These @functions convert or modify a complex number (a number whose square is a negative real number).
Miscellaneous	These @functions have various applications in engineering, including converting values to different measures, testing the relationship of two numeric values, and returning error functions.
Number conversion	These @functions convert a value from one number system to another.

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Bessel Engineering @Functions

@BESSELI	Returns the modified Bessel function In(x).
@BESSELJ	Returns the modified Bessel function Jn(x).
@BESSELK	Returns the modified Bessel function Kn(x).
@BESSELY	Returns the modified Bessel function Yn(x).

See Also

Engineering @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Boolean Engineering @Functions

@ADDB	Returns the sum of two binary numbers.
@ADDBO	Returns the overflow bit of the sum of two binary numbers.
@ADDH	Returns the sum of two hexadecimal numbers.
@ADDHO	Returns the overflow bit of the sum of two hexadecimal numbers.
@ANDB	Returns the AND operation of two binary numbers.
@ANDH	Returns the AND operation of two hexadecimal numbers.
@BITRB	Returns a binary number with a specified bit reset to 0.
@BITRH	Returns a hexadecimal number with a specified bit reset to 0.
@BITSB	Returns a binary number with a specified bit set to 1.
@BITSH	Returns a hexadecimal number with a specified bit set to 1.
@BITTB	Returns the value of a specified bit in a binary number.
@BITTH	Returns the value of a specified bit in a hexadecimal number.
@CATB	Returns the concatenation of two binary numbers.
@CATH	Returns the concatenation of two hexadecimal numbers.
@CATNB	Returns the concatenation of *n* binary numbers.
@CATNH	Returns the concatenation of *n* hexadecimal numbers.
@INVB	Returns the inverse of a binary number.
@INVH	Returns the inverse of a hexadecimal number.
@ORB	Returns the OR operation of binary numbers.
@ORH	Returns the OR operation of hexadecimal numbers.
@SHLB	Returns a binary number shifted left by a specified number of bits.
@SHLBO	Returns the overflow bit of a binary number after it has been shifted left one bit.
@SHLH	Returns a hexadecimal number shifted left by a specified number of bits.
@SHLHO	Returns the overflow bit of a hexadecimal number after it has been shifted left one bit.
@SHRB	Returns a binary number shifted right by a specified number of bits.
@SHRBO	Returns the overflow bit of a binary number after it has been shifted right one bit.
@SHRH	Returns a hexadecimal number shifted right by a specified number of bits.
@SHRHO	Returns the overflow bit of a hexadecimal number after it has been shifted right one bit.
@SUBB	Returns the difference of two binary numbers.
@SUBBO	Returns the overflow bit of the subtraction of one binary number from another.
@SUBH	Returns the difference of two binary numbers.
@SUBHO	Returns the overflow bit of the subtraction of one hexadecimal number from another.
@XORB	Returns the exclusive OR operation of binary numbers.
@XORH	Returns the exclusive OR operation of hexadecimal numbers.

See Also

Engineering @Functions
Entering Boolean @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Complex Number Engineering @Functions

@COMPLEX	Converts real & imaginary coefficients into a complex number.
@IMABS	Returns the distance from the origin on a complex plane for a complex number.
@IMAGINARY	Returns the imaginary coefficient of a complex number.
@IMARGUMENT	Returns the argument theta, an angle expressed in radians.
@IMCONJUGATE	Returns the complex conjugate of a complex number.
@IMCOS	Returns the cosine of a complex number.
@IMDIV	Returns the quotient of two complex numbers.
@IMEXP	Returns the exponential of a complex number.
@IMLN	Returns the natural logarithm of a complex number.
@IMLOG10	Returns the base-10 logarithm of a complex number.
@IMLOG2	Returns the base-2 logarithm of a complex number.
@IMPOWER	Returns a complex number raised to a complex power.
@IMPRODUCT	Returns the product of two complex numbers.
@IMREAL	Returns the real coefficient of a complex number.
@IMSIN	Returns the sine of a complex number.
@IMSQRT	Returns the square root of a complex number.
@IMSUB	Returns the difference of two complex numbers.
@IMSUM	Returns the sum of complex numbers.

See Also

Engineering @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Miscellaneous Engineering @Functions

@CONVERT	Converts a number from one measurement system to another.
@DELTA	Tests whether two numbers are equal.
@ERF	Returns the error function.
@ERFC	Returns the complementary function.
@GESTEP	Tests whether a number is greater than a threshold value.

See Also

Engineering @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Number Conversion Engineering @Functions

@ASCTOHEX	Returns the hexadecimal string equivalent of an ASCII value
@BASE	Converts a base-10 number into another base.
@BINTOHEX	Converts a binary number to hexadecimal.
@BINTOHEX64	Converts a binary number (up to 64 bits) to hexadecimal.
@BINTONUM	Converts a binary number to decimal.
@BINTONUM64	Converts a binary number (up to 64 bits) to decimal.
@BINTOOCT	Converts a binary number to octal.
@BINTOOCT64	Converts a binary number (up to 64 bits) to octal.
@HEXTOASC	Converts a hexadecimal number to ASCII.
@HEXTOBIN	Converts a hexadecimal number to binary.
@HEXTOBIN64	Converts a hexadecimal number (up to 64 bits) to binary.
@HEXTONUM	Converts a hexadecimal number to decimal.
@HEXTONUM64	Converts a hexadecimal number (up to 64 bits) to decimal.
@HEXTOOCT	Converts a hexadecimal number to octal.
@HEXTOOCT64	Converts a hexadecimal number (up to 64 bits) to octal.
@NUMTOBIN	Converts a decimal number to binary.
@NUMTOBIN64	Converts a decimal number (up to 64 bits) to binary.
@NUMTOHEX	Converts a decimal number to hexadecimal.
@NUMTOHEX64	Converts a decimal number (up to 64 bits) to hexadecimal.
@NUMTOOCT	Converts a decimal number to octal.
@NUMTOOCT64	Converts a decimal number (up to 64 bits) to octal.
@OCTTOBIN	Coverts an octal number to binary.
@OCTTOHEX	Converts an octal number to hexadecimal.
@OCTTONUM	Converts an octal number to decimal.

See Also

Engineering @Functions
Entering Number Conversion @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Financial @Functions

Annuity	The investment @functions involve a series of periodic payments over a term measured in the number of payment periods. This set of @functions allows you to compute one value, knowing three of the other values.
Bill	These @functions compute values for Treasury billls.
Bond	These @functions compute values for bonds.
Cash Flow	These @functions operate on tables of data that record income and expenditures.
CD	These @functions compute values for certificates of deposit.
Depreciation	These @functions compute depreciation over time.
Stock	These @functions compute values for common stock.

See Also

Calendar Conventions

@Functions and Formulas

Entering Formulas

@Function Arguments

Annuity Financial @Functions

@AMAINT	Calculates the accumulated interest paid on an amortized loan after n payments.
@AMINT	Calculates the periodic interest rate for an amortized loan.
@AMPMT	Calculates the periodic payment for an amortized loan.
@AMPMTI	Calculates the interest portion of the nth periodic payment of an amortized loan.
@AMPRN	Calculates the initial principal of an amortized loan.
@AMRES	Calculates the end value of an amortized loan or the future value of an annuity.
@AMRPRN	Calculates the remaining balance of an amortized loan after n payments.
@AMTERM	Calculates the length of an amortized loan, expressed as number of payments.
@CTERM	Returns the number of compounding time periods required to achieve a specified future value, given the present value and interest rate.
@FV	Returns the future value of an investment at the end of the term, given the payment, interest rate, and number of payments.
@FVAL	Returns the future value of an investment, given the interest rate, number of payments, periodic payment rate, and optional present value and type.
@IPAYMT	Returns the interest portion of a single payment given the interest rate, period, number of periods, present value, and optional future value and type.
@IRATE	Returns the interest rate given the number of payments, payment amount, present value, and optional future value and type.
@MTGACC	Returns the new loan term, the payoff-date, or interest saved by paying extra monthly principal for a home loan.
@NPER	Returns the number of periods given the interest rate, payment amount, present value, and optional future value and type.
@PAYMT	Returns periodic payments given the rate, number of payments, present value, and optional future value and type.
@PMT	Returns the periodic payment required to fully amortize the principal during the term, given present value, interest rate, and number of payments.
@PPAYMT	Returns the principal portion of a single payment.
@PV	Returns the present value of an investment, given periodic payment, interest rate, and number of periods.
@PVAL	Returns the present value of an investment, given the interest rate, number of payments, periodic payment rate, and optional future value and type.
@RATE	Returns the interest rate required to achieve a given future value, given the future value, present value, and number of payments.
@TERM	Returns the number of periodic payments required to achieve a specified future value, given the periodic payment amount and interest rate.
@YLD2YLD	Converts a yield expressed in one compounding frequency and time length to that in another frequency and/or time length.

See Also

Annuity @Function Arguments

Financial @Functions

Calendar Conventions

Bill Financial @Functions

@DISC	Returns the discount rate for a security.
@INTRATE	Returns the simple annualized yield for a fully invested security.
@PRICEDISC	Returns the price per 100 face value of a security that pays periodic interest.
@RECEIVED	Returns the amount received at maturity for a fully invested security.
@TBILLEQ	Returns the bond equivalent yield for a Treasury bill.
@TBILLPRICE	Returns the price per 100 face value for a Treasury bill.
@TBILLYIELD	Returns the yield for a Treasury bill.
@YIELDDISC	Returns the annual yield for a discounted security.

See Also

Financial @Functions
Calendar Conventions
@Functions and Formulas
Entering Formulas
@Function Arguments

Bond Financial @Functions

@ACCRINT	Returns the accrued interest for a bond.
@COUPDAYBS	Returns the number of days from the beginning of the coupon period to the settlement date.
@COUPDAYS	Returns the number of days in the coupon period that contains the settlement date.
@COUPDAYSNC	Returns the number of days from the settlement date to the next coupon date.
@COUPNCD	Returns the next coupon date after the settlement date.
@COUPNUM	Returns the number of coupons payable between the settlement date and maturity date.
@COUPPCD	Returns the previous coupon date before the settlement date.
@DURATION	Returns the Macaulay duration of a security with par value of 100.
@MDURATION	Returns the modified Macauley duration for a security with an assumed par value of 100.
@ODDFPRICE	Returns the price per 100 face value of a security with an odd first period.
@ODDFYIELD	Returns the yield of a security with an odd first period.
@ODDLPRICE	Returns the price per 100 face value of a security with an odd last period.
@ODDLYIELD	Returns the yield of a security with an odd last period.
@PRICE	Returns the price per 100 face value of a security that pays periodic interest.
@YIELD	Returns the yield on a security that pays periodic interest.

See Also

Financial @Functions

Bond @Function Arguments

Calendar Conventions

@Functions and Formulas

Entering Formulas

@Function Arguments

Cash Flow Financial @Functions

@DURAT	Calculates the Macaulay duration of a cash flow stream.
@FUTV	Calculates the future value of a cash flow stream.
@IRR	Returns the internal rate of return of an investment.
@NETPV	Calculates net present value of a stream of cash flows.
@NPV	Returns the net present value of a future cash flow.
@PIRATE	Calculates the internal rate of return for a stream of cash flows.
@SCMARG	Calculates the discount scenario margin, the margin to add to each discount rate in order to arrive at a given net present value.

See Also

Financial @Functions

Entering Cash Flow @Functions

Calendar Conventions

@Functions and Formulas

Entering Formulas

@Function Arguments

CD Financial @Functions

@ACCRINTM	Returns the accrued interest for a security that pays interest at maturity.
@PRICEMAT	Returns the price per 100 face value of a security that pays interest at maturity.
@YIELDMAT	Returns the annual yield of a security that pays interest at maturity.

See Also

Financial @Functions
Calendar Conventions
@Functions and Formulas
Entering Formulas
@Function Arguments

Depreciation Financial @Functions

@DDB	Returns the double-declining balance depreciation of an asset during the given period.
@SLN	Returns the straight-line depreciation of an asset over each period in its given useful life.
@SYD	Returns the sum-of-the-years'-digits depreciation of an asset during the given period.

See Also

Calendar Conventions

Financial @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Stock Financial @Functions

@FEETBL	Returns fee calculations for stock transactions.
@STKOPT	Returns the time value and earnings value of a stock option.

See Also

Calendar Conventions
Financial @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Logical @Functions

@FILEEXISTS	Checks to see whether the named file exists. Returns a logical true (1) or false (0) value.
@IF	Evaluates a given condition, and returns the specified expression if it is true, or another specified expression if it is false.
@ISERR	Returns 1 if a given cell contains ERR (error indicator), otherwise 0.
@ISNA	Returns 1 if a given cell is NA (not available), otherwise 0.
@ISNUMBER	Returns 1 if a given cell contains a number, otherwise 0.
@ISSTRING	Returns 1 if a given cell contains a label or text string, otherwise 0.
@TRUE	Always returns the logical value 1.
@FALSE	Always returns the logical value 0.

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Mathematical @Functions

Mathematical @functions take one or more numeric values as arguments, and return a numeric value. You must enter all angles in radians for @COS, @SIN, and @TAN. Accordingly, @ASIN, @ATAN, and @ATAN2 return all angles in radians. To convert radians to degrees, use @DEGREES; to convert degrees to radians, use @RADIANS.

@ABS	Returns the absolute value of a given number.
@ACOS	Returns the angle whose cosine is a given value.
@ASIN	Returns the angle whose sine is a given value.
@ATAN	Returns the angle whose tangent is a given value.
@ATAN2	Returns the angle represented by a given pair of coordinates.
@CEILING	Rounds a number up to the nearest integer.
@COS	Returns the cosine of a given angle.
@DEGREES	Converts a given value from radians to degrees.
@DFRAC	Converts a decimal number to a whole number and fractional component.
@EVEN	Rounds a number up to the nearest even integer.
@EXP	Returns the exponential of a given value, which is the value of *e* (the mathematical constant) raised to the power of the given value. The value must be less than or equal to 709.
@FACT	Calculates the factorial of a number.
@FACTDOUBLE	Returns the double factorial of a number.
@FIB	Calculates the nth term of a Fibonacci sequence.
@FLOOR	Rounds a number down, toward zero.
@FRACD	Converts a number with a fractional component to a decimal.
@GCD	Calculates the greatest comman divisor of x and y.
@INT	Returns the integer portion of of a given value. In this @function, the number is simply truncated, not rounded off.
@LCM	Calculates the least common multiple of x and y.
@LINTERP	Performs linear interpolation between sets of xy pairs.
@LN	Returns the natural logarithm of a given value. This is the logarithm of the number to the base *e*; @LN is the inverse of @EXP (the value must be greater than 0).
@LOG	Returns the logarithm of a given value to base 10 (the value must be greater than 0).
@MDET	Calculates the determinant of a matrix.
@MOD	Returns the modulus of a given value with respect to another. (Modulus is the remainder when the first is divided by the second.)
@MROUND	Returns a number rounded to the desired multiple.
@MULT	Calculates cumulative product of a set of numbers.
@MULTINOMIAL	Returns the multinomial of a set of numbers.
@ODD	Rounds a number up to the nearest odd integer.

@PI	Returns the value of pi.
@QUOTIENT	Returns the integer portion of a division.
@RADIANS	Converts a given value from degrees to radians.
@RAND	Returns a fractional random number between 0 and 1.
@RANDBETWEEN	Returns a random number between the numbers you specify.
@ROUND	Rounds off a given value (with up to 15 digits) to a given number of decimal places.
@SERIESSUM	Returns the sum of a power series based on a formula.
@SIN	Returns the sine of a given angle.
@SQRT	Returns the square root of a given value (the value must be greater than or equal to 0).
@SQRTPI	Returns the square root of a number multiplied by pi.
@TAN	Returns the tangent of a given angle.

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Miscellaneous @Functions

Attribute	Each cell has a number of attributes, including its address, contents, type, protection status, format, etc. These @functions return the attributes of a specified cell. If a block of cells is given, they use the top left cell in the block.
Cell and Table	These @functions generally return simple information from a cell or block of cells.
Status	Return the current setting for commands, properties, and other elements.
Table Lookup	These @functions are used to search for a value in a block of cells that has been specified as a data table.

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Cell Attribute @Functions

The attribute @functions return the requested attribute of a cell (or of the upper left cell in a block).

@CELL	Returns the requested attribute of the upper left cell in a given block of cells.
@CELLINDEX	Returns the requested attribute of the cell at the intersection of a given column and row in a block of cells.
@CELLPOINTER	Returns the requested attribute of the active (currently selected) cell.

See Also

Miscellaneous @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Miscellaneous Cell and Table @Functions

@@	Returns the contents of a given cell.
@ARRAY	Returns the result of an expression (a formula or @function) using array syntax.
@BLOCKNAME	Returns the name of a specified block.
@BLOCKNAME2	Returns the block name created in a given notebook for a specified block.
@BLOCKNAMES	Returns a two-column table showing the block names that intersect with a specified block.
@BLOCKNAMES2	Returns a two-column table showing the block names created in a given notebook that intersect with a specified block.
@CHOOSE	Returns the element from a given list in the specified position.
@COLS	Returns the number of columns in a given block of cells.
@DDELINK	Creates a "live" data link from another Windows application that supports DDE (Dynamic Data Exchange).
@ERR	Always returns ERR (the error indicator).
@FIRSTBLANKPAGE	Returns the page letters for the first unnamed blank page in a notebook that isn't part of a group.
@FIRSTINGROUP	Returns the page letters for the first page in the specified group.
@GETGROUP	Returns the name of the group that contains the page for a specified block, or the name of the group in the specified block that contains a specified page name.
@INDEXTOLETTER	Returns a one- or two-character string for the index number of a page or column.
@ISLEGALPAGENAME	Returns 1 if the specified page name is valid (whether it exists or not); otherwise, returns 0.
@LASTBLANKPAGE	Returns the page letters for the last unnamed blank page in a notebook that isn't part of a group.
@LASTINGROUP	Returns the page letters for the last page in the specified group.
@LETTERTOINDEX	Returns the index number for column letters or page letters.
@NA	Always returns NA (not available).
@PAGEINDEX	Returns the index number for a notebook page with a specified name.
@PAGEINDEX2	Returns the index number for a notebook page with a specified name in a given notebook.
@PAGENAME	Returns the name of a notebook page with a given index number.
@PAGENAME2	Returns the name of a notebook page with a given index number in a specified notebook.
@PAGENAMES	Returns a two-column table showing the page letters and corresponding page names for the active notebook.
@PAGENAMES2	Returns a two-column table showing the page letters and corresponding page names for a specified notebook.
@ROWS	Returns the number of rows in a given block of cells.

@SHEETS	Returns the number of pages in a given block of cells.

See Also

Miscellaneous @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

Status @Functions

@COMMAND	Returns current settings for the given command equivalent.
@CURVALUE	Returns the current value of a given menu command setting.
@MEMAVAIL	Returns the number of bytes of available memory.
@MEMEMSAVAIL	Under DOS, returns the number of bytes of available expanded memory; under Windows, returns NA (included for compatibility with Quattro Pro for DOS).
@PROPERTY	Returns current property settings for the given object and property.
@VERSION	Returns the version number of Quattro Pro.

See Also

Miscellaneous @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Table Lookup @Functions

@HLOOKUP	Searches for a given value in the first row of a given block of cells. Returns the value a given number of rows from the first.
@INDEX	Returns the value at the intersection of a given column and row in a specified block of cells.
@VLOOKUP	Searches for a given value in the first column of a given block of cells. Returns the value from a given number of columns from the first.

See Also

Miscellaneous @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Statistical @Functions

The statistical @functions perform aggregation, counting, and analysis operations on a group of values expressed as a list (or lists) of one or more arguments. These arguments can be numeric values or block values.

Descriptive	These @functions return a value that helps you summarize and describe a group of values.
Inferential	These @functions return a value (or values) that helps you draw conclusions about a group (or groups) of values.

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

Descriptive Statistical @Functions

@AVG	Returns the average (mean) of all numeric values in a list.
@COUNT	Returns the number of nonblank cells in a list.
@GEOMEAN	Returns the geometric mean of all numeric values in a list.
@HARMEAN	Returns the harmonic mean of all numeric values in a list.
@KURT	Returns the kurtosis (peakedness or flatness) of a data set.
@LARGEST	Returns the k-th largest value in a data set.
@MAX	Returns the largest numeric or last date value in a list.
@MEDIAN	Returns the median of a data set.
@MIN	Returns the smallest numeric or earliest date value in a list.
@MODE	Returns the most common value in a data set.
@PERCENTILE	Returns the value from a group of values at a specified percentile.
@PERCENTRANK	Returns the percentage rank of a value in a data set.
@QUARTILE	Returns the quartile of a data set.
@RANK	Returns the rank of a number in a list of numbers.
@SKEW	Returns the skewness of a distribution.
@SMALLEST	Returns the k-th smallest value in a data set.
@STANDARDIZE	Returns a normalized value.
@STD	Returns the population standard deviation of all values in a list.
@STDS	Returns the sample standard deviation of all values ina list.
@SUM	Returns the total of all numeric values in a list.
@TRIMMEAN	Returns the mean of all numeric values in a list with a fraction of values excluded.
@VAR	Returns the population variance of all values in a list.
@VARS	Returns the sample variance of all values in a list.

See Also

Statistical @Functions

@Functions and Formulas

Entering Formulas

@Function Arguments

Inferential Statistical @Functions

@AVEDEV	Performs the average of the absolute deviations of data points from their means.
@BETA	Returns the beta function.
@BETADIST	Returns the cumulative beta probability density function.
@BETAI	Returns the incomplete beta function.
@BETAINV	Returns the inverse of the cumulative beta probability density function.
@BINOMDIST	Returns the binomial probability mass function.
@CHIDIST	Returns the cumulative chi-square distribution.
@CHIINV	Returns the inverse of the cumulative chi-square distribution.
@CHITEST	Computes the probability that the actual and expected frequencies are similar by chance (chi-square test).
@COMB	Calculates the number of unordered subgroups of given size in a group.
@CONFIDENCE	Returns the confidence interval for a population mean.
@CORREL	Returns the correlation coefficient of two data sets.
@COVAR	Returns the covariance of two data sets.
@CRITBINOM	Returns the smallest value for which the cumulative binomial distribution is less than or equal to a criterion value.
@DEVSQ	Returns the sum of the squares of the deviations.
@EXPONDIST	Returns the exponential distribution.
@FDIST	Returns the F distribution function.
@FINV	Returns the inverse of the cumulative F distribution function.
@FISHER	Returns the Fisher transformation.
@FISHERINV	Returns the inverse of the Fisher transformation.
@FORECAST	Returns a value along a linear trend.
@FTEST	Returns the result of the F-test.
@GAMMADIST	Returns the gamma distribution function.
@GAMMAINV	Computes the inverse of the cumulative Gamma distribution function.
@GAMMALN	Returns the natural logarithm of the gamma function.
@GAMMAP	Returns the incomplete gamma function.
@GAMMAQ	Returns the complement of the incomplete gamma function.
@HYPGEOMDIST	Returns the hypergeometric distribution.
@INTERCEPT	Returns the intercept of the linear regression line.
@LOGINV	Returns the inverse of the lognormal distribution.
@LOGNORMDIST	Returns the lognormal distribution.
@NEGBINOMDIST	Returns the negative binomial distribution.
@NORMDIST	Returns the normal cumulative distribution.
@NORMINV	Computes the inverse of the cumulative normal distribution function.

@NORMSDIST	Computes the standard normal cumulative distribution.
@NORMSINV	Returns the inverse of the standard normal cumulative distribution.
@PEARSON	Returns the Pearson product moment correlation coefficient.
@PERMUT	Calculates the number of ordered subgroups of given size in a group (permutations).
@POISSON	Returns the Poisson probability distribution.
@PROB	Returns the probability that values in a range are between two limits.
@RSQ	Returns the square of the coefficient of correlation of the linear regression line through data points in known xs and known ys.
@SLOPE	Returns the slope of the linear regression line.
@STEC	Returns the standard error of the regression coefficient.
@STEYX	Standard error of the predicted y-value for each x.
@SUMPRODUCT	The dot (scalar) product of the vectors corresponding to a block of cells.
@SUMSQ	Returns the sum of the squares of the arguments.
@SUMX2MY2	Returns the sum of the differences of the squares of the corresponding values in two arrays.
@SUMX2PY2	Returns the sum of the sum of the squares of corresponding values in two arrays.
@SUMXMY2	Returns the sum of squares of differences of corresponding values in two arrays.
@SUMXPY2	Returns the sum of the squares of corresponding values in two arrays.
@SUMXY	Sum of the products of the corresponding numbers in two arrays.
@SUMXY2	Sum of the product of values and the squares of the corresponding numbers in two arrays.
@TDIST	Returns the Student's t-distribution.
@TINV	Returns the inverse of the Student's t-distribution.
@TTEST	Returns the probability associated with the Student's t-test.
@WEIBULL	Returns the Weibull distribution.
@ZTEST	Returns the two-tailed probability value of a z-test.

See Also

Statistical @Functions
@Functions and Formulas
Entering Formulas
@Function Arguments

String @Functions

String @functions work on strings of characters, or text. They include one or more strings as arguments, and return either a string or a numerical value.

@CHAR	Returns the ANSI character that corresponds to the decimal code given as the argument.
@CLEAN	Returns a given string with any non-printable ASCII codes removed.
@CODE	Returns the ANSI code of the first character in a given string.
@EXACT	Returns 1 if two given strings are identical (including capitalization), otherwise returns 0.
@FIND	Looks for a given substring in a given string, beginning with the character in the specified position. Returns the position of the first matched character in the string.
@LEFT	Returns a given number of characters from the beginning of a given string.
@LENGTH	Returns the length of a given string, including spaces.
@LOWER	Returns a given string with all the alphabetic characters converted to lowercase (small letters).
@MID	Returns a given number of characters from a given string, starting with the character in the specified position.
@N	Returns the numeric value of the top left cell of a block of cells.
@PROPER	Returns a given string with the first letter of each word capitalized, and with all other letters lowercase.
@REPEAT	Returns a string made up of a given number of repetitions of a given string.
@REPLACE	Deletes a given number of characters in a given string, beginning with a specified position, and replaces them with a specified string. Returns the modified string.
@RIGHT	Returns a given number of characters from the end of a given string.
@S	Returns the string value of the top left cell of a block of cells.
@STRING	Converts a given numeric value into a string, rounding to a given number of decimal places.
@TRIM	Returns a given string without leading or trailing spaces, and without multiple spaces.
@UPPER	Returns a given string with all of the alphabetic characters converted to uppercase (capital letters).
@VALUE	Returns the numeric value of a given string. (Returns ERR if the argument is not a simple number string.)

See Also

@Functions and Formulas

Entering Formulas

@Function Arguments

@Function Index

@BETAI
@BETAINV
@BINOMDIST
@BINTOHEX
@BINTOHEX64
@BINTONUM
@BINTONUM64
@BINTOOCT
@BINTOOCT64
@BITRB
@BITRH
@BITSB
@BITSH
@BITTB
@BITTH
@BLOCKNAME
@BLOCKNAME2
@BLOCKNAMES
@BLOCKNAMES2
@BUSDAY
@CATB
@CATH
@CATNB
@CATNH
@CDAYS
@CEILING
@CELL
@CELLINDEX
@CELLPOINTER
@CHAR
@CHIDIST
@CHIINV
@CHITEST
@CHOOSE
@CLEAN
@CODE
@COLS
@COMB
@COMMAND
@COMPLEX

@CONFIDENCE
@CONVERT
@CORREL
@COS
@COUNT
@COUPDAYBS
@COUPDAYS
@COUPDAYSNC
@COUPNCD
@COUPNUM
@COUPPCD
@COVAR
@CRITBINOM
@CTERM
@CURVALUE
@DATE
@DATEVALUE
@DAVG
@DAY
@DCOUNT
@DDB
@DDELINK
@DEGREES
@DELTA
@DEVSQ
@DFRAC
@DISC
@DMAX
@DMIN
@DSTD
@DSTDS
@DSUM
@DURAT
@DURATION
@DVAR
@DVARS
@EMNTH
@ERF
@ERFC
@ERR

@EVEN
@EXACT
@EXP
@EXPONDIST
@FACT
@FACTDOUBLE
@FALSE
@FBDAY
@FDIST
@FEETBL
@FIB
@FILEEXISTS
@FIND
@FINV
@FIRSTBLANKPAGE
@FIRSTINGROUP
@FISHER
@FISHERINV
@FLOOR
@FORECAST
@FRACD
@FTEST
@FUTV
@FV
@FVAL
@GAMMADIST
@GAMMAINV
@GAMMALN
@GAMMAP
@GAMMAQ
@GCD
@GEOMEAN
@GESTEP
@GETGROUP
@HARMEAN
@HEXTOASC
@HEXTOBIN
@HEXTOBIN64
@HEXTONUM
@HEXTONUM64

@HEXTOOCT
@HEXTOOCT64
@HLOOKUP
@HOLS
@HOUR
@HYPGEOMDIST
@IF
@IMABS
@IMAGINARY
@IMARGUMENT
@IMCONJUGATE
@IMCOS
@IMDIV
@IMEXP
@IMLN
@IMLOG10
@IMLOG2
@IMPOWER
@IMPRODUCT
@IMREAL
@IMSIN
@IMSQRT
@IMSUB
@IMSUM
@INDEX
@INDEXTOLETTER
@INT
@INTERCEPT
@INTRATE
@INVB
@INVH
@IPAYMT
@IRATE
@IRR
@ISBDAY
@ISERR
@ISLEGALPAGENAME
@ISNA
@ISNUMBER
@ISSTRING

@KURT
@LARGEST
@LASTBLANKPAGE
@LASTINGROUP
@LBDAY
@LCM
@LEFT
@LENGTH
@LETTERTOINDEX
@LINTERP
@LN
@LOG
@LOGINV
@LOGNORMDIST
@LOWER
@LWKDAY
@MAX
@MDAYS
@MDET
@MDURATION
@MEDIAN
@MEMAVAIL
@MEMEMSAVAIL
@MID
@MIN
@MINUTE
@MNTHS
@MOD
@MODE
@MONTH
@MROUND
@MTGACC
@MULT
@MULTINOMIAL
@N
@NA
@NBDAY
@NEGBINOMDIST
@NETPV
@NORMDIST

@NORMINV
@NORMSDIST
@NORMSINV
@NOW
@NPER
@NPV
@NUMTOBIN
@NUMTOBIN64
@NUMTOHEX
@NUMTOHEX64
@NUMTOOCT
@NUMTOOCT64
@NWKDAY
@OCTTOBIN
@OCTTOHEX
@OCTTONUM
@ODD
@ODDFPRICE
@ODDFYIELD
@ODDLPRICE
@ODDLYIELD
@ORB
@ORH
@PAGEINDEX
@PAGEINDEX2
@PAGENAME
@PAGENAME2
@PAGENAMES
@PAGENAMES2
@PAYMT
@PBDAY
@PEARSON
@PERCENTILE
@PERCENTRANK
@PERMUT
@PI
@PIRATE
@PMT
@POISSON
@PPAYMT

@PRICE
@PRICEDISC
@PRICEMAT
@PROB
@PROPER
@PROPERTY
@PV
@PVAL
@QUARTILE
@QUOTIENT
@RADIANS
@RAND
@RANDBETWEEN
@RANK
@RATE
@RECEIVED
@REPEAT
@REPLACE
@RIGHT
@ROUND
@ROWS
@RSQ
@S
@SCMARG
@SECOND
@SERIESSUM
@SHEETS
@SHLB
@SHLBO
@SHLH
@SHLHO
@SHRB
@SHRBO
@SHRH
@SHRHO
@SIN
@SKEW
@SLN
@SLOPE
@SMALLEST

@SQRT
@SQRTPI
@STANDARDIZE
@STD
@STDS
@STEC
@STEYX
@STKOPT
@STRING
@SUBB
@SUBBO
@SUBH
@SUBHO
@SUM
@SUMPRODUCT
@SUMSQ
@SUMX2MY2
@SUMX2PY2
@SUMXMY2
@SUMXPY2
@SUMXY
@SUMXY2
@SYD
@TAN
@TBILLEQ
@TBILLPRICE
@TBILLYIELD
@TDIST
@TERM
@TIME
@TIMEVALUE
@TINV
@TODAY
@TRIM
@TRIMMEAN
@TRUE
@TTEST
@UPPER
@VALUE
@VAR

@VARS
@VERSION
@VLOOKUP
@WEIBULL
@WKDAY
@XORB
@XORH
@YDAYS
@YDIV
@YEAR
@YEARFRAC
@YIELD
@YIELDDISC
@YIELDMAT
@YLD2YLD
@ZTEST

See Also
@Functions and Formulas
Entering Formulas
@Function Arguments

System Menu

The System menu is common to all Windows applications. Commands are

Restore	Returns a window to its previous size and position
Move	Moves a window
Size	Changes window size
Minimize	Reduces a window to an icon
Maximize	Expands a window to fill the screen
Close	Closes a window
Next	Switches to the next window in the series of current windows
Test	Tests a custom application
Switch To	Switches to another window that you pick from a list

Help Error

To get help on this topic, click the Help Contents button above, then select the appropriate Help subject area.

Help Error No. 2

To get help on this topic, click the Help Contents button above, then select the appropriate Help subject area.

3-D block

A block that spans multiple pages.

@function

One of a set of special commands you can enter in a cell, either alone or in a formula. @Functions perform calculations and return the resulting value.

absolute cell reference

A cell reference in a formula that always references the same cell, even if the formula is copied to a different part of the notebook. To make a cell reference absolute, use the Abs key (F4) to insert dollar signs in its address--for example, $A$5. See also *relative cell reference*.

active cell

The cell affected by commands you choose. The *selector* distinguishes the active cell from other cells in the page.

active window

The window affected by commands you choose. In Quattro Pro, the active window is on top and its title bar and frame are the same color as the Quattro Pro application title bar.

address

The identifier for the location of a cell in a notebook. Within a spreadsheet page, a cell address is the letter of its column followed by the number of its row. For example, D5 is the address of the cell in column D and row 5. From one spreadsheet page to another within the same notebook, the address begins with the page letter or name. For example, B:D5 is the address of the cell in column D, row 5 of page B. From one notebook to another, the address begins with the notebook file name in brackets. For example, [EAST]B:D5 is the address of the cell in a notebook named East in page B, column D, row 5.

ANSI

The American National Standards Institute (ANSI) character set used in Windows products. ANSI characters can be translated into decimal and hexadecimal numbers.

argument

Information required by many @functions and macro commands. It supplies information to the command. For example, in @SUM(A4..A10), the block coordinates A4..A10 is the argument; it indicates what values to add together.

array

Rows and columns of data you can work with independently, not just as a whole block. @ARRAY is used in most array calculations.

arrow keys

Keys on the numeric keypad (usually on the right side of the keyboard) used to move the selector or insertion point. They are usually marked with arrows: Up, Down, Left, Right.

aspect ratio

The proportion of width to height of a graph. There are several ways to control a graph's aspect ratio, depending on where you want to display or print the graph.

autoload file

A file that opens automatically when you start Quattro Pro. You specify it with the Autoload File option of the application Startup property.

axis

Most graphs contain two axes: The *x-axis* usually runs horizontally along the bottom of the graph; the *y-axis* usually runs vertically on the left. (On rotated graphs, x-axis and y-axis positions are reversed.)

binary file

A file that contains instructions in your printer's native language for creating a printout. By default, binary files have the .PRN extension.

bitmap image

A picture in one of many graphic file formats (such as .BMP, .CGM, .CLP, or .EPS) that you can use to fill bars, backgrounds, or graphic elements.

bitmap object

A floating object containing an image in bitmap format that was pasted from another application.

block

Any rectangular group of cells, which can be selected by dragging the mouse. A block is identified in formulas by its *coordinates* or by its name.

border (graphic)

The outer edge of a *graphic element,* as opposed to its *fill.*

borders (notebook)

The lettered row at the top of the notebook page (A to IV) and the numbered column at the left (1 to 8192) used to identify cell addresses.

buttons

Rectangles that look like "push buttons" on the *SpeedBar* or in *dialog boxes*. Clicking a button performs a particular action. Most dialog boxes have at least three buttons: OK, Cancel, and Help.

You can also create *SpeedButtons,* which run a macro when clicked.

cascade

To arrange the open Quattro Pro windows in cascade formation, where the bottom window in the stack is nearest to the upper left corner of the screen. The command for this is Window|Cascade.

cell

A single unit of a spreadsheet, used to store a formula or data. A notebook is made up of many spreadsheet pages, each containing more than two million cells identified by row and column location.

cell address

The identifier for the location of a cell in a notebook. Within a spreadsheet page, a cell address is the letter of its column followed by the number of its row. For example, D5 is the address of the cell in column D and row 5. From one spreadsheet page to another within the same notebook, the address begins with the page letter or name. For example, B:D5 is the address of the cell in column D, row 5 of page B. From one notebook to another, the address begins with the notebook file name in brackets. For example, [EAST]B:D5 is the address of the cell in a notebook named East in page B, column D, row 5.

cell identifier

The left part of the *input line* that displays the active cell's address.

cell selector

The black-bordered rectangle that indicates the active cell; called "selector" in the documentation.

check box

A control in a dialog box that switches a feature on and off. It displays a mark when on. Click the check box to turn it on or off.

client application

The application receiving data from another through a *DDE* or *OLE* link. See also *server application.*

Clipboard

A temporary storage area used to hold data for transfer between locations or between Windows applications.

Clipboard commands

The Edit|Cut, Copy, and Paste commands. In Quattro Pro, you can choose these commands quickly by clicking the Cut, Copy, and Paste buttons in the *SpeedBar.*

command equivalents

Special commands that correspond to menu commands, used in macros.

command object

A collection of properties and actions that defines a menu command.

Consolidator

A Quattro Pro tool that combines source blocks using your choice of several statistical @functions, such as @SUM. To activate it, choose Tools|Consolidator.

constraint

A formula that defines a limit in a mathematical problem. For example, `+A4 > 50` is a constraint formula which states that the value in cell A4 must be greater than 50.

contiguous

A rectangular block of adjoining cells.

controls

The elements of a *dialog box* or *SpeedBar* that affect program actions: *buttons*, *check boxes*, and *edit fields* are examples of controls.

coordinates

The two addresses that define a block. The upper left and lower right addresses are always used. For example, the coordinates of the block stretching from column B,row 5 to column E, row 9 are B5..E9.

criteria table

The block in a spreadsheet that specifies what *field*(s) to search and what data to search for (the search criteria). Quattro Pro looks for all database records that match search criteria in the specified criteria table.

database

An organized collection of information. In Quattro Pro, a database is organized by rows, or *records,* of information, divided into separate columns, or *fields*.

database block

A Quattro Pro *database*, including *field* names in the first row and data *records* in the remaining rows. Each column is a data field.

data label

The label (or the part of the label) that states the value of the slice of a pie graph, or the section of a column graph. This is usually expressed as a percentage of the entire pie or column. The Data Label property in the pie or column graph Object Inspector controls the format of these labels, and has four options: Currency, Value, Percent, and None.

data point

A single cell value displayed in a graph.

data series

A block of data plotted as a group on a graph. Each value in a data series is represented by a bar in a bar graph, by a point on a line in a line graph, by a "slice" in a pie graph, and so forth. A data series is usually just called a "series," except when contrasted to a *label series*.

date/time serial number

A number assigned to a date or time, counting the number of days before and after December 30, 1899 (day 0), and the portion of the day that has elapsed. The integer portion is for the date; the fractional portion is used for time (with .00000=12:00 midnight, .5=12:00 noon, and .999999=11:59 p.m.).

DDE

Dynamic Data Exchange. A method of exchanging data between Windows applications. You can use it to set up dynamic *links* between applications so that if the data in one application changes, it also changes in the other.

default

A standard setting used "by default" when no other is specified. For example, Quattro Pro's default file extension is .WB1. You can change a default setting. You can also depart from the default setting for specified areas of the notebook (for example, you can change the width of an individual column).

destination block

The target block for an action, such as a copy or move operation.

dialog box

A type of window used to exchange information with Quattro Pro (and other Windows applications). Most dialog boxes have options you set to control application behavior.

dialog builder

A tool used to create dialog boxes. Choose Tools|Dialog Builder to access it.

dialog objects

Drawings and controls in dialog boxes.

dimmed

The state of a menu command when it is unavailable for choosing. The command name appears in dimmed (gray) type on the menu.

dithered

A color that looks like a checkerboard pattern.

DLL

Dynamic Link Library. Contains functions written in a programming language. You can use functions in DLLs as custom functions in a Quattro Pro for Windows spreadsheet.

DOS

The computer operating system required to run Windows and Quattro Pro.

Drag and Drop

The method of moving a selected block of cells by dragging it to a new location with the mouse. When you begin to drag the block, the mouse pointer changes to a hand and a colored outline appears. When you release the mouse button the data moves to wherever you've moved the colored outline. You can copy a block by holding down the Ctrl key while dragging.

drill

To enter data through several grouped pages at the same time. After creating a group and activating Group mode, you can select a cell and make an entry. Instead of pressing Enter, you can press Ctrl+Enter to drill the entry through all corresponding cells in the *group.*

drop-down list box

A dialog box area that consists of an *edit field* with a list. The list appears only when you click or press the down arrow at the right.

edit field

A dialog box field where you can enter data by typing or pointing.

ellipses (...)

Following a menu command or dialog box button name, these indicate that the command or button leads to a dialog box.

event

A user interface action that triggers a link command (for example, mouse clicks, keystrokes, values changing, and so on).

Experts

An alternate way to perform certain spreadsheet tasks; they help you use some features at a basic level with no learning required. To activate an Expert, choose Help|Experts or click the Experts button.

explode

To move a pie graph slice away from the center of the graph to emphasize it.

extended selection

A block selected by holding down the Shift key while clicking the opposite corner cell or while pressing arrow keys.

extension

A code of up to three letters attached to the end of a file name and separated from the rest of the name by a period. A file's extension often identifies the type of file it is. For example, Quattro Pro notebook files have the default extension .WB1.

field

A category of information in a database. In a Quattro Pro database, fields are set up as columns of information.

field name

The name used to identify a field. In Quattro Pro, field names are the headings of the columns of information.

file

A collection of data, stored under one name on a disk.

fill

The interior of a *graphic element,* as opposed to its *border.*

find string

In a search-and-replace operation, the group of characters to be found. You enter the find string in the Find *edit field*. See also *replacement string.*

floating graph

A graph that appears on a spreadsheet page. You can create a floating graph directly on a spreadsheet, or place an existing graph on a spreadsheet using Graph|Insert.

floating object

A rectangular object that overlaps cells. Quattro Pro for Windows has floating graphs, *SpeedButtons, OLE objects, bitmap objects*, and *picture objects.*

font

A set of type used to print or display text. In Quattro Pro, you can assign different fonts to blocks of a spreadsheet and to different parts of a graph. You can change the fonts by choosing a different typeface, point size, and style (bold, italics, and so on). For example, you can change the font of a block using the block Font property.

format line

A line used with Data|Parse to break text into separate cells. Symbols on the line tell Quattro Pro how to break down and interpret the data.

frequency distribution

The number of times numbers within given ranges are found in a block of data. To build a frequency distribution, choose Data|Frequency.

function

One of a set of special commands you can enter in a cell, either alone or in a formula. Quattro Pro documentation refers to these as "@functions." @Functions perform calculations and return the resulting value.

function keys

The keys labeled F1 through F10 or F12 at the left or top of the keyboard, used to perform special actions and commands.

graph

A visual representation of numerical information, often used to show trends and relationships in data.

graph button

A special type of text box that appears on a graph.You can click it to branch to another graph (using visual transition effects if desired), run a macro, or run a macro and then branch to another graph. The background area of a graph can also be programmed as a graph button. Graph buttons are active only when a graph is viewed in a slide show or with the Graph|View command.

graph objects

Graph parts that are selectable during editing. Graph objects include integral parts of a graph (for example, axes, legend, title, and bars or lines) plus drawn objects and text boxes added with the graph *SpeedBar* buttons. These drawn objects and text boxes are called *graphic elements* in the documentation.

graph pane

The area bordered by the x- and y- axes in a 2-D graph. Grid lines and the bars, lines, or areas that represent series, are drawn on top of this area.

graph type

The plotting style of a graph, such as line, bar, pie, 3-D ribbon, or text.

graph window

An area where you enhance graphs and change their *properties.* Graph windows have two properties of their own: Aspect Ratio and Grid options. Since each graph has its own unique graph window, you set properties for each window individually.

graphic element

Explanatory text or a drawn object added to a graph. Types of graphic elements include lines, arrows, polylines, polygons, rectangles, ellipses, freehand shapes, and *text boxes.*

Graphs page

The last page in a *notebook.* It contains icons representing the *graphs* and *dialog boxes* you've created. You can create and modify graphs, slide shows, and dialog boxes in the Graphs page.

To immediately display the Graphs page, choose the SpeedTab button at the bottom of the notebook window. Its icon is an arrow. Choose it again to return to the current notebook page.

grid

In a *spreadsheet* page, the lines that separate rows and columns.

In a *graph window,* the dotted lines spaced at regular intervals in the background. You can use the grid to line up elements in a graph window.

grid lines

Lines that separate cells of a spreadsheet. Also lines extending at regular intervals from axes across the graph frame. Graph grid lines help show the values represented by lines, bars, or areas on 2-D, 3-D, and combination graphs.

group

In a notebook window, a series of contiguous pages gathered together with Tools|Define Group. To activate the group, click the Group button (labeled G) at the bottom of the window; a blue line indicates that Group mode is on and the group is active. Changes made to one page in the group affect others while the Group mode is on.

In a graph window, the ability to select multiple drawn elements and tie them together into one group. You can then move, resize, or change properties of the group as a whole.

handles

Small black squares that appear on the border of an object when the object is selected. Drag a handle to resize the object. Drag within the selected object to move it.

headings

Descriptive label entries that appear at the top of columns or at the left of rows. You can specify that these entries be repeated on the left or top of each printed page.

hint

Text on the left of the status line that briefly describes the highlighted menu command.

hotspot

This term is specific to Help. It can mean either green-colored text or part of an illustration that, when clicked, takes you to a new topic or displays a popup window like the one this explanation is displayed in.

indicator

An uppercase word on the *status line* that indicates the current state, or *mode*, of the program. For example, when you press F2 to edit a cell's contents, the EDIT indicator appears.

input line

The line below the SpeedBar where Quattro Pro displays information about the active cell. In Edit mode, the input line shows the data you're entering or editing, and it contains checkmark (Enter) and X (Cancel) buttons.

insertion point

A flashing vertical bar indicating where text will be inserted as you type.

Interactive Tutors

Online lessons that help you learn Quattro Pro features using your own data. To use them, choose Help|Interactive Tutors or click the Tutors button in the Productivity Tools SpeedBar.

key

A column used to sort *records*. In Quattro Pro, you can sort records in a database by the entries in a given field, such as Date.

label

In a notebook cell, any entry that begins with a letter or a *label-prefix character.* You can type a label, or it can result from a string formula.

In a graph, entries of any type that you assign to the graph to define plotted values.

label-prefix character

A character preceding a label entry that indicates how to align the entry. A single quote (') left-aligns a label entry, a double quote (") right-aligns it, and a caret (^) centers it.

label series

A block of cells containing the labels that correspond to the values in a *data series*. These labels are placed over bars in a bar graph, or over data points in a line graph. In pie and column graphs, the label series is the same as the *x-axis series*, which places a text label next to the *data label* on each pie slice or column section.

legend

A key displayed beside or beneath a *graph* that specifies the colors, marker symbols, or fill for each *series* graphed.

legend series

A block of cells that provides the labels that indicate what each *data series* represents. In 2-D and some 3-D graphs, these labels appear beside the series markers in the legend box. In many 3-D graphs, legend labels appear along the *z-axis*. In multiple graphs, legend labels appear as the "titles" of the sub-graphs that represent each data series. Legend labels don't appear in pie and column graphs.

link

A reference to a cell or block in another *notebook,* or a *DDE* or *OLE* operation with another application. You can also create links from *dialog boxes* (described in *Building Spreadsheet Applications*).

For example, this notebook link refers to block D6..E7 on page C of notebook SALES: `[SALES]C:D6..E7.` If that block is named February, you could also enter the link as [SALES]C:February.

link command

A command attached to a dialog object that runs when an event occurs.

list box

Part of a dialog box that shows a list of choices.

locked titles

Columns or rows fixed onscreen at the left and/or above spreadsheet data to serve as identification *headings*. They remain onscreen when you scroll the spreadsheet page.

macro

A sequence of keystrokes or commands, stored in a notebook, that Quattro Pro can run to perform an action. Macros are useful whenever a long or complicated procedure is done frequently.

macro command

One of a set of special commands that can be used within *macros*.

macro library

A *notebook* used to store *macros.* When you run a macro, if it is not stored in the active notebook, Quattro Pro looks for it in the open macro library. You designate a notebook as a macro library with the notebook Macro Library property.

matrix

A rectangular array of numbers used to solve linear formulas and equations. Choose Tools|Advanced Math to invert and multiply matrices.

maximize

To enlarge the active window to the fullest possible size in the available *workspace.* Choose Maximize from the window control menu or click the up arrow in the upper right corner of the window.

menu

A list of items. Quattro Pro offers pull-down command menus accessed through the *menu bar*. To display a menu, click its title or press Alt plus the menu's *underlined letter*.

menu bar

The horizontal line just below the title bar of Quattro Pro for Windows that offers program command menus.

menu-equivalent commands

See *command equivalents*.

menu system

The arrangement of commands that appear at the top of the Quattro Pro window, and their subcommands and options. You can edit the Quattro Pro menu system.

menu system definition

A text file containing the information that makes up a menu system: menu titles, menu commands, their *shortcuts*, and their *underlined letters*. See also *menu file*.

minimize

To reduce the active window to an icon. Choose Minimize on the window control menu or click the down arrow in the upper right corner of the window.

mode

A state the spreadsheet can be in. Actions you take have different effects depending on the current mode. For example, pressing the arrow keys in READY mode moves the selector between cells; doing so in EDIT mode moves the insertion point in the entry. The current mode is shown by an *indicator* on the *status line*.

multiple selection

A series of blocks simultaneously selected by holding down the Ctrl key while dragging each block. Any action you perform will affect all the separate blocks. You may give the multiple selection a single name (see *named block*).

named block

A block of cells that has been assigned a name. You can reference such blocks by name instead of by coordinates.

noncontiguous block

A block consisting of more than one rectangular notebook area selected by holding down Ctrl while dragging each area. You can reference such a block in a formula as a list of area coordinates separated by commas or by a single name (see *named block*).

notebook

A Quattro Pro data file that is a collection of *pages*. The default file-name extension for a notebook is .WB1.

numeric format

The format in which Quattro Pro displays a value. To change it for a selected cell or block, display the block Object Inspector by right-clicking and choosing Block Properties, or by choosing Property| Current Object, then choosing Numeric Format.

object

Quattro Pro objects are blocks, pages, notebooks, graphs, graph objects, dialog box objects, floating objects, and the application itself. To modify an object, point to it and right-click the mouse (choose the Properties command if a *SpeedMenu* appears) or choose Property|Current Object.

object group

A set of objects that have been grouped using the Draw|Group command.

Object Help

Information on a screen object, such as a button on the SpeedBar. To display Object Help, point to an object, hold down Ctrl, then right-click the mouse.

object ID

The number attached to an object upon creation, enabling it to be manipulated by macros and link commands.

object inspection

Quattro Pro's innovative way to change the characteristics of objects. By right-clicking most objects and choosing the Properties command in a *SpeedMenu,* you display an *Object Inspector* and can change the setting for any of the *properties* listed there.

Object Inspector

The dialog box or list that appears when you right-click a selected object and choose the Properties command in a *SpeedMenu* (except for title bars and page tabs, which have no SpeedMenu) or choose one of the Property menu commands. Object Inspectors contain *properties* you can set to change the characteristics of objects.

OLE

Object Linking and Embedding. A method of displaying a "snapshot" of data from another Windows application.

OLE object

An object created in another Windows application that can be inserted into Quattro Pro for Windows as a floating object. Double-clicking an OLE object in Quattro Pro starts the application in which the object was created so you can use the object.

operating system

The base software your computer uses to control application programs such as Quattro Pro for Windows. Quattro Pro for DOS, Quattro Pro for Windows, and Windows require DOS (Disk Operating System) as a base in order to run.

operator

A mathematical symbol used in a formula to express a relationship between two values. For example, + and / are operators in the formula `+A6+10/B2`.

Optimizer

A Quattro Pro tool that lets you solve models for an optimal solution based on constraints. To use it, choose Tools|Optimizer.

orientation

The direction that Quattro Pro prints. Landscape orientation prints sideways across the length of the paper and Portrait orientation prints across the width of the paper (the default).

output block

In a database query operation, an area of the *notebook* where Quattro Pro copies records that match the specified *search criteria.* The output block must include the names of the *fields* you want copied for those records.

In a *parse* operation, an area where Quattro Pro copies the parsed data.

page

One layer of a *notebook*. Each notebook has 256 spreadsheet pages and one Graphs page.

page break

The place where one printed page ends and another begins. You can have soft page breaks (those that Quattro Pro calculates for you) and hard page breaks (those that you create yourself with Block| Insert Break).

pane

Part of a window. In Quattro Pro, you can break a notebook window into two panes and view different parts of the notebook in each.

parse

To split a column of long labels (such as those created by importing a text file) into two or more columns of data, usually by copying the data to a new destination. To do this, use Data|Parse.

paste

To place something from the Windows *Clipboard* into the active area.

picture object

A floating object containing an image in the metafile format that was pasted from another application.

play

To give control of an *OLE object* to the *server application* by double-clicking the object. The activity that begins when you play an object (such as playing a sound, displaying a slide show, or editing the object) is determined by the server application.

plotter

A printing device that uses colored ink jets or pens to print multicolored text and graphs.

pointer

The onscreen indicator (arrow, double-arrow, graph icon, and so on) that tracks mouse movement.

pointing

The act of indicating a cell, menu selection, or other object by moving the mouse pointer to it. Also the method of adding a block reference to a formula or to an *edit field* in a dialog box by dragging with the mouse or by moving the *selector* to its location.

precedence

The order in which *operators* are handled in a formula. For example, the multiplication operator (*) has a higher precedence than the addition operator (+); values are multiplied first, then added. Use parentheses to override default precedence. For example, in the following expression, the values will be added first, then multiplied: `7*(6+9)`.

primary notebook

A *notebook* containing a *link* to another notebook. See also *supporting notebook.*

properties

The characteristics of an *object*. For example, two of a cell's properties are Alignment and Numeric Format. To change the properties of most object, select it; then right-click and choose the Properties command in a *SpeedMenu,* or choose Property|Current Object. An *Object Inspector* appears, containing a list of the object's properties.

protection

A security measure that prevents the contents of an entire notebook page from being changed. Specific blocks within the page can then be unprotected.

query

To find specific information in a database. In Quattro Pro, you use Data|Query or Data|Table Query to look for and extract information in a database.

radio button

A type of *control* in a dialog box. Only one radio button can be chosen in a given group of radio buttons.

recalculation

The act of recomputing formulas after a change in a spreadsheet.

record

A set of information in a database. In a Quattro Pro for Windows database, records are rows of data.

regression analysis

A kind of analysis that shows how one set of variables is affected by one or more other sets of variables. To do a regression analysis, choose Tools|Advanced Math|Regression. For advanced regression analysis, choose Tools|Analysis Tools, then click the Advanced Regression tool.

relative cell reference

A cell reference Quattro Pro tracks by its position in relation to the cell containing the formula, not by its absolute address. When you move or copy a formula containing a relative cell reference, the reference is adjusted to reflect its new location. See also *absolute cell reference*.

replacement string

In a search-and-replace operation, the group of characters to replace the *find string*.

scale

The range of values assigned to an axis and used for plotting data on a graph. Quattro Pro scales each numeric axis to best display the data plotted on it. You can also adjust the scale manually.

scenario

A set of conditions and results displayed in the notebook as changing and result cells. You can create and save groups of scenarios with Tools|Scenario Manager or the Scenario Expert.

scroll bars

Bars along the right and lower right edges of a notebook window. Use these bars with a mouse to scroll the active notebook page: Drag the scroll box until the display is where you want it, click the scroll arrows, or click in the scroll bar itself. See also *tab scroller*.

search criteria

The specification of what *field*(s) in a database to search and what data to search for. The first row of the *criteria table* includes fields to use as search keys. The following rows may include exact values and strings or conditional formulas to match.

secondary y-axis

An optional vertical scale on the right side of standard (non-rotated) 2-D graphs. You can plot any series in 2-D bar, line, variance, and high-low graphs on a secondary y-axis by choosing this series option in the y-axis Object Inspector. When a graph has a secondary y-axis, the scale on the left side is called the *primary* y-axis.

select

To activate an object before working with it in some way. Most objects are selected by pointing with the mouse and clicking the left mouse button. To select a block, drag with the mouse from the upper left cell to the lower right cell.

selector

The bordered rectangle that indicates the *active cell*.

sensitivity table

A table that shows the results of varying one or two essential values in a formula. Also called a *what-if table*, because it shows what would happen to other figures if certain values change. To build one, choose Data|What-If.

series

A block of data plotted as a group on a graph. Each value in a series is represented by a bar in a bar graph, by a point on a line in a line graph, by a "slice" in a pie graph, and so forth.

series labels

A group of text labels, one of which is placed over each bar in a bar graph, or over each data point in a line graph. To add series labels to a graph, define a *Label Series* in the bar series or line series Object Inspector.

server application

The application sending data to another through a *DDE* or *OLE* link. See also *client application.*

shortcut

A key sequence assigned to a menu command that appears to the right of the command name in the menu. See also *underlined letter*.

slide show

A series of text and data *graphs* shown full screen in any sequence you choose. You can add transition effects and control the amount of time each graph stays visible.

sort key

A column used to sort *records*. In Quattro Pro, you can sort records in a database by the entries in a given field, such as Date. In other applications, sort keys are sometimes called "sort fields."

source block

The original location of data to be copied or moved to a *destination block*.

SpeedBar

The row of buttons and tools you can click just below the *menu bar.* The contents of the SpeedBar change according to the current activity. You can also create your own SpeedBars, as described in *Building Spreadsheet Applications.*

SpeedButton

A button you can create to run macros. You can use the SpeedButton tool in the SpeedBar to create SpeedButtons, which float in a notebook page. Clicking the SpeedButton runs the macro.

SpeedMenu

A menu that appears when you right-click most objects in Quattro Pro. For example, if you right-click the selected block, the SpeedMenu displays the active block Object Inspector and performs other block tasks such as Cut and Copy.

spillover text

Part of a long label displayed in empty cells to the right of where the label was entered.

spreadsheet

A *page* of a *notebook* organized by rows and columns into cells.

startup directory

The directory used by default for saving and loading files. The startup directory is the directory from which you started Quattro Pro unless you specify a different directory with the Directory option of the application Startup property.

status line

The bottom line of the Quattro Pro screen, showing CAP (if Caps Lock is on), OVR (if Overwrite mode is on), and the *mode* (such as READY or EDIT). It also shows menu *hints* as commands are highlighted.

style

A set of Block property settings gathered under a name. You use a style by choosing it from the Style list in the *SpeedBar*. You can create new styles with Edit|Define Style. Also, the properties of a graph that affect its appearance, such as text fonts, line styles, and fill colors.

supporting notebook

A notebook that is referenced by a *link* from another notebook. See also *primary notebook.*

syntax

The rules for the way formulas, @*functions*, and *macros* must be entered.

system notebook

A notebook that, when hidden, remains open when you choose File|Close All. Application developers can use system notebooks to hold application macros and ensure their availability while Quattro Pro is running. To make a notebook a system notebook, choose Property|Active Notebook|System.

tab

A marker you can click to move to the associated page of the notebook. Initially, tabs contain the letters of pages, but you can specify full page names with the page Name property. Tabs display at the bottom of the notebook window.

tab scroller

A small horizontal scroll bar to the left of the page *tabs* at the bottom of the notebook window. Move its slider to quickly view pages not currently displayed onscreen. To move directly to the Graphs page, choose the *SpeedTab* button, the button to the right of the page tabs.

template

A notebook file you can save to serve as a basis for other notebooks. You may want to use a template to save standard property settings or column headings, for example.

text box

A graphic design element that can contain word-wrapped text.

tick mark

A small line on a graph axis indicating a value.

tile

To divide the available space in the Quattro Pro workspace between all open windows. The command for this is Window|Tile.

title bar

The line at the top of a window that displays the name of the window. The title bar is a different color or intensity when a window is active.

titles

In a *spreadsheet* page, row or column headings that you can lock into view.

In a *graph window,* lines of explanatory text. You can have two major titles above the graph and one each along the *x-axis,* the *y-axis,* and the secondary y-axis.

twip

A unit of measurement equal to 1/1440th of an inch.

typeface

A design of a set of type. Times and Helvetica are examples of different typefaces. Typeface is one attribute of a *font*. Style (bold, italic, and so on) and point size are other attributes. Each combination of typeface, style, and size makes up a different font.

underlined letter

The character in a command or dialog box control that you can type to choose the command or control. In Copy in the Block menu, C is the underlined letter. To open a menu using its underlined letter, press Alt, then the letter. See also *shortcut.*

user interface (UI)

The system of interaction through which you communicate with the computer. In Quattro Pro, some elements of the UI are dialog boxes, pull-down menus, and the SpeedBar. You can define your own UI using techniques described in *Building Spreadsheet Applications*.

value

Any numeric entry in the notebook, entered either as a number, a date, or a formula that calculates a number.

value (of an object)

The current value of a dialog box control.

variable

Part of a formula that can have more than one value. For example, in the formula `+B3+8`, B3 is a variable whose value changes depending on what is entered in cell B3. The other part of this formula, 8, is a constant. Its value never changes.

what-if table

A table that shows the results of varying one or two essential values in a formula. Also called a *sensitivity table*, because it is sensitive to value changes. To build one, choose Data|What-If.

wildcard

Placeholder characters that can stand for one or more characters. In Quattro Pro, you can use wildcards in search conditions in a find string, when querying a database, when accessing files, and when linking notebooks. There are two wildcards: * represents any number of characters, and ? represents a single character.

workspace

The arrangement of windows and files in Quattro Pro, including the position and size of all windows, and the files contained in each. You can save the current workspace with File|Workspace|Save. Workspace files have a .WBS file-name extension.

x-axis

The horizontal line, usually at the bottom of a graph, used to plot values.

x-axis series

A block of cells that contains the labels that are placed at each division of the x-axis. (In XY graphs, the x-axis series contains data, instead of labels.) In pie and column graphs, the x-axis series places a text label next to the *data label* on each pie slice or column section.

y-axis

The vertical line, usually at the left of a graph, used to plot values.

z-axis

Present in many 3-D graphs, the z-axis projects from the right edge of the *x-axis* to the back wall of the graph. If a legend series is defined, the legend labels appear along this axis.

zoom factor

The enlargement or reduction of the display of all cells in a notebook. Zoom Factor is a Notebook property. Percentage settings greater than 100% enlarge cells, and settings under 100% reduce them. Zoom Factor settings only affect the screen display, not printed size.

AddressType

A number from 1 to 8 indicating which cell coordinates are relative or absolute

AppName

The DDE-server application to contact

Attribute

One or more of the attributes listed for @CELL

Block

A cell block reference or name

Block

A cell block (reference or name) containing cash flow information for the investment

Block

The 2-D cell block (reference or name) containing the database, including field names

Cell

A single cell address or a block name for a single-cell block

Code

A numeric value between 1 and 255

Col

The number of the referenced column, from 0 to 255 (the first column in Block = 0, the second = 1, and so on)

Column

A numeric value > 0.

Column

The number of the column containing the field for which you want to find the maximum or minimum value. The first column in Block is 0, second is 1, and so on.

Column

The number of the column containing the field for which you want to find the standard deviation. The first column in Block is 0, the second is 1, and so on.

Column

The number of the column containing the field for which you want to compute variance. The first column in Block is 0, second is 1, and so on.

Column

The number of the column containing the field you want to average. The first column in Block is 0, second is 1, and so on.

Column

The number of the column containing the field you want to count. The first column in Block is 0, second is 1, and so on.

Column

The number of the column containing the field you want to total. The first column in Block is 0, second is 1, and so on.

CommandEquivalent

A Quattro Pro for Windows command equivalent. To display a list, press Shift+F3 and choose Command Equivalents.

Cond

A logical expression representing the condition to be tested

Cost

A numeric value representing the amount paid for an asset

Criteria

A cell block containing search criteria

DataToReceive

The field, block, bookmark, or other information to receive from the application (DDE Item string)

DateString

A numeric or string value in any valid date format, enclosed by quotes (or block coordinates or a block name for a block that contains a date string)

DateTimeNumber

A numeric value between -109571 and 474816.9999999, representing a date/time serial number: -109571 = January 1, 1600; 0 = December 31, 1899; 474816 = December 31, 3199; the decimal = time (24 hr)

Day

A numeric value between 1 and 31

Decimal

Decimal number to convert

DecPlaces

A numeric value between 0 and 15

EndDate

A date serial number greater than StartDate

FalseExpr

A numeric or string value representing the value to use if Cond is false

FileName

Any file name

Fv

A numeric value representing the future value of an investment (the value the investment will reach at some point)

GeneralAction

A general menu category

Guess

A numeric value that estimates the internal rate of return on an investment

Hex

A hexadecimal number enclosed by double quotes.

Hr

A number between 0 and 23, representing Hour

Life

A numeric value representing the expected useful life of an asset (in years)

List

One or more numeric or string values, cell addresses, and block references or names, separated by commas

List

One or more numeric values, cell addresses, and block references or names, separated by commas

Min

A number between 0 and 59, representing Minute

Mo

A numeric value between 1 and 12

nCols

The number of columns in the data block (optional)

NewString

A string value, representing the characters to insert at position Num

Nper

A numeric value > 0, representing the number of periods of the loan (the number of payments to be made) or investment (the number of compounding periods)

nRows

The number of rows in the data block (optional)

nSheets

The number of pages in the data block (optional)

Num

A numeric value equal to or greater than 0

Num

A numeric value equal to or greater than 0, representing the number of characters to delete

Num

A numeric value between -15 and 15

Number

A positive integer equal to or less than the number of items in List - 1

Object

The name of the object whose property settings you are requesting

Page

The number of the referenced page, from 0 to 255 (the first page in Block = 0, the second = 1, and so on)

Per

The number of the loan period for which the principal is desired (where Nper is the total number of periods)

Period

A numeric value representing the time period for which you want to calculate depreciation

Pmt

A numeric value representing the amount of the periodic payment

Property

The property whose settings you are requesting

Pv

A numeric value representing the amount borrowed (the principal)

Pv

A numeric value representing the current value of an investment (the present value)

Rate

A numeric value > -1, representing the periodic interest rate (the fixed interest rate per compounding period)

Row

The number of the referenced row; if an offset, the first row in Block = 0, the second = 1, and so on.

Salvage

A numeric value representing the value of an asset at the end of its useful life

Sec

A number between 0 and 59, representing Second

SpecificAction

A menu item that requires setting

StartDate

A valid date serial number (see Help for @DATE)

StartNum

A numeric value equal to or greater than 0, representing the character position to begin with

StartNumber

A numeric value equal to or greater than 0, representing the character position to begin searching with

StartNumber

A numeric value equal to or greater than 0

String

A hexadecimal number enclosed by quotes

String

A valid string value, representing the text to operate on or search through

SubString

A valid string value, representing the value to search for

TimeString

A numeric value or a string value in any valid time format, enclosed by quotes

Topic

The table, spreadsheet, document, or other file in the DDE-server application from which to retrieve data

TrueExpr

A numeric or string value representing the value to use if Cond is true

Type

An optional numeric value that indicates whether payments or cash flows occur at the beginning (1) or the end (0) of the period; default = 0

X

A cell address or expression

x

A numeric or string value

x

A numeric value

x

A numeric value equal to or greater than 0

x

A numeric value equal to or less than 709

x

A numeric value > 0

x

A numeric value between -1 and 1

x

A numeric value representing degrees (0 to 360)

x

A numeric value representing radians

Y

A numeric value

Y

A numeric value equal to or greater than 0

Y

A numeric value not equal to 0

Yr

A numeric value between -300 and 1299; -300 = 1600, 0 = 1900, 1299 = 3199

Glossary

3-D block

-A-
@function
absolute cell reference
active cell
active window
address
ANSI
argument
array
arrow keys
aspect ratio
autoload file
axis

-B-
binary file
bitmap image
bitmap object
block
border (graphic)
borders (notebook)
buttons

-C-
cascade
cell
cell address
cell identifier
cell selector
check box
client application
Clipboard
Clipboard commands
command equivalents
command object
Consolidator
constraint
contiguous
controls
coordinates
criteria table

-D-
database
database block
data label
data point
data series
date/time number
DDE

default
destination block
dialog box
dialog objects
dimmed
dithered
DLL
DOS
Drag and Drop
drill
drop-down list box

-E-
edit field
ellipses (...)
event
Experts
explode
extended selection
extension

-F-
field
field name
file
fill
find string
floating graph
floating object
font
format line
frequency distribution
function
function keys

-G-
graph
graph button
graph objects
Graphs page
graph type
graph window
graphic element
grid
grid lines
group

-H-
handles
headings
hint
hotspot

-I-
indicator
input line
insertion point

play
plotter
pointer
pointing
precedence
primary notebook
properties
protection

-Q-
query

-R-
radio button
recalculation
record
regression analysis
relative cell reference
replacement string

-S-
scale
scenario
scroll bars
search criteria
secondary y-axis
select
selector
sensitivity table
series
series labels
server application
shortcut
slide show
sort key
source block
SpeedBar
SpeedButton
SpeedMenu
spillover text
spreadsheet
startup directory
status line
style
supporting notebook
syntax
system notebook

-T-
tab
tab scroller
template
text box
tick mark
tile
title bar

titles
tree definition
typeface

-U-
underlined letter
user interface (UI)

-V-
value
value (of an object)
variable

-W-
what-if table
wildcard
workspace

-X-
x-axis
x-axis series

-Y-
y-axis

-Z-
z-axis
zoom factor

3-DColName

Column name given to an additional column of data in Data Modeling Desktop if *SourceBlock* is a 3-D block; if passing a 2-D block, enter "" (two quotation marks).

AccessMode

R, M, W, or A

Align?

1 to apply the alignment; 0 otherwise

Alpha

The significance level at which to evaluate values for the F-statistic; the default is 0.05

Alpha

Significance level of the test; the default is 0.05

Annotations?

Whether to copy annotation objects: yes (1), no (0)

AppName

Name (in quotes) of the application to execute

AppName

Name of the Windows application to communicate with

Argument1,Argument2,...

Arguments to the @function

ArgumentList

List of one or more arguments to be passed to the specified subroutine (optional)

Arguments

Block to store the initial settings for controls in the dialog and their final settings

Arguments

List of one or more arguments to be passed to the specified macro command (optional)

AuditErrors?

Whether to highlight the source of error for each cell containing NA or ERR in the active notebook; 0 = no, 1 = yes

AutoWidth?

1 to automatically size the columns; 0 otherwise

BeforePage

New location for *SrcPages*

BinBlock

Set of numbers defining the bin ranges; *BinBlock* numbers must be in ascending order; if *BinBlock* is omitted, bins are distributed evenly from the minimum to the maximum values in *InBlock*, with the number of bins equal to the square root of the number of values in *InBlock*

BkgColor

New background color of the selected object(s)

Blanks

"NoBlanks" to omit blank cells when pasting; "" otherwise

Block

Block name or coordinates of the text and/or floating object to display in the message box

Block

Block to store data in

Block

Coordinates of the block(s) to display and/or select

Block

Coordinates of the block to sum, including blank cells for results

Block

Block containing rows or columns to hide, show, or resize

Block

Block to reformat

Blocks

Block or blocks to extract

Blocks

Block or blocks within *Filename* to combine (optional)

Block

A database block, including field labels and records

BorderColor

New border color of the selected object(s)

BoxType

New border style of the selected object(s)

BranchLocation

First cell of the macro to be executed in case of error

Bytes

Number of bytes of characters to read from a file

CandPctFees

Percentage fees ("points") for the candidate loan

CandRate

Annual interest rate for the candidate loan

Cell

Cell to store the property setting in

CellAtPointer?

Specifies which cell should be active when the pane switches (0 or 1, optional)

ChannelLoc

Cell to contain the Channel ID number of the DDE conversation if communication succeeds

Checked

Type "Yes" if the command should have a checkmark display by it (optional)

Cold|Hot

"Cold" to specify cold DDE links;"Hot" to specify hot links; if you specify hot links, changes made to one application are automatically reflected in the other

ColHead?

1 to apply the column heading format; 0 otherwise

ColTotal?

1 to apply the column total format; 0 otherwise

Column#

Number of columns into the specified block to store Value

Columns

A value indicating the number of random number sets to generate; default is the number of columns in *OutBlock*

Command

String containing a menu command definition

CompileFormulas?

Whether to compile formulas in the active notebook for faster calculation; 0 = no; 1 = yes

Compression?

Type of .TIF file compression to use: none (0), PackBits (1)

Condition

Condition to be met before recalculation is halted (optional)

Condition

A logical expression (or the address of a cell containing a label, value, or expression)

Confidence

Confidence level of the mean; the default is 0.95

Confidence

A value indicating the confidence level to apply to the regression

Contents

Type "Formulas" to paste formulas from Clipboard; "Values" to paste formula results from Clipboard; "" otherwise

CounterLoc

Cell used to track the number of macro iterations

Cum

Flag indicating whether to generate a column in *OutBlock* showing cumulative percentages: yes (1) or no (0); the default is 0

CurrBal

Remaining principal on the current loan

CurrRate

Annual interest rate on the current loan

Damping

Damping factor used as the exponential smoothing constant; indicates the percentage for error to adjust each prior forecast value; must be 0; the default is 0.3

Data

String or number to store in the active cell

Data?

Whether to copy graph data: yes (1), no (0)

DataToSend

Block of cells containing the information to send to the application

DataToReceive

Information to receive from the application

DestBlock

Block to store the data in

DateTimeNumber

The date and time at which macro execution can resume

DDEChannel

Channel ID number of the application to send information to

DDEChannel

Channel number of the DDE conversation to terminate

DDEChannel

DDE channel number of the application to receive data from

DDEChannel

Channel ID number of the application to execute a macro in

DDEChannel

Channel ID number of the application to communicate with

DependString

Areas in which the command is available (optional)

Dest

Cell you want data written to

DestBlock

New location for the block

DestBlock

Upper left cell of data returned from Data Modeling Desktop

DestGraph

New graph (the copy)

Destination

Location in the application that receives the sent information

DialogName

Name of dialog box to display

DialogName

Name of the SpeedBar to delete

DialogName

Name of a dialog box or predefined SpeedBar to add

Difference

Hypothetical mean difference; the default is 0

Dim1

Width of the left pane or height of the upper pane

Dim2

Width of the right pane or height of the lower pane

Distance

Distance in screens to scroll the active notebook horizontally

Distance

Distance in columns to scroll the active notebook horizontally

Distance

Number of rows to scroll the active notebook vertically

Distance

Number of screens to scroll the active notebook vertically

Distribution

A value indicating which random distribution to use: 1 = uniform, 2 = normal, 3 = Bernoulli, 4 = binomial, 5 = Poisson, 6 = patterned, 7 = discrete

DLLMacro

Name of the macro command to execute

DLLName

Name of the DLL (Dynamic Link Library) containing the macro command to execute

DLLName

Name of the DLL (Dynamic Link Library) file (if not already loaded) containing the function to execute

DLLName

The name of a DLL file to load

DoSave?

Whether to display a save prompt for modified files: no (0) or yes (1); 1 is the default

EndBal

Balance at loan completion; the default is $0

ErrorLocation

Cell in which to store the address of the cell containing the macro error (optional)

Extend?

Whether to extend the selection from the current selection to the specified block; 0 = no, 1 = yes; the default is 0

Filename

Name of the file to open, retrieve, combine, or import

Filename

Name of the binary file to send print jobs to

Filename

Name of the bitmap or other graphics file to import or export

Filename

Name of the new file containing *Blocks*

FilePosition

The number of bytes into a file to set the file pointer to

FillColor

New fill color of the selected object(s)

FillStyle

New fill style of the selected object(s)

FirstPane?

1 to affect rows or columns in left or top window pane; 0 to affect them in the right or bottom window pane

FmtName

Name of the format to apply

Font?

1 to apply the font; 0 otherwise

fPageOnly?

Whether to operate on only the active page of a group (1) or on all pages (0); only applies in Group mode

Format#

Optional format code

Format

Flag indicating what suffix and format to use for imaginary coefficient of complex number; the default is 1; 1 = x + yi, 2 = x + yj, 3 = x + iy, 4 = x + jy

Formulas?

1 to copy all source block cells containing formulas; 0 to not copy them; the default is 1

FromGraph

Graph containing the style, data, or annotation objects to copy

FunctionName

The name of an @function contained in the DLL

GraphName

Name of the graph to make current

GraphName1

Name of the first graph to display (optional)

GraphName2

Name of the second graph to display (optional)

GrayScale?

Whether to gray-scale: no (0), yes (1)

Grouped

"C" to group results by column or "R" to group results by row; the default is "C"

Hint

Help text to display on the status line when the command is highlighted (optional)

HorizFlip?

1 if the object or group is flipped horizontally from its previous position (optional)

HotKey

Shortcut key that chooses the command (optional)

InBlock

Input block containing two or more sets of numeric data arranged in columns or rows

InBlock

Input block containing two or more sets of numeric data arranged in columns; the first row must contains labels for each group; the first column must contain row labels indicating the beginning of each sample

InBlock

One or more numeric block values representing the input block

InBlock

Input block containing a single column or row with at least four numeric values; the block must not contain labels

InBlock

One or more numeric block values representing the input block; can be real or complex numbers; the number of values in *InBlock* must be a power of 2 between 2 and 1024 inclusive (for example, 2, 4, 8, 16,...); if the number of values in *InBlock* does not equal a power of 2, pad the block with additional zeros.

InBlock

Input block containing one or more columns or rows of numeric values; the block must not contain labels

InBlock

Input block containing one or more columns or rows of numeric values

InBlock

One or more numeric block values representing the input block, which contains a range of values and their probabilities, each in a separate column

InBlock1

The first input block containing a column or row of numeric values

InBlock2

The second input block containing a column or row of numeric values

InBlockX

Input block containing one or more columns of x values (the independent variables)

InBlockY

Input block containing a single column of y values (the dependent variables)

Interval

Number of values to include in the moving average; the default is 3

Inverse

0 to perform a Fourier transformation; 1 to perform the inverse Fourier transformation; the default is 0

Iteration#

Maximum number of times to recalculate *Location* trying to meet *Condition* (optional)

Key

Keyboard macro command (PGUP, DOWN, and so on)

Labels

1 if labels are located in the first column or row of the input block; 0 if the input block doesn't contain labels; the default is 0

Labels

1 if labels are located in the first column or row of *InBlockY* and *InBlockX*; 0 if the input blocks don't contain labels

Lambda

A parameter to the Poisson distribution representing the expected number of events in each unit

Largest

A value n which, if present, makes {DESCR} report the nth largest data point; if omitted, the largest data point isn't reported

LastYear

Last year through which the amortization period is generated; the default is equal to *Term* (the end of the loan); can be a fractional value to designate months (for example, 3+5/12)

Left

Screen column number (counting from 0) where the top left corner of the box should appear

LineDraw?

1 to apply the line drawing; 0 otherwise

Link

Action to perform when the command is chosen (optional)

LinkType

Format to use when pasting object in the Clipboard

Location

A cell or block name in which to store a typed or returned character

Location

Block to recalculate

Location

Block within which *Value* will be stored, either as a value or label, as specified by *Type*

Location

Cell in which to store the characters read

Location

Cell in which to store the argument being passed

Location

Cell in which to store the user's response

Location

A single cell containing the address or block name of another macro

Location

Any cell address or block name

Location

Block containing a custom Quattro Pro menu

Location

Block containing a menu block

Location

Cell in which to store the retrieved value

Location

Cell or block you want erased

Location

Location or name of another macro

LowerBound

A value indicating the lower bound on the set of numbers to generate

LowerCell

Cell containing the lower right corner of the floating object

MacroString

String containing the macro command(s) to execute

MacUse

1 if the user should manipulate the dialog, 0 if the macro manipulates it

Main

Main title of the graph

Mean

A value indicating the mean of the set of numbers to generate

MenuBlk

Block containing a menu definition

MenuDefinition

Block containing a menu system definition

MenuPath

Location in the menu system to insert a new menu item

MenuPath

Menu to delete

MenuPath

Location in the menu system to insert a new menu

MenuPath

Menu item to delete

MessageLocation

Cell in which to store any error message(optional)

ModelCopy?

Whether to use the Model Copy option; 0 = no, 1 = yes; the default is 0

Name

Name of the command to add

Name

Name of the graph to create, edit, or delete

Name

Name of the hidden window to show

NewHeight

The new height, in pixels, of the object or group

NewWidth

The new width, in pixels, of the object or group

Number

Number of times to repeat the operation (optional)

Number

Any positive integer or the address of a cell containing a positive integer (optional)

Number

Number of an open notebook window (1-9)

Number

1 to 10 (optional)

Number

1 through 8; 1 = pressing F4 once, 2 = pressing F4 twice, and so on (optional)

NumFmt?

1 to apply the numeric format; 0 otherwise

Object

Object to alter property of or to study

ObjectIDx

Identification number or name of the object(s) to select

ObjectName

Type of object to create

Objects?

1 to copy all SpeedButtons, floating graphs, and other objects in the source block; 0 to not copy them; the default is 1

ObjectType

The type of object to create or insert

OKExit?

Cell to store how the dialog box closed (1 for OK, 0 for Cancel)

OrigBal

Original loan balance; the default is $100,000

OutBlock

Upper left cell of the output block

Pareto

1 to arrange the output table in both descending frequency order and ascending *BinBlock* order; 0 to arrange the output table in ascending *BinBlock* order; the default is 0

Path

Path and wildcard specifying the list (optional)

Port

Port where the printer is connected

Printer

Printer to send print jobs to

PrintToFile

1 to send all print jobs to a binary file; 0 otherwise

Prob

A value indicating the probability of success on each trial run; must be 0 and 1

ProbOutBlock

Upper left cell of the output block for the probabilities table (allow at least two columns)

Prompt

String displayed to the user as a prompt

Property

String representing the property to change or study

Properties

Type "Properties" to paste properties from Clipboard; "" otherwise

Properties?

1 to copy all block properties applied to each source block cell; 0 to not copy them; the default is 1

Rate

Yearly interest rate; the default is 0.12

Rate

A value indicating a sampling rate; if *Method* = "P", *Rate* indicates the periodic interval used for sampling; if *Method* = "R", *Rate* indicates the number of samples

RemTerm

Remaining term on the current loan

RepeatNumber

A value indicating the number of times to repeat each value

RepeatSequence

A value indicating the number of times to repeat each sequence of values

ReplaceOption

Specifies what to do when the binary file already exists

ResidualOutBlock

Upper left cell of the output block for the residuals table (allow at least four columns)

Residuals

1 or 0; if 1, includes residuals in the output table

ResultLoc

A cell which contains the coded explanation of the operation's success (optional)

Row#

Number of rows into the specified block to store *Value*

Rows

A value indicating the number of rows of random numbers to generate for each one of *Columns* columns

Row/Col_Sizes?

1 to make the destination block rows and columns the same size as those in the source block; 0 to not change the size of the destination block rows and columns; the default is 1

Row or Col?

1 to reveal or hide a row; 0 to reveal or hide a column

RowHead?

1 to apply the row heading format; 0 otherwise

RowTotal?

1 to apply the row total format; 0 otherwise

SampleRows

The number of rows in each sample

SDev

A value indicating the standard deviation of the set of numbers to generate

Seed

Starting number for the random number generation algorithm

Set/Reset

0 to set the row height; 1 to reset the row height

Set/Reset/Auto

0 to set the column width; 1 to reset the column width; 2 to automatically size the column(s)

Setting

String representing the setting to apply to the property

Shading?

1 to apply the shading; 0 otherwise

Show?

1 to reveal rows or columns; 0 to hide rows or columns

Size

New height (in twips) if setting size (*Set/Reset* = 0); not needed if resetting size (*Set/Reset* = 1)

Size

New width (in twips) if *Set/...* = 0; not needed if *Set/...* = 1; extra characters (optional) if *Set/...* = 2

SlideEffect

Effect name exactly as used in the Light Table, enclosed in quotes

SlideName

Name of the next slide to display in the slide show

SlideShowName

Name of the slide show to run

SlideSpeed

Transition speed setting from 0 to 15 (0 is fastest)

SlideTime

Slide display time in seconds, from 0 to 3600

Smallest

A value n which, if present, makes {DESCR} report the nth smallest data point; if omitted, the smallest data point isn't reported 3

Source

Cell you want data copied from

SourceBlock

Block to copy, move, or transpose

SourceBlock

Cell block to send to Data Modeling Desktop

SrcPages

Range of pages to move

Start#

Initial value to place in *CounterLoc*

StartLoc

Cell containing the subroutine to execute

StdErrs

Flag indicating whether standard errors are included in the output table: yes (1) or no (0); the default is 0

StdResiduals

1 or 0; if 1, includes standardized residuals in the output table

Step#

Amount added to *CounterLoc* after each iteration

Step

Increment value between *LowerBound* and *UpperBound*

Stop#

Maximum value for *CounterLoc*

String

String of characters to be written into the open file as a single line

String

String of characters to be written into the open file

String

Any seven-character string

String

Any macro name for a key (such as PGUP, END, GRAPH, and so on) without braces, or a string that returns a key macro name without braces

String

Numbers or letters up to 1021

String

Numbers or letters to enter into cell as a comment

Style?

Whether to copy properties that affect the appearance of the graph: yes (1), no (0)

Sub

Title appearing below the main title of the graph

Subroutine

Name of the subroutine being called (which can be a block name or a cell address)

Summary

1 to display summary statistics; 0 to omit summary statistics; the default is 0

SumOutBlock

Upper left cell of the output block for the summary table (allow at least seven columns)

Synch?

Whether the panes are synchronized: yes (1) or no (0)

Term

Number of years in the loan; the default is 30 years; can be a fractional value to designate months (for example, 3+5/12)

Text

For SpeedButtons, the text to display on the button; for floating graphs, the name of the source graph. Text must appear in quotes.

TextColor?

1 to apply the text color; 0 otherwise

Time

Quattro Pro time serial number

Top

Screen line number (counting from 0) where the top left corner of the box should appear

Topic

Name of the file within the Windows application to communicate with

Transpose

Type "Transpose" to transpose data in the Clipboard and paste it; "" otherwise

Trials

A value indicating the number of trials

Type

Type of file to import: "ASCII Text File" or "Comma & """" Delimited File" or "Only Commas"

Type

String or value; string (or s) stores the value or formula as a label, and value (or v) stores the actual value or value resulting from a formula (optional)

Type

"P" to specify periodic sample; "R" to specify random sampling

UpperBound

A value indicating the upper bound on the set of numbers to generate

UpperCell

Cell containing the new upper left corner of the floating object

UpperCell

Cell containing the upper left corner of the floating object

UpperLeftX

Distance between the left side of the Quattro Pro window and the left side of active window, in pixels

UpperLeftY

Distance between the bottom of the input line and the top of active window, in pixels

UseCurrentBlock?

Whether to graph the current selected graph; 0 = no, 1 = yes; the default is 0

Value

Label or value to store

Value

New property setting or the Object:Property pair to copy the new setting from

Values?

1 to copy all source block cells containing values; 0 to not copy them; the default is 1

Variance1

A value indicating the variance of data set one; the default is 0

Variance2

A value indicating the variance of data set two; the default is 0

VertFlip?

1 if the object or group is flipped vertically from its previous position (optional)

Width#

Optional column width (1 to 1023)

Window

Dialog window to make active

WindowName

Name of the window to make active

WindowMode

Size state of the application's window: 1 or 101 for normal size, 2 or 102 for minimized, 3 or 103 for maximized; use 101, 102, or 103 to suspend macro execution until the application terminates

x,y

New position of the currently selected object(s), in pixels

x

The amount in pixels the lower right corner of the object or group should be placed away from the left edge of the window.

x

New window width, in pixels

X-Axis

Title of the graph's x-axis

x1,y1

xy coordinates for the starting point of the object in pixels; the upper left corner for rectangles and objects bounded by rectangles

x2,y2

xy coordinates for the end point or next point of the object in pixels; the width and height for rectangles and objects bounded by rectangles

x3,y3

xy coordinates for the next or last point of a polyline or polygon object

xoffset

Offset in pixels from the left edge of *UpperCell* to the left edge of the floating object

xoffset2

Offset in pixels from the left edge of *LowerCell* to the right edge of the floating object

y

The amount in pixels the lower right corner of the object or group should be placed from the top of the window.

Y

New window height, in pixels

Y-Axis

Title of the graph's y-axis

Y2-Axis

Title of the graph's secondary y-axis

YIntZero

1 if the y-intercept is 0 (the line of regression passes through the origin); 0 if the y-intercept isn't 0

yoffset

Offset in pixels from the top edge of *UpperCell* to the top edge of the floating object

yoffset2

Offset in pixels from the top edge of *LowerCell* to the bottom edge of the floating object

Special Keys

The following keys provide useful shortcuts for many Quattro Pro tasks:

Key	Description
File|New	Ctrl+N
File|Open	Ctrl+O
File|Close	Ctrl+W
File|Save	Ctrl+S
File|Quit	Ctrl+Q
Edit|Undo	Ctrl+Z
Edit|Cut	Ctrl+X
Edit|Copy	Ctrl+C
Edit|Paste	Ctrl+V
Edit|Clear Contents	Ctrl+B
Backspace	Erases the character to the left of the cursor. If pressed in a help screen, returns to the previous level of help.
Caps Lock	Enters Caps mode, in which all letters you type are displayed in uppercase letters. Press again to exit Caps mode.
Ctrl+Backspace	Clears an existing entry in a dialog box, or on the input line in Edit mode.
Ctrl+Break	Exits from a menu and returns to the notebook's Ready mode. If pressed while a macro is executing, terminates the macro.
Del	Erases the contents (but not properties) of the current cell or selected block. To delete contents and properties, use Edit|Clear.
Ctrl+Del	In Group mode, acts like Del, but "drills" the deletion through the group (deletes the selected cell, block, or object on all grouped pages).
/ (Slash)	Activates the active menu system, set with the Slash Key option of the Macro property in the Application Object Inspector.
\ (Backslash)	Lets you enter one or more characters to repeat across an entire cell width.
Enter	In the notebook, writes the entry on the input line into the current cell and returns to Ready mode. In a menu, choice list, or dialog box, chooses the highlighted item.
Esc	Cancels whatever you are doing. For example, it can back you out of a menu, exit a dialog box, or erase any changes you made to an entry on the input line.
Num Lock	Toggles performance of the numeric keypad keys. When you first press Num Lock, the NUM indicator displays and you can use the keypad for entering numbers. When you press it again you can use these keys as direction keys.
Pause	Enters macro Debug mode (works like Ctrl+F2).
Scroll Lock	Toggles performance of the arrow keys. When you first press Scroll Lock, the SCR indicator displays and the arrow keys scroll the contents of the active window without moving the selector. When you press it again you

	can move the selector with the arrow keys.
Ctrl+PgDn or PgUp	Moves from one page to the next or previous page; In an Object Inspector or dialog box with multiple settings, changes panes (groups of property options).
Ctrl+Shift+PgDn or PgUp	Selects multiple pages.
Up or Down	In a dialog box edit field, enters Point mode.
Tab or Ctrl Right	In the notebook, moves right one screen. In a dialog box, Tab moves between controls. In a graph window, Tab inserts a tab space in text boxes; use Ctrl+Tab to select the next object.
Shift+Tab or Ctrl Left	In the notebook, moves left one screen. In a dialog box, moves to the previous control. In a graph window, Shift+Tab moves to the start of the current word in a text box; use Ctrl+Shift+Tab to select the previous object.
Ctrl+Shift+D	Press this key combination before entering a date.
Ctrl+Shift+S	Displays a list of styles to apply to the selected block.

See Also
Mouse Techniques

Edit Mode Keys

The following keys are available when Quattro Pro is in Edit mode.

Key	Function
Esc	Exits Edit mode; if Startup|Compatible Keys is checked in the application Object Inspector, also erases the contents of the input line.
Enter	Enters the data and exits Edit mode.
Ctrl+Enter	In Group mode, acts like Enter, but "drills" data through all grouped pages at once.
Up	Enters the data, exits Edit mode, and moves up one cell. When the insertion point follows an operator in a formula, enters Point mode. If Startup|Compatible Keys is checked in the application Object Inspector, moves the insertion point up a line (with data wrapped on more than one line).
Down	Enters the data, exits Edit mode, and moves down one cell. When the insertion point follows an operator in a formula, enters Point mode. If Startup|Compatible Keys is checked in the application Object Inspector, moves the insertion point down a line (with data wrapped on more than one line).
PgDn	Enters the data, exits Edit mode, and moves down one screen. When the insertion point follows an operator in a formula, enters Point mode.
PgUp	Enters the data, exits Edit mode, and moves up one screen. When the insertion point follows an operator in a formula, enters Point mode.
Ins	Toggles between Insert and Overwrite modes. (Insert mode is the default.)
Backspace	Deletes characters to the left of the insertion point.
Del	Deletes characters to the right of the insertion point, or the selected cell block or graph object.
Ctrl+Backspace	Erases the contents of the input line.
Tab or Ctrl+Right	Moves five spaces to the right (or to the next word if Startup|Compatible Keys is checked in the application Object Inspector).
Shift+Tab or Ctrl+Left	Moves five spaces to the left (or to the previous word if Startup| Compatible Keys is checked in the application Object Inspector).
Ctrl+Shift+Right	Highlights five characters to the right of the insertion point.
Ctrl+Shift+Left	Highlights five characters to the left of the insertion point.
Ctrl+Shift+D	Press this key combination before entering a date.
F2	Toggles to display an indicator on the status line that tells you what type of data you're editing--either value or label.
F3	With the insertion point positioned after an operator, displays a list of block names.
Shift+F3	Displays a list of macro commands.
Alt+F3	Displays a list of @functions.
F9	Calculates and then displays result of formulas on the input line.

See Also
Mouse Techniques

Point Mode Keys

The following keys are available when Quattro Pro is in point mode.

Key	Function
Period key + block	To point to a block, move to one corner, press the period key to "anchor" the block, move to the opposite corner, then press Enter. To extend the selection from a different corner of the highlighted block, press the period key to move clockwise around the corners of the block.
Enter	Enters the highlighted block in the formula or dialog box and exits Point mode.
Esc	If pressed while pointing to a cell block, returns you to the cell that was current before you entered Point mode.
Backspace	If pressed while pointing out a cell block, returns you to the anchor cell.
F4	Makes the cell address to the left of the cursor absolute. Press repeatedly to cycle through the absolute combinations; for example, $A:$B$4, $A:B$4. (Does not disturb the position of the selector.)
Alt+F5	Toggles Group mode.
F6	If the window is split into two panes, jumps to the other pane.
Ctrl+F6	If multiple windows are open, jumps to the next window.

See Also
Mouse Techniques

Print Preview Keys

The following table lists keys you can use while previewing:

Key	Effect
Esc	Exits the preview.
F1	Displays online help.
PgUp	Displays the previous page.
PgDn	Displays the next page.
+	(plus key) Zooms in a level, increasing detail.
-	(minus key) Zooms out a level, decreasing detail.
Up arrow	Scrolls the zoomed display up.
Down arrow	Scrolls the zoomed display down.
Right arrow	Scrolls the zoomed display right.
Left arrow	Scrolls the zoomed display left.
Home	Displays the top left of a zoomed page.
End	Displays the bottom right of a zoomed page.

See Also

Mouse Techniques

Navigation Keys

The following keys allow you to move easily through Quattro Pro notebooks:

Key	Description
Left arrow	Moves left one cell.
Right arrow	Moves right one cell.
Up arrow	Moves up one cell.
Down arrow	Moves down one cell.
Ctrl+PgDn	Moves forward one page.
Ctrl+PgUp	Moves backward one page.
PgUp	Moves up one screen.
PgDn	Moves down one screen.
Home	Moves to upper left cell (A1) of the active page.
Ctrl+Home	Moves to upper left cell (A1) of the first page.
End	Must be used with another direction key.
End+Home	Moves to lower right corner of the non-blank part of the notebook page.
Ctrl+End+Home	Moves to the last nonblank cell in the notebook.
End + arrow key	If in a filled cell, the selector moves in the direction of the arrow to the next non-blank cell before an empty one; if the next cell in the direction of the arrow is empty, moves to the next filled cell in that direction.
Ctrl+Left arrow or Shift+Tab	Moves left one screen.
Ctrl+Right arrow or Tab	Moves right one screen.
F5 (Goto)	Moves to the cell you specify.

See Also
Mouse Techniques

Function Keys

The following list describes the use of each function key when pressed alone, or together with the Control or Shift keys.

F1	Activates the Help system.
F2	Activates Edit mode so you can change a cell entry.
Shift+F2	Activates Debug mode so you can execute a macro step by step.
Alt+F2	Displays the Tools|Macro|Execute dialog box.
F3	When you are prompted for a block, or in Edit mode with the insertion point positioned after an operator, press F3 to display a list of block names. To expand the list of block names to show their coordinates, press the Expand key (+). Press the Contract key (-) to remove coordinates. Press F3 again to "zoom" the names list to full screen or to shrink it back down.
Shift+F3	Displays a list of available macros, by category.
Alt+F3	In Value or Edit mode, displays a list of @functions.
Ctrl+F3	Lets you create a named block; the keyboard equivalent of Block|Names|Create.
F4	In Edit, Value, or Point mode, makes the cell address to the left of the cursor absolute. Press repeatedly to cycle through the eight absolute combinations; for example, $A: $B$4, $A:B$4. You can use F4 when entering or editing a formula. You can also use it in Point mode without disturbing the position of the selector.
Alt+F4	Exits Quattro Pro.
Ctrl+F4	Closes a window (dialog box, graph window, notebook, or dialog window).
F5	Moves the selector to a specified cell address.
Shift+F5	Works like the TurboTab button; displays the Graphs page, then returns to a spreadsheet page.
Alt+F5	Switches Group mode on and off.
F6	If the window is split into two panes, jumps to the other pane.
Ctrl+F6	Displays the next open window.
F7	Repeats the previous Query command.
Shift+F7	Press Shift+F7 and use the arrow keys to select a block of text in Ext mode.
Alt+F7	Lets you find and replace strings.
F8	Repeats the last Data|What-If command.
F9	In Ready mode, recalculates the notebook. In Edit mode, calculates and then displays result of formulas on the input line. For example, if you type 8*9 on the input line, then press F9, Quattro Pro replaces the formula with the result, 72. To recalculate the entire notebook afterwards, press F9 again in Ready mode.
F10	Works like / or Alt; moves the selector to the menu bar.
F11	Displays the current graph (equivalent to Graph|View). Press Esc to return to the notebook.
F12	Displays an Object Inspector for the selected object.
Shift+F12	Displays an Object Inspector for the active window.
Alt+F12	Displays the application Object Inspector.

See Also
Mouse Techniques

{ }

{ } does nothing; use it to reserve space or insert blank lines in the macro without stopping it. Quattro Pro continues running the macro command following { }.

See { ; } and {STEPON} for examples.

See Also

Using Macros

Macro Command Descriptions

{;String}

{ ; } lets you add explanatory remarks or comments to a macro. When Quattro Pro encounters this macro command, it skips over it.

{ ; } is a convenient way to temporarily hide other macro commands, such as a branch to an incomplete macro.

Example

This example runs the _int_update subroutine which calculates interest to date, then branches to the_print_inv subroutine to print the invoice. Notice how the comments embedded in the macro help make it easier to follow.

```
{; calculate interest to date}
{_int_update}
{  }
{; print the invoice}
{BRANCH _print_inv}
```

See Also

Using Macros

Macro Command Descriptions

{ ? }

{ ? } pauses macro execution and lets macro users press function keys, access menus, and choose commands until Enter is pressed. At that point the macro resumes execution.

Since this command gives users complete program control, use it with caution.

You must append a {CR} or ~ command after the { ? } to complete any user entry that requires Enter. For example, if the macro { ? } was used, and the user typed `67` then pressed Enter, the value would remain on the input line at the top of the application window even though the macro resumes. {CR} or ~ (or pressing Enter again) would then enter `67` into the cell.

Example

This macro selects cell E15, enters the label `Check Number?` in the cell to act as a prompt, then passes control to the keyboard so the user can enter a check number. The information the user enters replaces the Check Number? prompt.

```
{EditGoto E15}
{PUTCELL "Check Number?"}
{?}~
```

See Also

Using Macros

Macro Command Descriptions

{ABS <Number>}

{ABS} is equivalent to the Abs key, F4. The {ABS} macro converts relative cell addresses to absolute addresses. *Number* provides control over what part of the formula converts to an absolute address. `{ABS 1}` is equivalent to pressing F4 one time with the cell address selected; `{ABS 2}` is equivalent to pressing F4 twice, and so on, up to 8. If *Number* is missing, `{ABS 1}` is assumed.

{ABS} is most commonly used to reference a constant value that only appears in one place in a notebook but is copied in several formulas. The advantage of using absolute addresses is that formulas don't have to be edited if the absolute addresses are created before the formula is copied.

Example

The following example converts the formula in the active cell with a relative cell address to an absolute cell address. Then it copies the formula to a new location five cells below and five cells to the right of the active cell.

```
\A   {EDIT}{ABS}~
     {BlockCopy C(0)R(0),C(5)R(5)}
```

See Also

Using Macros

Macro Command Descriptions

{ACTIVATE WindowName}

You can find the name of a window on its title bar.

{ACTIVATE} makes the window specified by the string *WindowName* active. For example, to make the named graph PROFITS (in the notebook REPORT.WB1) active, use

```
{ACTIVATE "C:\SALES\REPORT.WB1:PROFITS"}
```

Use the same syntax for activating dialog windows. To make the notebook itself active, use

```
{ACTIVATE "C:\SALES\REPORT.WB1"}
```

See Also

{CHOOSE}

{WINDOW}

{ADDMENU MenuPath,MenuBlk}

{ADDMENU} lets you add menus to the active menu system. (Use {ADDMENUITEM} to add individual menu commands to the active menu system.) *MenuPath* is a string that specifies where the new menu should appear. For example, to insert a menu before the Block menu, use `/Block`; to insert a menu before the Copy command on the Block menu, use `/Block/Copy`. You can use <- and -> to place a menu at the top or bottom of a menu, respectively. For example, `/File/<-` specifies the last item on the File menu.

You can also use numbers to identify menu commands. For example, `/File/0` specifies the first item on the File menu (the ID numbers start at zero). When identifying a menu item with numbers, divider lines are considered menu items (for example, `/File/5` specifies the first divider line on the File menu, not Retrieve).

MenuBlk is a block containing a menu definition. The block must include all cells in the new menu.

Caution: Changes made to the menu system using this command aren't saved; they're lost when you exit Quattro Pro. Each time you run a macro containing {ADDMENU}, the menu changes appear again.

See Also

Building Menus
{DELETEMENU}
{MENUBRANCH}
{MENUCALL}

{ADDMENUITEM MenuPath, Name, <Link>, <Hint>, <HotKey>, <DependString>, <Checked>}

{ADDMENUITEM} is like {ADDMENU}, but inserts a single menu item before *MenuPath* instead of a new menu.

See the description of {ADDMENU} for the syntax of *MenuPath*. *Name* is the name of the new menu item; if a command, precede its underlined letter with an ampersand (&).

Link specifies the actions the menu command performs (for example, `"MACRO _remove_file"` runs the macro _remove_file). See Building Menus for more information on link commands and other menu parts.

Caution: Changes made to the menu system using this command aren't saved; they're lost when you exit Quattro Pro. Each time you run a macro containing {ADDMENUITEM}, the menu changes appear again.

Example

The following macro adds the menu command Erase to the top of the Block menu and places a divider line between it and Block|Move. Whenever Erase is chosen, the macro _erase_it runs.

```
\A   {ADDMENUITEM "/Block/<-","&Erase", "MACRO _erase_it"}
     {ADDMENUITEM "/Block/Move","--------"}
```

See Also

{DELETEMENUITEM}

{ALT+Key <Num>}

{ALT} emulates holding down the Alt key while pressing the key specified by Key. Uses of {ALT} that would perform Windows task-switching functions (like Alt+Tab and Alt+Esc) are ignored.

You can combine {ALT}, {CTRL}, and {SHIFT}. For example, `{CTRL+SHIFT+D}` is equivalent to pressing Ctrl+Shift+D, the Date keys.

Example

`{ALT+F}` emulates pressing Alt+F, which could display the File menu, or activate a control in a dialog box with an underlined letter of F.

See Also

Using Macros

Macro Command Descriptions

{ANOVA1 InBlock, OutBlock, <Grouped>, <Labels(0|1)>, <Alpha>}

{ANOVA1} performs a one-way analysis of variance. Use {ANOVA1} to test whether two or more samples come from the same population. {ANOVA1} is equivalent to the Anova: One-Way analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{ANOVA2 InBlock, OutBlock, SampleRows, <Alpha>}

{ANOVA2} performs a two-way analysis of variance, with more than one sample for each group of data. {ANOVA2} is equivalent to the Anova: Two-Way with Replication analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{ANOVA3 InBlock, OutBlock, <Labels(0|1)>, <Alpha>}

{ANOVA3} performs a two-way analysis of variance, with only one sample for each group of data. {ANOVA3} is equivalent to the Anova: Two-Way Without Replication analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{ANSIREAD #Bytes,Location}

{ANSIREAD} reads *#Bytes* bytes of characters from a file previously opened using {OPEN} (starting at the current position of the file pointer), and stores them as a label in *Location*, like {READ} but without any character mapping. This macro is provided for international users. See {READ}, for more information.

See Also

Using Macros

Macro Command Descriptions

{ANSIREADLN Location}

{ANSIREADLN} is like {ANSIREAD}, but instead of using a number of bytes to determine the amount of text to read, {ANSIREADLN} reads forward from the current file pointer location up to and including the carriage-return/linefeed at the end of the line, like {READLN} but without any character mapping. This macro is provided for international users. See {READLN}, for more information.

See Also

{WRITELN}

Using Macros

Macro Command Descriptions

{ANSIWRITE String,<String2>,<String3,...>}

{ANSIWRITE} copies *String* to a file opened with the {OPEN} command, starting at the location of the file pointer, like {WRITE} but without any character mapping. This macro is provided for international users. See {WRITE}, for more information.

See Also

Using Macros

Macro Command Descriptions

{ANSIWRITELN String,<String2>,<String3,...>}

{ANSIWRITELN} copies *String*(s) to a file opened with {OPEN}, starting at the location of the file pointer, and ends the string(s) with the carriage-return and linefeed characters, like {WRITELN} but without any character mapping. This macro is provided for international users. See {WRITELN}, for more information.

See Also
{WRITE}

{BACKSPACE <Number>} and {BS <Number>}

{BACKSPACE} and {BS} are equivalent to the Backspace key, which deletes one character to the left of the insertion point in Edit mode. The optional argument *Number* specifies how many times to repeat the operation; for example, `{BACKSPACE 2}` is equivalent to pressing Backspace twice.

See Also

Using Macros

Macro Command Descriptions

{BACKTAB <Number>}

{BACKTAB} is equivalent to the Ctrl+Left Arrow or Shift+Tab key. It's the same as {BIGLEFT}, which selects the leftmost cell of the screen to the left of the current one. The optional argument *Number* specifies how many times to repeat the operation; for example, `{BACKTAB 3}` is equivalent to pressing Shift+Tab three times.

See Also

Using Macros

Macro Command Descriptions

{BEEP <Number>}

{BEEP} sounds the computer's built-in speaker.

Number dictates the tone of the beep. If *Number* is omitted, {BEEP 1} sounds. If *Number* is larger than 10, the pattern repeats; for example, {BEEP 11} is the same as {BEEP 1}.

Use {BEEP} to catch the user's attention. You can use it in interactive macros to introduce a prompt for information or to indicate a macro has finished.

Example

The following macro checks a block named error_check for an error condition (indicated by error_check containing zero). If there is no error, it branches to a macro called _continue, which carries on the previous procedure. If there is an error, it gives a low beep, then a medium beep, and moves the selector to the block called message_area, where an error message is stored.

```
{IF error_check = 0}{BRANCH _continue}
{BEEP 1}{BEEP 5}{EditGoto message_area}
```

See Also

Using Macros

Macro Command Descriptions

{BIGLEFT <Number>}

{BIGLEFT} is equivalent to the Ctrl+Left Arrow or Shift+Tab key. It's the same as {BACKTAB}, which selects the leftmost cell of the screen that's to the left of the current one. The optional argument *Number* specifies how many times to repeat the operation; for example, `{BIGLEFT 2}` is equivalent to pressing Shift+Tab twice.

See Also

Using Macros

Macro Command Descriptions

{BIGRIGHT <Number>}

{BIGRIGHT} is equivalent to the Ctrl+Right Arrow or Tab key. It's the same as {TAB} and selects the leftmost cell of the screen that's to the right of the current one. The optional argument *Number* specifies how many times to repeat the operation; for example, `{BIGRIGHT 2}` is equivalent to pressing Tab twice.

See Also

Using Macros

Macro Command Descriptions

{BLANK Location}

{BLANK} erases the contents of the block referred to as *Location*. You can also use the command equivalent {EditClear} to erase the contents of the currently selected block.

Example

This macro erases the block named part_list:

```
\F   {BLANK part_list}
```

See Also

Using Macros

Macro Command Descriptions

{BRANCH Location}

{BRANCH} runs the macro stored in *Location*. If *Location* references a block, Quattro Pro starts with the macro command in the top left cell.

{BRANCH} can change the flow of execution based on a condition test. For example, you can use it to run a different macro depending on the contents of a certain cell.

{BRANCH} is like {*Subroutine*} in that it passes control to another macro. Unlike {*Subroutine*}, it doesn't hold your place in the original macro, waiting for control to return. Use {BRANCH} when you don't intend to return to the original macro. To run another macro and then return to the calling macro, use {*Subroutine*}.

Example

The following macro branches to a macro named _high if the value in cell D10 is greater than 1000; otherwise, it continues to run the same macro:

```
{IF D10 > 1000}{BRANCH _high}
```

See Also

Using Macros

Macro Command Descriptions

{BREAK}

{BREAK} clears any displayed dialog boxes or prompts and returns Quattro Pro to Ready mode. It doesn't stop macro execution; use {QUIT} for that operation.

See Also

Using Macros

Macro Command Descriptions

{BREAKOFF}

{BREAKOFF} disables Ctrl+Break, which can be used to end a macro before it's done. After {BREAKOFF}, the user won't be able to exit a macro until the end of the macro or until {BREAKON} is used.

Caution: Use {BREAKOFF} only when necessary. Without access to Ctrl+Break, the only way to stop a "runaway" macro is to reboot or turn off the computer.

Example

This macro disables Ctrl+Break while the user inputs a name:

```
{PUTCELL "Enter your name here:"}
{BREAKOFF}
{?}~
{BREAKON}
{PUTCELL "Try again: "}
{?}~
```

See Also

Using Macros

Macro Command Descriptions

{BREAKON}

{BREAKON} enables Ctrl+Break after a previous {BREAKOFF} command has disabled it.

{BREAKON} should be used as soon as possible after {BREAKOFF}, because with Ctrl+Break disabled, the only way to halt a "runaway" macro is to reboot or turn off the computer.

See {BREAKOFF} for an example of {BREAKOFF} and {BREAKON}.

See Also

Using Macros

Macro Command Descriptions

{CALC}

{CALC} is equivalent to the Calc key, F9, which recalculates the active notebook, or converts the formula on the input line into its result when editing a cell.

See Also

Using Macros

Macro Command Descriptions

{CAPOFF} and {CAPON}

{CAPOFF} and {CAPON} are equivalent to Caps Lock off and Caps Lock on, respectively.

See Also
Using Macros
Macro Command Descriptions

{CHOOSE}

{CHOOSE} displays a pick list of open windows. Your choice becomes the active window.

See Also
{ACTIVATE}
Using Macros
Macro Command Descriptions

{CLEAR}

{CLEAR} is the equivalent of Ctrl+Backspace, which erases any previous entry in a prompt line or on the input line in Edit mode. This command is useful when loading or retrieving files.

See Also

Using Macros

Macro Command Descriptions

{CLOSE}

{CLOSE} ends access to a file previously opened using {OPEN}. This lets another file be opened (only one can be open at a time). {CLOSE} completes the process of writing information to a file, including an update of the disk directory. This step is crucial to the integrity of any file. If your computer is turned off before a file is closed, that file's contents may become corrupted or lost.

{CLOSE} fails in the event of a disk error, such as when a disk is removed from the disk drive before the file is closed. In this case, {ONERROR} is useful in intercepting the error. If {CLOSE} succeeds, macro execution continues in the cell below the cell containing the {CLOSE} command, ignoring any other commands in that cell. If {CLOSE} fails, macro execution continues in the same cell as the {CLOSE} command.

Example

The following macro opens a new file in drive A called AFILE, writes the text line `Hello, world!` to the file, and closes the file.

```
\F   {OPEN "A:\AFILE",W}
     {WRITELN "Hello, world!"}
     {CLOSE}
```

See Also

Using Macros

Macro Command Descriptions

{COLUMNWIDTH Block, FirstPane?, Set/Reset/Auto, Size}

{COLUMNWIDTH} provides three ways to change the width of a column or block of columns (it is equivalent to the block property Column Width). The columns to change are specified by *Block*. *FirstPane?* is used when the active window is split into panes (using Window|Panes). To resize the columns in the left or top pane, set *FirstPane?* to 1; to resize the columns in the right or bottom pane, set *FirstPane?* to 0.

The argument *Set/Resize/Auto* specifies how to change the width. To set a column width, use this syntax: {COLUMNWIDTH *Block, FirstPane?,* 0*, NewSize*}.

NewSize is the new column width, in twips (a twip is 1/1440th of an inch). The maximum width is 20 inches (28,800 twips).

To reset a column to the default width (set by Default Width in the page Object Inspector) use this syntax: {COLUMNWIDTH *Block, FirstPane?*, 1}.

To automatically size a column based on what's entered in it, use this syntax: {COLUMNWIDTH *Block, FirstPane?*, 2, *ExtraCharacters*}

ExtraCharacters is the number of characters to add on to the calculated width. If this argument is omitted, the default is used (1 character).

Examples

`{COLUMNWIDTH A:A..B,1,0,1440}` sets the width of columns A and B (on page A) to one inch (1,440 twips).

`{COLUMNWIDTH A:A..B,0,0,2160}` sets the width of columns A and B (on page A) to one and a half inches (2,160 twips). If the window is split, the columns are resized in the left or top pane.

`{COLUMNWIDTH A:C,1,1}` resets the width of column C (on page A) to the default width.

`{COLUMNWIDTH A:C,1,2,3}` automatically sizes column C (on page A) and adds three characters to the calculated width.

See Also

Column Width Property

{ROWHEIGHT}

{ROWCOLSHOW}

Using Macros

Macro Command Descriptions

{CONTENTS Dest, Source, <Width#>, <Format#>}

{CONTENTS} copies the contents of *Source* into *Dest*, but unlike {LET} or other copy commands, if *Source* contains a value entry, it translates the copied value into a label and stores it in *Dest*. It also lets you specify a different numeric format and column width using the *Width#* and *Format#* arguments.

Width# can be any number from 1 to 1023. Quattro Pro won't alter the width of the destination column but will treat the resulting string as if it came from a column with the specified width. For example, if a value is displayed as ***** in the source column because the column isn't wide enough, specifying a wider *Width#* will let the value be copied as it would be displayed within that width, not as *****. *Width#* is optional, but must be provided if *Format#* is used. If you don't specify *Width#*, the width of the source column is assumed. Use the maximum width if you want all values to come across properly. You can use @TRIM with a {LET} command to remove any leading spaces from the label.

Format# can be any number from 0 to 127. Each number in this range corresponds to a specific numeric format and decimal precision. *Format#* affects the *Dest* entry only, not the *Source* value. See Numeric Format Codes for a list of special codes used to indicate numeric formats with *Format#*.

Examples

The following examples assume cell C18 contains the value 48,988 in comma format with a column width of 12.

`{CONTENTS A18,C18}`	places the 12-character label ' 48,988 in cell A18 (six spaces are inserted at the beginning).
`{CONTENTS E10,C18,3}`	places the 3-character label '*** in cell E10. (Only asterisks are copied because the value doesn't fit within three spaces.)
`{CONTENTS A5,C18,15,34}`	places the 15-character label ' $48,988.00 in cell A5 (five spaces are inserted at the beginning).

See Also

Using Macros

Macro Command Descriptions

Numeric Format Codes

Code	Description
0-15	Fixed (0-15 decimals)
16-31	Scientific (0-15 decimals)
32-47	Currency (0-15 decimals)
48-63	% (percent; 0-15 decimals)
64-79	, (comma; 0-15 decimals)
112	+/- (bar graph)
113	General
114	Date [1] (DD-MMM-YYYY)
115	Date [2] (DD-MMM)
116	Date [3] (MMM-YYYY)
117	Text
118	Hidden
119	Time [1] (HH:MM:SS AM/PM)
120	Time [2] (HH:MM AM/PM)
121	Date [4] (Long International)
122	Date [5] (Short International)
123	Time [3] (Long International)
124	Time [4] (Short International)
127	Default (set with Normal style)

{CR} or ~

{CR} or ~ (tilde) are equivalent to the Enter key.

See Also
Using Macros
Macro Command Descriptions

{CREATEOBJECT ObjectType, x1, y1, x2, y2<, x3, y3, ...>}

With {CREATEOBJECT} you can add objects to the active window normally added using the SpeedBar. {CREATEOBJECT} is context-sensitive, letting you create lines in a graph window or check boxes in a dialog window. Quattro Pro interprets the coordinates specified after *ObjectType* differently based on the object type. The following table lists the possible graph object settings for *ObjectType*, and how each graph object uses the (x,y) coordinates.

Graph Objects {CREATEOBJECT} Can Generate

Object	# of (x,y)'s	Coordinates
Line	2	1st: Start point, 2nd: End point
Arrow		*(same as for Line)*
Rect (Rectangle)	2	1st: Upper left corner, 2nd: Rectangle width and height (in pixels)
Ellipse	2	1st: Upper left corner of a rectangle bounding the ellipse; 2nd: Width and height of the bounding rectangle
Rounded_Rect		*(same as for Rectangle)*
Text		*(same as for Rectangle)*
Polyline	Varies	1st: Start point, 2nd: End point of first segment and start of second segment; 3rd: End point of second segment and start of third segment, ... *n*th: End point
Polygon		*(same as for Polyline)*
Freehand_Polyline		*(same as for Polyline)*
Freehand_Polygon		*(same as for Polyline)*

Dialog Controls {CREATEOBJECT} Can Generate

You can create these dialog controls listed in the order they appear on the Dialog SpeedBar: Button, CheckBox, RadioButton, BitmapButton, Label, EditField, SpinCtrl, Rectangle, GroupBox, RangeBox, ComboBox, PickList, FileCtrl, ColCtrl, ScrollBar, HScrollBar, TimeCtrl. When creating a control, x1 and y1 specify the upper-left corner of the control; x2 and y2 specify the width and height of the control, in pixels.

ObjectType is enclosed in quotes. The x and y coordinates for each point follow, separated by commas.

Examples

`{CREATEOBJECT "Rect",86,11,94,74}` creates a rectangle with upper-left corner = (86,11), width = 94, and height =74 (pixels).

`{CREATEOBJECT "Line",260,238,356,228}` creates a line that starts at (260,238) and ends at (356,228).

`{CREATEOBJECT "Polyline",2,2,23,59,11,26}` creates a polyline that starts at (2,2) draws a line to (23,59), and then draws a line from that point to (11,26).

See Also
{MOVETO}
{RESIZE}

{CTRL+Key <Num>}

{CTRL} emulates holding down the Ctrl key while pressing a keystroke. For example, `{CTRL+PGDN 5}` emulates pressing Ctrl+PgDn five times to move down five notebook pages. Keystrokes that would perform Windows task switching, such as Ctrl+Esc, are ignored.

You can combine {ALT}, {CTRL}, and {SHIFT}. For example, `{CTRL+SHIFT+D}` is equivalent to pressing Ctrl+Shift+D, the Date keys.

See Also
Using Macros
Macro Command Descriptions

{DATAMODEL SourceBlock, DestBlock, Cold|Hot, 3-DColName}

{DATAMODEL} is the command equivalent for Data|Data Modeling Desktop. It lets you quickly view data from different perspectives and arrange the data for crosstabular reports.

See Also

Data|Data Modeling Desktop

Using Macros

Macro Command Descriptions

{DATE}

{DATE} is equivalent to pressing Ctrl+Shift+D, which lets users enter a date or time into the active cell.

Examples

`{DATE}8/6/90~` enters 8/6/90 in the active cell as a date.

`{DATE}{?}~` pauses to let the user enter a date, then enters that date into the active cell.

See Also

Using Macros

Macro Command Descriptions

{DEFINE Location1<:Type1>, Location2<:Type2>,...}

When you pass control to a subroutine with the {Subroutine} command, you can also pass arguments for use by that subroutine. If you do, you must include a {DEFINE} command in the subroutine's first line. This command defines the data type of each argument passed and indicates which cells to store the arguments in. If no {DEFINE} command is included, the arguments are ignored.

{DEFINE} sequentially defines the arguments passed to the subroutine. The first location and type given are assigned to the first argument passed, the second location and type to the second argument, and so on.

You must specify a location for each argument. This tells Quattro Pro where to copy them. If there are more locations given than arguments passed, or more arguments than locations, the macro ends immediately, and an error message displays.

Type is optional. It tells Quattro Pro whether the argument is a value or string. If no data type is given, the argument is assumed to be a literal string (even if it is a valid block name, cell address, or value). If you add `:string` (or `:s`) to the location, any argument passed is stored as a label. If you add `:value` (or `:v`), Quattro Pro treats the coming argument as a number or value resulting from a numeric formula. It it isn't a numeric value, Quattro Pro treats it as a string (or string value from a formula).

Example

In the following example, the \F macro passes three arguments (principal, interest, and term) to the subroutine _calc_loan, which stores the arguments in named blocks and defines them as values. It then

uses the arguments to calculate the monthly payment on a loan

stores the result in a cell named amount

creates a label in a cell named payment displaying that amount as currency

returns control to the main macro

The main macro (\F) displays the result in the active cell preceded by the string "The monthly payment will be ".

```
\F          {_calc_loan 79500,12%,30}

_calc_loan  {DEFINE prin:value,int:value,term:value}
            {LET amount,@PMT(prin,int/12,term*12)}
            {CONTENTS payment,amount,9,34}{EditGoto txt_area}{RETURN}

prin        79500
int         0.12
term        30
amount      817.747
payment     $817.75

txt_area    +"The monthly payment will be "&@TRIM(payment)
```

See Also

Using Macros

Macro Command Descriptions

{DEL} and {DELETE}

{DEL} and {DELETE} are equivalent to the Del key.

See Also
Using Macros
Macro Command Descriptions

{DELETEMENU MenuPath}

{DELETEMENU} removes the menu specified by *MenuPath* from the menu system. See the description of {ADDMENU} for the syntax of *MenuPath*. Use {DELETEMENUITEM} to remove an individual menu command.

Caution: Changes made to the menu system using this command aren't saved; they're lost when you exit Quattro Pro. Each time you run a macro containing {DELETEMENU}, the menu changes appear again.

Example

`{DELETEMENU "/File"}` removes the File menu from the active menu system.

See Also

Building Menus

Using Macros

Macro Command Descriptions

{DELETEMENUITEM MenuPath}

{DELETEMENUITEM} removes the menu command specified by *MenuPath* from the menu system. Use {DELETEMENU} to remove entire menus from the active menu system. See {ADDMENU} for the syntax of *MenuPath*.

Caution: Changes made to the menu system using this command aren't saved; they're lost when you exit Quattro Pro. Each time you run a macro containing {DELETEMENUITEM} the menu changes appear again.

Examples

`{DELETEMENUITEM "/Edit/Clear"}` removes the Edit|Clear command.

`{DELETEMENUITEM "/Edit/<-"}` removes the first item on the Edit menu.

See Also

Building Menus

{ADDMENUITEM}

{DESCR InBlock, OutBlock, Grouped, <Labels(0|1)>, <Summary(0|1)>, <Largest>, <Smallest>, <Confidence>}

{DESCR} returns a table of descriptive statistics that characterize a sample. {DESCR} is equivalent to the Descriptive Statistics analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{DISPATCH Location}

{DISPATCH} is similar to {BRANCH}, except it uses *Location* differently:

If *Location* is a cell address or one cell named block, {DISPATCH} branches to the address stored in that cell.

If *Location* is a text formula resulting in a cell or one cell named block, {DISPATCH} branches to that result address.

If *Location* is a multi-celled block or a text formula resulting in a multi-celled block, {DISPATCH} branches to the first cell of that block.

{DISPATCH} is useful when you want to branch to one of several alternative macros, depending on circumstances. You can also set up a macro that modifies the contents of *Location*, depending on user input or test conditions.

Example

The macro _continue in the following example uses {DISPATCH jump_cell} to create a form letter that changes depending on the result of a credit check.

Type the line to the right of \F with no hard returns until after the first {BRANCH _continue}, then press Enter to insert the text into one cell.

```
credit        OK

\F            {EditGoto text_area}
              {IF credit="OK"}{LET jump_cell,"_ok_note"}{BRANCH _continue}
              {LET jump_cell,"_bum_note"}{BRANCH _continue}

_continue     {PUTCELL "Because of your current standing with P.K."}
              {DOWN}
              {PUTCELL "Finance, we have decided to"}{DOWN}
              {DISPATCH +jump_cell}

jump_cell     _ok_note

_ok_note      {PUTCELL "extend your credit limit."}{DOWN 2}
              {BRANCH _finish}

_bum_note     {PUTCELL "CANCEL your account."}{DOWN 2}
              {BRANCH _finish}

_finish       {PUTCELL "Please call our office for details."}
              {DOWN 2}
              {PUTCELL "Sincerely,"}
              {DOWN 3}
              {PUTCELL "James Madison (Account Officer)"}
              {DOWN}

text_area     Because of your current standing with P.K.
              Finance, we have decided to
```

```
extend your credit limit.

Please call our office for details.

Sincerely,

James Madison (Account Officer)
```

See Also

Using Macros

Macro Command Descriptions

{DLL <DLLName.>FunctionName, Argument1, Argument2,...}

{DLL} runs a macro or returns a value from an add-in @function contained in a dynamic-link library file. The @function can have up to 16 arguments.

Example

This statement calls the @function AMPLITUDE, included in the DLL Math, with two blocks as arguments:

```
{DLL Math.AMPLITUDE,A1..A10,B1..B10}
```

See Also

Using Macros

Macro Command Descriptions

{DLL.Load DLLName}

{DLL.Load} loads a dynamic-link library (DLL) program. You can use {DLL.Load} to load a DLL containing add-in @functions or macros. When the DLL is loaded, you can reference add-in @functions contained in the DLL without typing the DLL name. Similarly, macros contained in the DLL become resident in memory.

You can use {DLL.Load} to define a startup macro in QPW.INI

Example

`{DLL.Load MYDLL}` loads a DLL program named MYDLL

See Also

Using Startup and Exit Macros

Using Macros

Macro Command Descriptions

{DODIALOG DialogName, OKExit?, <Arguments>, <MacUse?>}

{DODIALOG} displays a dialog box for the user or the macro to manipulate. *MacUse?* specifies how the dialog box is manipulated; if 0, the macro manipulates the dialog box, if 1, the macro pauses so the user can manipulate the dialog box.

DialogName is the name of the dialog box to display. *Arguments* is a block containing initial settings for controls in the dialog box. Each cell in *Arguments* corresponds to one control in the dialog box with a Process Value property set to Yes; the order of controls in the dialog box determines which cell maps to which control (the first cell maps to the first control, the second cell maps to the second, and so on). See *Building Spreadsheet Applications* for a discussion of ordering controls and the Process Value property.

If a dialog box is under macro control (*MacUse?* set to 0), you can make it revert to user control using {PAUSEMACRO}.

Once the dialog box closes (performing its function), Quattro Pro writes the new control settings into their respective cells. *OKExit?* is a cell containing the result of the dialog box operation; 1 if OK was chosen, 0 if the dialog box was canceled.

See Also
{GETLABEL}
{GETNUMBER}

{DOWN <Number>} and {D <Number>}

{DOWN} and {D} are equivalent to the Down key. The optional argument *Number* moves the selector down the corresponding number of rows. You can also use cell references or block names as arguments.

Examples

`{DOWN}{DOWN}`	moves the selector down two rows.
`{DOWN 4}`	moves the selector down four rows.
`{DOWN G13}`	moves down the number of rows specified in cell G13.
`{DOWN count}`	moves down the number of rows specified in the first cell of the block named count.

See Also

Using Macros

Macro Command Descriptions

{EDIT}

{EDIT} is equivalent to the Edit key, F2. Its main use is in Edit mode, where it lets you edit the contents of the active cell.

See Also

Using Macros

Macro Command Descriptions

{END}

{END} is equivalent to the End key.

See Also
Using Macros
Macro Command Descriptions

{ESC} and {ESCAPE}

{ESC} and {ESCAPE} are equivalent to the Esc key, which is useful for clearing menus or dialog boxes from the screen, one at a time. To clear all prompts and return Quattro Pro to Ready Mode, use {BREAK}.

See Also

Using Macros

Macro Command Descriptions

{EXEC AppName, WindowMode, <ResultLoc>}

{EXEC} lets you run other Windows applications or DOS commands. *AppName* can be any valid command string you could type in the Program Manager's File|Run command. *AppName* can contain the path of the application. If no path is given (for example, `"\OVISION\VISION.EXE"`), then the application must be in the system path so Windows can find it.

ResultLoc is useful in some cases where the started application supports DDE along with an instance number. For example, Excel and Quattro Pro respond to their names and also respond to their names concatenated with an instance number.

Examples

`{EXEC "NOTEPAD.EXE",1}` runs the Windows Notepad and displays it just as if it were run from the Program Manager.

`{EXEC "VISION.EXE",2}` runs ObjectVision, and places it on the Windows Desktop as a minimized application.

`{EXEC "COMMAND /C DIR>TEST.TXT",1}` stores the current directory listing in the file TEST.TXT.

See Also

{INITIATE}

{EXECUTE DDEChannel, MacroString, <ResultLoc>}

With {EXECUTE} you can make other applications that support DDE run their macros or scripts. The macro to run is stored in *MacroString*. It must be in the syntax the application normally uses. For example, {PGDN} works as a macro command for Quattro Pro, but the macro command for it in Excel is `[VPAGE(1)]`. You must open a channel of conversation with the other application using {INITIATE} (which determines the value of *DDEChannel*) before using {EXECUTE}.

The contents of *ResultLoc* depend on the application you are contacting. Most often, if a macro succeeds, *ResultLoc* contains 1. If a macro fails, {EXECUTE} results in an error.

See {REQUEST} and {POKE} for more information on receiving data from and sending data to other applications using DDE.

Example

This example initiates a DDE conversation with ObjectVision and opens the file NOTES.OVD (using the ObjectVision command @APPOPEN). The example assumes that ObjectVision file TASKLIST.OVD is open when the Quattro Pro macro runs.

```
channel         15
command         [@APPOPEN("NOTES.OVD")]
result          0

_new_visio      {INITIATE "VISION","TASKLIST.OVD",channel}
  n
                {EXECUTE channel,+command,result}
```

See Also

About DDE Macro Commands

Using Macros

Macro Command Descriptions

{EXPON InBlock, OutBlock, <Damping>, <StdErrs(0|1)>}

{EXPON} performs exponential smoothing on a series of values. {EXPON} is equivalent to the Exponential Smoothing analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{FILESIZE Location}

{FILESIZE} calculates the number of bytes in a file previously opened using {OPEN} and places the result in *Location* as a value. If *Location* is a block, the result is placed in the upper left cell of the block. If the command is successful, macro execution continues in the cell below the cell containing the {FILESIZE} command, ignoring any other commands in that cell. If {FILESIZE} fails (for example, when no file is open), macro execution continues in the cell containing the {FILESIZE} command.

Example

The following example opens the file MYFILE.TXT and places its size (in bytes) in the cell named num_bytes.

```
\F          {OPEN "MYFILE.TXT",R}
            {FILESIZE num_bytes}
            {CLOSE}

num_bytes   45
```

See Also

Using Macros

Macro Command Descriptions

{FLOATCREATE ObjectType, UpperCell, xoffset, yoffset, LowerCell, xoffset2, yoffset2, Text}

{FLOATCREATE} lets you create a SpeedButton or floating graph in the active notebook window. Use {CREATEOBJECT} to create objects in dialog windows or graph windows.

All positions in {FLOATCREATE} are positive offsets from cells in the notebook containing the upper left and lower right corners of the object.

If you need to modify the floating graph or button after creating it, change the property settings immediately after creation. It is selected then, so you won't need to click it or use {SELECTFLOAT}. You might also want to change the name at this time and document it for later use with {SELECTFLOAT}.

Example

The following macro creates a SpeedButton that covers the block A1..B2, then stores the name of the button in A26. The button reads `Save File`:

```
{FLOATCREATE Button,A1,0,0,C3,0,0, "Save File"}
{GETPROPERTY A26, "Object_Name"}
```

The following macro creates a button 50 twips to the right and 50 twips below the upper left corner of the button in the previous example. It reads `Open File`:

```
{FLOATCREATE Button,A1,50,50,C3,50,50, "Open File"}
```

The following macro creates a floating graph that's offset 35 twips from the block C2..E10, but the same size:

```
{GraphNew Graph3}
FLOATCREATE Graph,C2,35,35,E10,35,35,"Graph3"}
```

See Also

{FLOATMOVE}

{FLOATSIZE}

{FLOATMOVE UpperCell, xoffset, yoffset}

{FLOATMOVE} lets you move a floating object in the active notebook window. The item to move is selected using {SELECTFLOAT}. The new position in {FLOATMOVE} is specified as a positive offset from a cell in the notebook.

See Also
{MOVETO}
{FLOATSIZE}

{FLOATSIZE UpperCell, xoffset, yoffset, LowerCell, xoffset2, yoffset2}

{FLOATSIZE} lets you resize a floating object in the active notebook window. The item to resize is selected using {SELECTFLOAT}.

All positions in {FLOATSIZE} are positive offsets from a cell in the notebook.

See Also
{RESIZE}
{FLOATMOVE}

{FOR CounterLoc,Start#,Stop#,Step#,StartLoc}

{FOR} repeatedly runs a macro subroutine beginning at *StartLoc*, creating a macro loop. Quattro Pro keeps track of how long the macro runs using *CounterLoc*. At the start of the loop, *CounterLoc* is set to *Start#*. Each time an iteration of the loop runs, *CounterLoc* is increased by *Step#*. When *CounterLoc* reaches or exceeds *Stop#*, execution stops.

Examples

`{FOR D15,1,5,1,E30}`	runs the subroutine beginning in cell E30 five times.
`{FOR D15,1,10,2,E30}`	runs the subroutine five times because the counter increments by two during each iteration.
`{FOR D15,1,5,0,E30}`	runs the subroutine continuously until you press Ctrl+Break, because adding 0 to the start value of 1 can never make the counter exceed 5.
`{FOR D15,1,0,1}`	results in an error because no subroutine is specified.

The following macro creates "Qtr" labels for reports; the year displays above `Qtr1`:

```
\Q        {; position cursor where labels should appear}
          {FOR counter, 1992,1995,1,_list}
          {; return to original position}
          {LEFT}{END}{LEFT}

counter

_list     {PUTCELL +counter}
          {BlockValues C(0)R(0),C(0)R(0)}
          {DOWN}
          {PUTCELL "Qtr1"}{RIGHT}
          {PUTCELL "Qtr2"}{RIGHT}
          {PUTCELL "Qtr3"}{RIGHT}
          {PUTCELL "Qtr4"}{RIGHT 2}{UP}
```

See Also

Using Macros

Macro Command Descriptions

{FORBREAK}

{FORBREAK} cancels the subroutine run by a {FOR} command and ends the processing of {FOR}. Macro execution continues normally with the command following the {FOR} command.

If {FORBREAK} appears anywhere other than in a subroutine called by {FOR}, macro execution ends, and Quattro Pro displays an error message.

{FORBREAK} is usually used in conjunction with {IF} to exit the macro if a specific condition is reached; for example, if an error condition is found.

Example

The following example shows {FORBREAK} terminating a loop used to enter names into a list. Enter `STOP` to stop the loop.

```
\F          {EditGoto list_top}
            {FOR counter,1,10000,1,_get_name}

_get_name   {GETLABEL "Enter next name: ",name_cell}
            {IF name_cell="STOP"}{FORBREAK}
            {LET @CELLPOINTER("Address"),name_cell}{DOWN}

name_cell   STOP
counter     4
list_top
```

See Also

Using Macros

Macro Command Descriptions

{FORM <Block>}

{FORM} is the command equivalent for Data|Form. It creates forms for entering and finding data records without programming. *Block* defaults to the current selected block.

See Also

Data|Form

Using Macros

Macro Command Descriptions

{FOURIER InBlock, OutBlock, <Inverse>}

{FOURIER} performs a fast Fourier transformation on a block of data. {FOURIER} is equivalent to the Fourier analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{FTESTV InBlock1, InBlock2, OutBlock, <Labels(0|1)>}

{FTESTV} performs a two-sample F-test to compare population variances. {FTESTV} is equivalent to the F-Test analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{FUNCTIONS}

{FUNCTIONS} is equivalent to the Functions key, Alt+F3, which displays a list of @functions to enter in the input line.

See Also

Using Macros

Macro Command Descriptions

{GET Location}

{GET} pauses macro execution, accepts one keystroke from the user (no carriage return is necessary), and stores the keystroke as a macro command in *Location*. It then continues macro execution. Any Quattro Pro keystroke can be recorded with {GET} except Shift+F2, Pause, Caps Lock, Num Lock, and Scroll Lock. Unlike {?}, the keystroke isn't passed to Quattro Pro.

{GET} is useful for creating macros that run in the background while the user works. It can detect each key, and restrict the actions the user performs. You can use {LOOK} with {GET} to check the typeahead buffer for user response.

Example

The following example asks users if they want to continue.

```
\F          {RIGHT 10}
            {; display msg_area in top left of screen}
            {GOTO}msg_area~

_again      Press Y to continue...~
            {GET keystroke}
            {IF keystroke<>"Y"}{BEEP2}{HOME}{QUIT}
            {BEEP 4}{BRANCH _again}

keystroke
msg_area    Press Y to continue...
```

See Also

{MENUBRANCH}
{MENUCALL}
{LOOK}
{GETNUMBER}
{GETLABEL}

{GETDIRECTORYCONTENTS Block, <Path>}

{GETDIRECTORYCONTENTS} enters an alphabetized list of file names (determined by the path and DOS wildcard specified by *Path*) into *Block*; if *Path* isn't included, {GETDIRECTORYCONTENTS} lists all the files in the current directory. *Path* must contain a DOS wildcard like *.BAT or *.*.

Caution: If *Block* is one cell, {GETDIRECTORYCONTENTS} overwrites any information beneath the cell (if it finds more than one file). To restrict the file names to a specific block, set *Block* to a block larger than one cell.

Examples

`{GETDIRECTORYCONTENTS A2,"C:\*.*"}` fills column A (starting at row 2) with a list of the files in the root directory of drive C.

`{GETDIRECTORYCONTENTS A2..C7,"C:\QPW\*.*"}` fills the block A2..C7 with a list of the files in the QPW directory on drive C. The first filename is stored in A2, the second in B2, and so on. If more than 18 files are found, the block is only filled with the first 18.

`{GETDIRECTORYCONTENTS C7,"C:\QPW\SAMPLES\*.W??"}` fills column C (starting at row 7) with a list of the files in the QPW\SAMPLES directory on drive C that have file extensions beginning with W.

See Also

Using Macros

Macro Command Descriptions

{GETLABEL Prompt,Location}

{GETLABEL} pauses macro execution, displays *Prompt* in a dialog box, and accepts keystrokes until the user chooses OK or presses Enter. At that time, Quattro Pro stores the characters typed as a label in *Location*. Unlike {GET}, it doesn't accept Quattro Pro's special keys (for example, Home).

Prompt can be a literal string, such as "Enter date:", or a text formula such as +B6 (which contains a label). The prompt string itself can be up to 70 characters long, and is truncated if longer. The response can be as long as 160.

If {PANELOFF} is in effect, {GETLABEL} ignores it, letting the menus, status line, and input line be viewed and updated as the input is typed. Once input is completed (indicated by Enter or choosing OK), then these portions of the display are turned off again.

Examples

This example displays the label in B6 as the dialog box prompt, and then stores the result in cell D1:

```
{GETLABEL +B6,D1}
```

The following example stores the user's last name in the cell named last_cell:

```
\F         {GETLABEL "Enter your last name:",last_cell}
last_cell
```

See Also

{GETNUMBER}

{GETOBJECTPROPERTY Cell,Object.Property}

{GETOBJECTPROPERTY} lets you view objects in Quattro Pro without using the mouse, including objects normally not selectable (like the application title bar). Selectable objects, such as blocks and annotations, can also be studied with {GETPROPERTY}. See {SETOBJECTPROPERTY} for the syntax of *Object.Property*.

Examples

`{GETOBJECTPROPERTY A23,"Active_Notebook.Zoom_Factor"}` stores the Zoom Factor property's current setting in cell A23.

`{GETOBJECTPROPERTY B42,"/File/Exit.Enabled"}` stores whether File|Exit is operational or not in the cell B42.

See Also

Property Reference
{GETPROPERTY}
{SETPROPERTY}

{GETNUMBER Prompt,Location}

{GETNUMBER} is like {GETLABEL}, but accepts only a numeric value, a formula resulting in a numeric value, or the cell address or block name of a cell returning a value. If a text value is input, ERR is entered in the cell. The prompt can be up to 70 characters long, and the response can be as long as 160 characters.

Example

The following example stores the user's age in the cell named age_cell. If a number isn't entered, which the macro detects by checking to see whether age_cell contains ERR, it prompts again.

```
\F          {GETNUMBER "Enter your age:",age_cell}
            {IF @ISERR(age_cell)}{BRANCH \F}

age_cell
```

See Also

Using Macros

Macro Command Descriptions

{GETPOS Location}

{GETPOS} places the position of the file pointer in *Location* as a value. If no file has been opened using {OPEN}, the command is ignored.

The file pointer is a number corresponding to the position at which the next character written to the file will be placed. If the file pointer is 0, the next character written to the file with {WRITE} or {WRITELN} is placed at the beginning of the file. If the file already contains information, it's overwritten, beginning at the first position in the file. After the file is written to, the file pointer is positioned immediately after the last character written. If a file is newly created, then written to for the first time, the file pointer will be the size of the file.

If {GETPOS} succeeds, macro execution continues in the cell below the cell containing the {GETPOS} command, ignoring any commands in the same cell as {GETPOS}; if {GETPOS} fails, macro execution continues in the same cell.

Example

The following example opens the text file TEST.TXT, reads in a line, and calculates the length of the line by subtracting the starting position from the ending position and subtracting 1 from the result. This adjustment is necessary because the carriage return and linefeed characters (found at the end of each line in a typical text file) are stripped away by Quattro Pro when the text is read into cells. So, the calculated length is one character longer than the actual length. The text file read here was created by the macro example provided in the {WRITELN} command description.

```
\F          {OPEN "A:TEST.TXT",R}
            {GETPOS start}
            {READLN input}
            {GETPOS end}
            {CLOSE}
            {LET num_char,+(end-start)-1}

input       This is a short line.
start       0
end         22
num_char    21
```

See Also

Using Macros

Macro Command Descriptions

{GETPROPERTY Cell, Property}

{GETPROPERTY} lets you study the property settings of whatever object is selected. *Property* is the property to view (see Property Reference for a list of properties); its setting is stored in *Cell*.

Examples

`{GETPROPERTY A23,"Text_Color"}` stores the Text Color setting of the selected object in the cell A23.

`{GETPROPERTY B42,"Box_Type"}` stores the border style of the selected object in cell B42.

See Also

{GETOBJECTPROPERTY}
{SETPROPERTY}
{SETOBJECTPROPERTY}

{GETWINDOWLIST Block}

{GETWINDOWLIST} stores a list of the windows open on the Quattro Pro desktop in *Block*, including dialog windows and graph windows. Windows hidden by Window|Hide aren't included.

Caution: If *Block* is one cell, {GETWINDOWLIST} overwrites any information beneath the cell (if it finds more than one window open). To restrict the window names to a specific block, set *Block* to a block larger than one cell.

Example

`{GETWINDOWLIST A2..C5}` stores a list of open windows in the block A2..C5. The first window name is stored in A2, the second in B2, and so on. If more than twelve windows are open, only the first twelve are stored in the block.

See Also

{CHOOSE}

Using Macros

Macro Command Descriptions

{GOTO}

{GOTO} lets you move the selector to a specific location.

A related macro, {QGOTO}, selects the cell or block specified as its destination, while {GOTO} moves to the cell in the upper-left corner of the specified block.

See Also

Using Macros

Macro Command Descriptions

{GRAPH}

{GRAPH} is equivalent to the Graph key, F11, which displays the current graph.

See Also
Using Macros
Macro Command Descriptions

{GRAPHCHAR Location}

{GRAPHCHAR} returns the key a user presses to leave a graph or to remove a message box. The character is stored at *Location*. This command lets you create a graph or message that functions as a menu of choices. The character returned can be used to branch or display any other data.

{GRAPHCHAR} works with {MESSAGE} to record what key the user pressed to remove a message box. It captures only characters the user can normally type into a cell; special characters such as Esc aren't stored.

Examples

The following example displays a message box (its contents being the_msg) and returns the character the user pressed to remove it. The key pressed is stored in a block named the_key.

```
\A          {MESSAGE the_msg,5,5,0}
            {GRAPHCHAR the_key}
the_key
the_msg     Press C to continue or any other key to quit...
```

The next example displays a graph and waits for the user to press a key. If the user presses P, B, or A, Quattro Pro displays a (predefined) pie graph, bar graph, or area graph, respectively. The process repeats until the user presses any key but those three.

```
 the_graph      bar
 the_key        b

 _graphic       {GraphView}

 _continue      {GRAPHCHAR the_key}
                {if @UPPER(the_key)="P"}{_draw "pie"}
                {if @UPPER(the_key)="B"}{_draw "bar"}
                {if @UPPER(the_key)="A"}{_draw
                  "area"}

 _draw          {DEFINE the_graph}
                {GraphView the_graph}
                {BRANCH _continue}
```

See Also

Using Macros

Macro Command Descriptions

{HELP}

{HELP} is equivalent to the Help key, F1. It displays a help topic.

See Also
Using Macros
Macro Command Descriptions

{HISTOGRAM InBlock, OutBlock, <BinBlock>, <Pareto(0|1)>, <Cum(0|1)>}

{HISTOGRAM} calculates the probability and cumulative distributions for a sample population, based on a series of bins. {HISTOGRAM} is equivalent to the Histogram analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{HLINE Distance}

{HLINE} scrolls the active notebook horizontally by *Distance* columns. If the number is positive, it scrolls right; if negative, it scrolls left. {HLINE} doesn't move the selector; only the view of the notebook is altered, just as if the scroll bars were used. Use the commands {RIGHT} or {LEFT} to move the selector horizontally.

Examples

`{HLINE 10}` scrolls the display 10 columns to the right.

`{HLINE -5}` scrolls the display 5 columns to the left.

See Also

{HPAGE}

{VLINE}

{VPAGE}

{HOME}

{HOME} is equivalent to the Home key.

See Also
Using Macros
Macro Command Descriptions

{HPAGE Distance}

{HPAGE} scrolls the active notebook horizontally by *Distance* screens. If the number is positive, it scrolls right; if negative, it scrolls left. {HPAGE} doesn't move the selector; only the view of the notebook is altered. Use {BIGRIGHT} or {BIGLEFT} to move the selector horizontally.

See Also

{HLINE}

{VLINE}

{VPAGE}

{IF Condition}

{IF} operates like @IF. {IF} evaluates *Condition*; if it's numeric and nonzero, it's considered to be TRUE and macro execution continues in the same cell; if *Condition* is anything else, it's considered FALSE, and macro execution continues in the cell directly below the {IF} command. Unlike @IF, {IF} commands can't be nested within each other.

Examples

The following example says, in English, "If the value stored in gpa is greater than 0.59, run the macro _pass_student; otherwise, run the macro _fail_student."

```
\F   {IF gpa>.59}{BRANCH _pass_student}
     {BRANCH _fail_student}
```

If you don't want both the true and false clauses executing, be sure to include {BRANCH} or {QUIT} in the same cell as {IF}, as shown in both examples. The following example uses a string of {IF} commands to award a grade based on a test result:

```
\G   {IF result>=.90}{BRANCH _give_a}
     {IF result>=.80}{BRANCH _give_b}
     {IF result>=.70}{BRANCH _give_c}
     {IF result>=.56}{BRANCH _give_d}
     {BRANCH _give_f}
```

See Also

@IF

Using Macros

Macro Command Descriptions

{IFKEY String}

{IFKEY} is like {IF}, but runs the next macro command in the current cell if *String* is the name of a key macro command (such as {ESC} or {HOME}). Don't enclose the name stored in *String* in braces. For example, `{IFKEY "HOME"}` evaluates as true because HOME is the name of a macro command; `{IFKEY "A"}` evaluates as false because it isn't the name of a macro command. *String* can be a string or a text formula.

See Also

{IF}

Using Macros

Macro Command Descriptions

{IMFORMAT Format}

{IMFORMAT} specifies how complex numbers display in the active notebook, and returns a label showing the selected format.

Examples

`{IMFORMAT 1}` returns "x+iy"
`{IMFORMAT 2}` returns "x+jy"

See Also

Using Macros
Macro Command Descriptions

{INDICATE String}

{INDICATE} sets the mode indicator in the lower right corner of the screen to read whatever is given as *String*. If *String* is longer than seven characters, only the first seven are used. To restore the mode indicator to its normal setting, use {INDICATE} with no arguments. To hide the mode indicator, use {INDICATE ""}.

Example

`{INDICATE "Save!"}` changes the indicator to read Save!.

`{INDICATE " Go! "}` changes the indicator to read Go! with a space preceding and following it.

`{INDICATE E14}` changes the indicator to E14 because cell references are ignored.

`{INDICATE}` restores the normal mode indicator.

See Also

Using Macros

Macro Command Descriptions

{INITIATE AppName, Topic, ChannelLoc}

{INITIATE} requests a channel of communication with the application *AppName*. This application must support Dynamic Data Exchange (DDE). If the request succeeds, the identification number of the conversation (DDE communication) is stored in the cell *ChannelLoc. Topic* can be an Excel spreadsheet, an ObjectVision form, or a Word for Windows document, or other DDE server-application file. If there is no specific file to communicate with in the other application, you can set *Topic* to `System`.

AppName, Topic, or both can be an empty string (`""`). The empty string works as a wildcard to display a list of possible conversations to choose from.

You can use {INITIATE} multiple times with the same application; if you do this, use a different *ChannelLoc* for each conversation. Once a channel opens, you can use {REQUEST}, {EXECUTE}, or {POKE} to communicate with the application.

When the DDE conversation is complete, use {TERMINATE} to end it. See {EXECUTE} for an example of {INITIATE}.

See Also

About DDE Macro Commands

Using Macros

Macro Command Descriptions

{INS}, {INSERT}, {INSOFF}, and {INSON}

{INS} and {INSERT} toggle the Ins key on or off. {INSOFF} is equivalent to Ins off, and {INSON} to Ins on.

See Also

Using Macros

Macro Command Descriptions

{INSPECT}

{INSPECT} is equivalent to the Inspect key, F12. It displays an Object Inspector for the current object.

See Also

Using Macros

Macro Command Descriptions

{LEFT <Number>} and {L <Number>}

{LEFT} and {L} are equivalent to the Left Arrow key. The optional argument *Number* moves the selector the corresponding number of columns to the left. You can use cell references or block names as arguments.

Examples

`{L}{L}{L}` moves the selector left three columns.

`{LEFT 6}` moves the selector six columns to the left.

`{LEFT D9}` moves to the left the number of columns specified in cell D9.

`{LEFT count}` moves to the left the number of columns specified in the first cell of the block named *count*.

See Also

Using Macros

Macro Command Descriptions

{LET Location, Value<:Type>}

With {LET}, you can enter a value into *Location* without moving to it. {LET} enters the value or string you specify with *Value in Location.*

You can use the optional *Type* argument to specify whether to store *Value* as an actual number or as a string. If you specify a formula as a string, the formula is written into *Location* as a string, not the resulting value. For example, `{LET A1,B3*23:string}` stores the formula B3*23 as a label in cell A1. If you omit *Type*, Quattro Pro tries to store the value as a numeric value; if unsuccessful, it stores the value as a string.

Location must be a cell address or cell name; you can use functions such as @CELLPOINTER as a *Location* in {LET} commands only if they return a cell address or cell name.

You can use {LET} to invoke add-in @functions or macros contained in DLLs. Specify the add-in as *Value,* using this syntax for functions:

```
@dllname.functionname(functionargument1, functionargument2, ...)
```

For example, this statement calls the @function MEDIAN, included in DLL Stats, with a five-item list as an argument and stores the result in *Location* G6:

```
{LET G6,@Stats.MEDIAN(2,4,6,8,10)}
```

The macro syntax is identical:

```
@dllname.macroname(macroargument1, macroargument2, ...)
```

Examples

`{LET(@CELLPOINTER("address")),99}` makes the value of the active cell 99.

The examples below assume A1 contains the label `'Dear`, A2 contains the label `'Sir`, and A3 contains the value 25. The result is shown to the right of each {LET}.

```
\     {LET F1,25}                  25

      {LET F2,A3}                  25
      {LET F3,+A1&""&A2}           Dear Sir
      {LET F4,+A1&""&A2:value}     Dear Sir
      {LET                         +A1&""&A2
        F5,+A1&""&A2:string}
      {LET F6,+A1&A3}              ERR (because A3 is a
                                     value)
```

See Also

Using Macros

Macro Command Descriptions

{LOOK Location}

When Quattro Pro runs a macro, it doesn't respond to keystrokes the user enters (except Ctrl+Break). If the user presses keys during macro execution, those keystrokes are stored in the computer's typeahead buffer and are responded to when the macro pauses for input or ends.

{LOOK} checks this typeahead buffer for stored keystrokes. If any are found, it places the first keystroke the user typed in Location as a macro command.

{LOOK} can be used while processing long macros to check for a keystroke that might signal the user wants to quit (see the next example).

{LOOK} doesn't remove the keystroke from the buffer. If a macro does nothing other than {LOOK}, the key still passes to Quattro Pro when the macro ends. To remove pending keystrokes from the buffer, use {GET}.

Example

In the following example, the macro gives you 15 seconds to choose the next menu choice. If you don't, you must reenter the password. Type the line to the right of _check_it with no hard returns until after {BRANCH _password}, then press Enter to insert it into one cell.

```
\M                {QGOTO}msg_area~
                  A. Add name{DOWN}
                  B. Edit name{DOWN}
                  C. Delete name{DOWN}
                  Enter choice: {RIGHT}
                  { }
                  {LET start_time,@NOW}

_check_it         {LOOK keystroke} {IF keystroke=""} {IF @NOW>start_time
                    + @TIME(0,0,15)} {BRANCH _password}
                  {IF keystroke<>""}{BRANCH _take_action}
                  {BEEP 4}
                  {BRANCH _check_it}

_password         {; get password from user}
                  {GETLABEL "Enter password : ",pass}

_take_action      { }
                  {BEEP 3}
start_time
keystroke
pass
msg_area
```

See Also

Using Macros

Macro Command Descriptions

{MACROS}

{MACROS} is equivalent to the Macros key, Shift+F3, which displays a menu of macro commands to type into the input line.

See Also

Using Macros

Macro Command Descriptions

{MARK}

{MARK} is equivalent to the Select key, Shift+F7, which lets you begin selection of a block.

See Also

Using Macros

Macro Command Descriptions

{MCORREL InBlock, OutBlock, <Grouped>, <Labels(0|1)>}

{MCORREL} computes the correlation matrix between two or more data sets. {MCORREL} is equivalent to the Correlation analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{MCOVAR InBlock, OutBlock, <Grouped>, <Labels(0|1)>}

{MCOVAR} returns the covariance matrix between two or more data sets. {MCOVAR} is equivalent to the Covariance analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{MENU}

{MENU} is equivalent to the Menu key (Alt or F10). This command moves the selector to the menu bar and highlights the first menu.

See Also

Using Macros

Macro Command Descriptions

{MENUBRANCH Location}

{MENUBRANCH} pauses macro execution to display the custom pop-up menu stored at *Location*. After the user chooses an item from the menu, macro execution continues with the cell below the description of the menu choice. Often this is a {BRANCH}. The only exception is if the user presses Esc, or clicks outside the menu; in this case, the macro continues in the {MENUBRANCH} cell.

With the exception of the Esc key, a custom pop-up menu acts the same as one of Quattro Pro's own menus. Users can press the arrow keys to look at each item, and can choose a menu item by either pressing Enter while highlighting it, pressing the first letter of the menu item's name, or choosing it with the mouse.

See Menu Design Tips for information on how to design a custom menu. See {ADDMENU} and {ADDMENUITEM} for information on adding menus or menu commands to the Quattro Pro menu bar.

Example

The following macro displays a custom menu that offers the user three choices. If the user presses Esc, the menu is redisplayed with `{BRANCH \F}`.

```
quit_menu   Continue                    Return                  Quit
            Continue macro              Return to Ready         Leave Quattro Pro
            {BRANCH \F}                 {BRANCH _continue}      {MENUBRANCH sure}

sure        No                          Yes
            Stay in Quattro Pro         Return to DOS
            {BRANCH \F}                 {FileExit}

\F          {MENUBRANCH quit_menu}
            {BRANCH \F}
_continue
```

See Also

{MENUCALL}

{MENUCALL Location}

{MENUCALL} pauses macro execution and displays the custom pop-up menu stored at *Location*. It treats the called menu as a subroutine. After the user chooses an item from the menu, macro execution continues in the cell containing {MENUCALL}.

See {ADDMENU} and {ADDMENUITEM} for information on adding menus or menu commands to the Quattro Pro menu bar.

Example

The following macro uses two consecutive custom menus to let the user change search criteria for a database. It then copies records that meet the criteria to the output block and branches to another macro to print the output block as labels. The block name stat references the cell in the criteria table containing A in the example; pay references the cell in the criteria table containing F.

```
m_status      Active                                Retired
              Active Members                        Retired Members
              {LET stat, "A"}                       {LET stat, "R"}

m_paid        Paid                                  Unpaid
              Dues are paid                         Dues are unpaid
              {LET pay, "T"}                        {LET pay, "F"}

\M            {;Choose members for label
               print}
              {MENUCALL m_status}
              {MENUCALL m_paid}
              {Query.Extract}
              {BRANCH print_labels}
```

Database Criteria Block:

```
              STATUS              PAI
                                    D
              A                   F
```

See Also

{MENUBRANCH}

{MESSAGE Block,Left,Top,Time}

{MESSAGE} displays the contents of *Block* in a dialog box until *Time* is reached; *Time* is a standard Quattro Pro date/time serial number. The window appears at the screen (not the notebook) coordinates *Left,Top*. The width and depth of the box depend on the defined width and depth of the text block.

The message box contains a "snapshot" of current *Block* contents, including all fonts, colors, and other formatting. If *Block* includes a floating graph, bitmap, or SpeedButton, its image displays in the message block.

The height and width of the message box depend on the defined width and depth of *Block*. If text overflows the cell it is typed into, be sure to include adjoining cells in *Block.* Otherwise, the message will be truncated at the edge of *Block.*

The message box always appears over the frontmost window, even if that window isn't a notebook (for example, a graph window).

If you enter 0 for *Time*, Quattro Pro displays the box until the user presses any key. (You can use {GRAPHCHAR} to test for which key is pressed.)

Example

This example displays a message box for 15 seconds and then displays another box indefinitely. The user removes the second box by pressing Y (which is returned in the block the_key).

```
the_key

\A          {MESSAGE msg1,15,5,+@NOW+@TIME(0,0,15)}

_warning    {MESSAGE msg2,15,5,0}
            {GRAPHCHAR the_key}
            {IF @UPPER(the_key)="Y"}{BRANCH _lose_data}
            {BRANCH _warning}

msg1        Warning! You may lose data if you continue.

msg2        Are you sure you want to continue?
            Press Y to continue or any other key to cancel...
```

See Also

{DODIALOG}

{MOVEAVG InBlock, OutBlock, <Interval>, <StdErrs(0|1)>}

{MOVEAVG} returns a moving average for a specified Interval based on the values for the preceding periods in *InBlock*. {MOVEAVG} is equivalent to the Moving Average analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{MOVETO x, y}

{MOVETO} moves all selected objects in the active window (dialog, graph, or Graphs page window) to the position specified by *x,y*. Since {MOVETO} is context-sensitive, you can use it to move controls in a dialog window or drawings in a graph window. It also moves graph icons on the Graphs page. (Use {FLOATMOVE} to move floating objects in a notebook window.)

The coordinates *x* and *y* represent where to move the upper left corner of the object(s). Object size doesn't change.

See Also
{RESIZE}
{SELECTOBJECT}

{MTGAMT <OutBlock>, <Rate>, <Term>, <OrigBal>, <EndBal>, <LastYear>}

{MTGAMT} generates an amortization schedule for a mortgage. {MTGAMT} is equivalent to the Amortization Schedule analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{MTGREFI <OutBlock>, <CurrBal>, <CurrRate>, <RemTerm>, <CandPctFees>, <CandRate>}

{MTGREFI} generates a table of information relating to refinancing a mortgage. {MTGREFI} is equivalent to the Mortgage Refinancing analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{NAME}

{NAME} is equivalent to the Choices key, F3, which displays a list of block names in the current notebook.

See Also
Using Macros
Macro Command Descriptions

{NEXTPANE <CellAtPointer?>}

{NEXTPANE} switches between the panes of a notebook window previously split using Window| Panes. The optional argument *CellAtPointer?* specifies whether the active cell in the pane will be at the location of the selector (1) or its previous position (0). This command is equivalent to the Pane key, F6.

See Also
{ACTIVATE}

{NEXTTOPWIN <Number>}

{NEXTTOPWIN} is equivalent to the Next Window key, Ctrl+F6. It makes the next window active and moves the selector to it.

See Also

Using Macros

Macro Command Descriptions

{NEXTWIN <Number>}

{NEXTWIN} is equivalent to Shift+F6. It makes the bottom window active and moves the selector to it. This macro is included for compatibility with Quattro Pro for DOS.

See Also

Using Macros

Macro Command Descriptions

{NUMOFF} and {NUMON}

{NUMOFF} and {NUMON} are equivalent to Num Lock off and Num Lock on, respectively.

See Also
Using Macros
Macro Command Descriptions

{ONERROR BranchLocation, <MessageLocation>, <ErrorLocation>}

Normally, if an error occurs during macro execution, the macro ends and you see an error message. {ONERROR} tells Quattro Pro to branch to a different location (*BranchLocation*) if an error is encountered and store the error message in *MessageLocation* for future reference. Quattro Pro stores the location of the error in *ErrorLocation*. *MessageLocation* and *ErrorLocation* are optional. If *MessageLocation* is omitted, no error message will be displayed or stored.

Only one {ONERROR} command can be in effect at a time, so each time it's called, the most recent {ONERROR} replaces the previous one. If an error occurs, the {ONERROR} state is "used up" and must be redeclared. It's best to declare {ONERROR} at the very beginning of your macro, or at least before any procedure is likely to result in an error.

In general, {ONERROR} won't capture macro programming errors, such as an incorrect sequence of commands, or an attempt to call a nonexisting subroutine. Nor will it detect syntax errors within a macro itself, such as an error resulting from the incorrect use of a macro keyword.

Typical errors that {ONERROR} detects include

- disk errors, such as a disk drive door left open, a full disk, or failure to find a file of a given name
- attempts to copy to a protected cell when a notebook's protection property is enabled
- attempts to insert a row that would push a nonblank cell off the bottom of the notebook

Example

The following macro uses {ONERROR} by first deliberately causing an error (trying to insert a row that would push text off the bottom of the notebook), and then beeps when that error occurs.

```
\G          {ONERROR _on_err,err_msg,err_loc}
            {EditGoto A8192}
            {PUTCELL "This is the end"}
            {UP}
            {BlockInsert.Row C(0)R(0), "Entire"}

_on_err     {BEEP 5}
err_msg
err_loc
```

See Also

Using Macros

Macro Command Descriptions

{OPEN Filename,AccessMode}

{OPEN} establishes a connection to *Filename* so you can use other file-access macro commands ({READ}, {WRITE}, and so on) on it. There are four different access modes:

R (Read-Only)	Allows only reading from this file, ensuring no changes are made to it. {WRITE} and {WRITELN} can't be used with a file opened as read-only.
M (Modify)	Opens an existing file for modification. All reading and writing commands can be used with a file opened for modification.
W (Write)	Opens a new file with the name given in {OPEN}. If a file already exists with that name, the existing file is erased. All reading and writing commands can be used with a file opened for writing.
A (Append)	Opens an existing file for modification. Allows writing to the selected file only, and positions the file pointer at the end of the file.

Filename's full name and access path must be given, and the entire name must be in quotes. For instance, to open and modify the file DATA.TXT in the subdirectory called FILES on drive C, use the command `{OPEN "C:\FILES\DATA.TXT",M}`. If any part of the access path or file name is left out, the file might not be found.

Caution: Although {OPEN} can provide access to any type of file (including .WB1 spreadsheet files), Quattro Pro's file-access macro commands are designed to work only with plain text files. Using these commands with any other file type isn't recommended and can result in corruption of that file. Make a backup copy of your file before using {OPEN} on it.

Once {OPEN} runs successfully, Quattro Pro skips to the next row of the macro and continues executing instructions there. {OPEN *Filename,*R}, {OPEN *Filename,*M}, and {OPEN *Filename,*A} fail if the file isn't found. {OPEN *Filename*,W} fails if the supplied access path or file name is invalid. If {OPEN} fails, Quattro Pro continues executing commands in the same cell as the {OPEN} command. (See the third macro below for an example.) You can use {ONERROR} to trap some errors that occur in {OPEN}.

Examples

The following example opens the file named MYDATA.TXT for reading. If the file doesn't exist in the default data directory, the command fails.

```
{OPEN "MYDATA.TXT",R}
```

The next example creates a file MYDATA.TXT on drive A for writing. If there's already a file on drive A with that name, it's erased. This command will fail only if there's no disk in drive A.

```
{OPEN "A:MYDATA.TXT",W}
```

The last example demonstrates how to do error trapping in an {OPEN} macro. When you press Ctrl+H, Quattro Pro attempts to open the file C:\MYDIR\DATA.TXT. If that succeeds, the macro stops, since there are no more commands in the row below. If {OPEN} fails (meaning the file doesn't exist), the adjacent {BRANCH} runs. The subroutine _try_again then attempts to create the file. If that succeeds, the macro restarts, since {OPEN} should succeed now that the file has been created. If _try_again fails, probably an invalid directory path was given. Therefore, the adjacent {BRANCH} goes to _bad_dir, which displays a relevant error message with pertinent instructions.

```
\H              {OPEN "C:\MYDIR\DATA.TXT",M}{BRANCH _try_again}
                {CLOSE}

_try_agai       {OPEN "C:\MYDIR\DATA.TXT",W}{BRANCH _bad_dir}
  n
                {CLOSE}
                {BRANCH \H}
```

```
_bad_dir      {EditGoto err_loc}{BEEP 4}
              {PUTCELL "The directory C:\MYDIR\ doesn't exist."}
              {DOWN}
              {PUTCELL "Create it, then try again."}
              {DOWN}
err_loc
```

See Also

Using Macros

Macro Command Descriptions

{PANELOFF}

{PANELOFF} disables normal display of menus and prompts during macro execution when Quattro Pro's Macro Suppress-Redraw property is set to None. It can significantly speed up execution for macros that use keystrokes to walk through menus, since it saves Quattro Pro the time normally needed to draw its menus on the screen. Its effect is canceled by Quattro Pro once the macro stops executing, so you needn't worry about locking macro users out of the menus. To cancel its effect during macro execution, use {PANELON}.

{PANELOFF} doesn't disable menus created by {MENUCALL} and {MENUBRANCH} or subroutine calls that use menus or dialog boxes. Use this command with {WINDOWSOFF} to completely disable normal screen updating.

See Also
{WINDOWSON}

{PANELON}

{PANELON} enables display of menus and prompts that have been disabled with {PANELOFF}. {PANELON} has no effect if used without an accompanying {PANELOFF}. Therefore, it can be used repeatedly with no ill effects.

Use this command with {WINDOWSON} to completely restore normal screen updating.

See Also
Using Macros

Macro Command Descriptions

{PAUSEMACRO}

{PAUSEMACRO} is used with {DODIALOG} or a command equivalent invoked with ! to pause the macro so that the user can "finish up" whatever dialog box is displaying. Once the user finishes using the dialog box (by choosing OK, or canceling it), macro exection resumes with any macro commands following the {PAUSEMACRO}.

Example

The following macro displays the Block|Copy dialog box, sets the From edit field to A1, activates the To edit field, and then waits for the user to complete the copy operation. Once the user finishes the dialog box, the macro beeps and moves down a cell.

```
_copy_a1        {BlockCopy!}
                {ALT+F}
                A1
                {ALT+T}
                {PAUSEMACRO}
                {BEEP}{DOWN}
```

See Also

{DODIALOG}

{PGDN <Number>} and {PGUP <Number>}

{PGDN} and {PGUP} are equivalent to PgDn and PgUp, respectively. Use *Number* to specify how many times the operation is repeated; for example, `{PGUP 7}` is equivalent to pressing PgUp seven times.

See Also

Using Macros

Macro Command Descriptions

{POKE DDEChannel, Destination, DataToSend}

{POKE} sends information to an application that supports Dynamic Data Exchange (DDE). This application is identified by *DDEChannel*. The type of application determines what *Destination* is; it could be a block of cells in Excel or a bookmark in Word for Windows. *DataToSend* is a block containing the information to send. You must use the command {INITIATE} to open a channel of conversation before you can use {POKE} (this also determines the value of *DDEChannel*).

Example

This example starts a conversation with TASKLIST.OVD, which is a file open in ObjectVision. It sets the ObjectVision field Task to the label stored in new_task, and unchecks the Completed check box. Then the new task is inserted into the task list. The command block contains an ObjectVision command not available in Quattro Pro:

```
dde_channel     10
command         [@INSERT("tasks")]
exec_result     0
new_task        Call Jim re: task priorities
task_status     No
_new_vision     {INITIATE "VISION","TASKLIST.OVD",dde_channel}
                {POKE dde_channel,"Task",new_task}
                {POKE dde_channel,"Completed",task_status}
                {EXECUTE dde_channel,+command,exec_result}
```

See Also

About DDE Macro Commands

{REQUEST}

{PTTESTM InBlock1, InBlock2, OutBlock, <Labels(0|1)>, <Alpha>, <Difference>}

{PTTESTM} performs a paired two-sample Student's t-Test for means. Each value from *InBlock1* is paired with a value from *InBlock2*. *InBlock1* and *InBlock2* must have the same number of values.

{PTTESTM} is equivalent to the t-Test analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{PTTESTV InBlock1, InBlock2, OutBlock, <Labels(0|1)>, <Alpha>}

{PTTESTV} performs a Student's t-Test using two indepependent (rather than paired) samples with unequal variances. {PTTESTV} is equivalent to the t-Test analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{PUT Location, Column#, Row#, Value<:Type>}

{PUT}, like {LET}, copies a value to a particular cell. However, instead of placing the value directly in the specified cell, {PUT} copies *Value* into the cell that is offset *Column#* columns and *Row#* rows into *Location*.

{PUT} processes *Value* the same way {LET} does, including the use of `:string` (or `:s`) and `:value` (or `:v`). If neither of these two optional arguments is supplied, {PUT} tries to store the value as a numeric value; if unsuccessful, it stores the value as a label.

The values for *Column#* and *Row#* can be any number between 0 and one less than the number of columns or rows within *Location*, respectively. A value of 0 implies the first column or row, 1 implies the second, and so on. If *Column#* or *Row#* exceeds the number of columns or rows in the block, the macro stops. ({ONERROR} cannot trap this error.)

Examples

Each of the following examples assumes cell A41 contains the value 25, the block named numbers has been defined as A44..B50, and data is a cell containing the value 295.

`{PUT numbers,1,4,A41:value}` copies the value 25 into the cell at the intersection of the second column and the fifth row of the block numbers (cell B48).

`{PUT numbers,1,5,A41:s}` copies the string "A41" into the cell at the 2nd column and the 6th row of the block numbers (cell B49).

`{PUT numbers,1,6,data}` copies the contents of the block data to the 2nd column and 7th row of numbers (cell B50). If there is no block named data, this example instead places a label ("data") into cell B50.

See Also

Using Macros

Macro Command Descriptions

{PUTBLOCK Value, <Block>}

{PUTBLOCK} lets you quickly enter the same value or label in multiple cells. *Value* is a string or value to place in *Block*. If *Block* isn't specified, the currently selected block is used. *Block* can be noncontiguous; if so, be sure to enclose it in parentheses. If *Value* is a formula containing relative addresses, those addresses are adjusted automatically.

Examples

`{PUTBLOCK "Quarter 1",A..D:A1}` enters the label Quarter 1 in cells A:A1 through D:A1.

`{PUTBLOCK 1990,A..D:B1}` enters the value 1990 in cells A:B1through D:B1.

`{PUTBLOCK "+A1",C3..C12)` enters the formula +A1 in C3, +A2 in C4, and so on.

See Also

{PUTCELL}
{PUT}
{LET}

{PUTCELL Data}

{PUTCELL} is an easy way to store information in the active cell.

Examples

`{PUTCELL "Peggy Danderhoff"}` stores Peggy Danderhoff as a label in the active cell.

`{PUTCELL 45067}` stores the number 45067 as a value in the active cell.

`{PUTCELL @SUM(A1..A27)}` stores the formula @SUM(A1..A27) in the active cell.

See Also

{LET}
{PUT}
{GETNUMBER}
{GETLABEL}

{QGOTO}

{QGOTO} displays and selects the specified block. This command is equivalent to the GoTo key, F5. You can also use the command equivalent {EditGoto *Block*}. A related command, {GOTO}, moves to the upper-left cell of a destination block, but doesn't select the block.

See Also

Using Macros

Macro Command Descriptions

{QUERY}

{QUERY} is equivalent to the Query key, F7, which repeats the last Data|Query operation performed.

See Also

Using Macros

Macro Command Descriptions

{QUIT}

{QUIT} ends all macro execution, and returns control of Quattro Pro to the user.

Use {QUIT} in conjunction with {IF}, {LOOK}, {MENUBRANCH}, or {MENUCALL} to end a macro under user control.

Example

The following macro displays a menu that has a "Quit" option, which returns the user to Ready mode.

```
quit_menu  Continue      Quit
           Keep going    Quit to Ready mode
           {BRANCH \G}   {QUIT}

\G         {MENUBRANCH quit_menu}
```

See Also

Using Macros

Macro Command Descriptions

{RANDOM}

{RANDOM} generates a block of random values drawn from a selected distribution. It's equivalent to the Random Number analysis tool. {RANDOM} has a different format for the following distribution types:

Uniform	every value has an equal probability of being selected.
Normal	has the qualities of a symmetrical, bell-shaped curve.
Bernoulli	has two possible outcomes, failure or success, represented by 0 and 1
Binomial	represents the distribution of successful outcomes in a given number of independent Bernoulli trials.
Poisson	the distribution of values in any interval depends on the length of the interval and the constant Lambda, the expected number of occurrences in an interval
Patterned	a pattern of repeated values and sequences.
Discrete	every value in a designated block has a specified probability of being selected (the cumulative probabilities equal 1).

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, LowerBound, UpperBound}

When the *Distribution* argument equals 1, {RANDOM} generates random values drawn from a uniform distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, Mean, SDev}

When the *Distribution* argument equals 2, {RANDOM} generates random values drawn from a normal distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, Prob}

When the *Distribution* argument equals 3, {RANDOM} generates random values drawn from a Bernoulli distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, Prob, Trials}

When the *Distribution* argument equals 4, {RANDOM} generates random values drawn from a binomial distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, Lambda}

When the *Distribution* argument equals 5, {RANDOM} generates random values drawn from a Poisson distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, LowerBound, UpperBound, Step, RepeatNumber, RepeatSequence}

When the *Distribution* argument equals 6, {RANDOM} generates random values drawn from a patterned distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANDOM OutBlock, Columns, Rows, Distribution, Seed, InBlock}

When the *Distribution* argument equals 7, {RANDOM} generates random values drawn from a discrete distribution.

See Also

{RANDOM}

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{RANKPERC InBlock, OutBlock, <Grouped>, <Labels(0|1)>}

{RANKPERC} returns the ordinal and percent rank of each value in *InBlock*. {RANKPERC} is equivalent to the Rank and Percentile analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{READ #Bytes,Location}

{READ} reads *#Bytes* bytes of characters from a file previously opened using {OPEN} (starting at the current position of the file pointer), and stores them as a label in *Location*. The file is left unchanged, and the file pointer moves to the position following the last character read. (See {GETPOS} for a discussion of the file pointer.)

{READ} is similar to {READLN}, except for two differences. While {READLN} reads one line of characters (terminated by a carriage-return/linefeed pair), {READ} reads the precise number of characters specified. This lets you read, for example, fields within a record rather than an entire record. The second difference is that while {READLN} strips out the carriage-return/linefeed pair at the end of a line, {READ} manipulates these as if they were no different from other characters. If you use {READ} to read a file created by {WRITELN}, you'll see two graphics characters at the end of each string read. These are the carriage return and linefeed characters. {READ} is best used only in conjunction with {WRITE}, or when you know the text file structure in detail.

If {READ} succeeds, macro execution continues in the cell below the cell containing the {READ} command; if {READ} fails, macro execution continues in the same cell.

Example

This macro opens a text file containing a phone directory database, and reads a name and phone number from the third record. The macro that created this text file is shown in the description of {WRITE}:

```
\L              {OPEN "A:PHONEDIR.PRN",R}
                {SETPOS rec_length*(rec_number-1)}
                {READ name_length,name}
                {READ phone_length,phone}
                {CLOSE}

rec_number      3
rec_length      27
name_length     14
phone_lengt     12
  h
name            Hall, Sue Ann
phone           617-555-5678
```

See Also

Using Macros

Macro Command Descriptions

{READLN Location}

{READLN} is like {READ}, but instead of using a number of bytes to determine the amount of text to read, {READLN} reads forward from the current file pointer location up to and including the carriage-return/linefeed at the end of the line. Unlike {READ}, it doesn't read the carriage return/linefeed into the cell. (See {GETPOS} for a discussion of the file pointer.)

Use {READ} to read lines from a record-structured file, where the lines are of uniform length. {READLN} can read the contents of a file one row at a time, making formatting of the data easier than {READ}.

Like the other file-access macros, if {READLN} fails, the macro continues execution in the current cell. If it's successful, the macro skips to the row below, and execution continues there. {ONERROR} can be used to trap disk and file errors, such as a disk drive door being left open.

Example

The following macro opens the text file TEST.TXT, reads in a line, and calculates the length of the line by subtracting the starting position from the ending position, then subtracting 1 from the result. This adjustment is necessary because the carriage-return and linefeed characters found at the end of each line in a typical text file are stripped away by Quattro Pro when the text is read into cells. The macro that created the file TEST.TXT is shown in the description of {WRITELN}.

```
\M          {OPEN "A:TEST.TXT",R}
            {GETPOS start}
            {READLN input}
            {GETPOS end}
            {CLOSE}
            {LET num_char,+(end-start)-1}

start       0
end         22
num_cha     21
  r
input       This is a short line.
```

See Also

{WRITELN}

Using Macros

Macro Command Descriptions

{RECALC Location,<Condition>,<Iteration#>}

{RECALC} causes Quattro Pro to recalculate a specified portion of the notebook in a row-by-row order. This is different from normal recalculation, where Quattro Pro recalculates the entire notebook in natural order; that is, before a formula calculates, each cell it references is recalculated first.

With the optional *Condition* argument you can tell Quattro Pro to recalculate formulas in a block repeatedly until the specified condition is met. You can also supply *Iteration#* to specify the maximum number of times to recalculate formulas trying to satisfy *Condition*. To use *Iteration#*, *Condition* must also be supplied.

{RECALC} is useful for rapid recalculation of specified parts of a notebook, particularly when the notebook is so large that global recalculations would significantly slow your work.

{RECALC} overrides the recalculation method specified for the notebook, enforcing row-by-row recalculation. If all the formulas reference only cells above, or to the left in the same row, the notebook will be correctly calculated. If there are references to cells to the left and below, you must use {RECALCCOL}. If there are references to cells below or to the right in the same row as your formula, you must use {CALC} to recalculate the entire notebook.

{RECALC} displays the results of recalculation.

Caution: If there are formulas within the block being recalculated that depend on formulas outside of the block, they might not evaluate correctly. Make sure *Location* encompasses all the cells referenced by formulas within the block.

See Also

Using Macros

Macro Command Descriptions

{RECALCCOL Location,<Condition>,<Iteration#>}

{RECALCCOL} recalculates the specified portion of a notebook in column-by-column order. It is similar to {RECALC}, which recalculates row by row. See {RECALC} for information on when {RECALCCOL} is appropriate and when you need to use {CALC} instead.

See Also
Using Macros
Macro Command Descriptions

{REGRESS InBlockY, InBlockX, YIntZero(0|1), Labels(0|1), Confidence, SumOutBlock, Residuals(0|1), StdResiduals(0|1), <ResidualOutBlock>, <ProbOutBlock>}

{REGRESS} performs multiple linear regression analysis. {REGRESS} is equivalent to the Advanced Regression analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{REQUEST DDEChannel, DataToReceive, DestBlock}

{REQUEST} gets information specified by *DataToReceive* from applications that support Dynamic Data Exchange (DDE). This information is stored in *DestBlock*. *DataToReceive* is a string representing the location of the data to receive in the other application. In Quattro Pro, this could be a block such as A2..A7 or a property such as "(Application.Display)". If requesting a property, the property must be enclosed in parentheses. In ObjectVision *DataToReceive* could be a field or record. You must use {INITIATE} to open a channel of communication and obtain the value *DDEChannel* before using {REQUEST}.

If your conversation isn't within a specific topic (in other words, you opened the channel using the command {INITIATE *AppName*,"System",*DDEChannel*}), you can use the following strings in *DataToReceive*, depending on the application:

Arguments for DataToReceive

String	Purpose
"SysItems"	A listing of all strings you can use with *DataToReceive*.
"Topics"	A listing of all projects open. For example, a list of open documents under Word for Windows.
"Status"	The current status of the application. For example, READY in Excel or EDIT in Quattro Pro when a cell is being edited.
"Formats"	A list of all Clipboard formats supported by the application.
"Selection"	A list of all items currently selected in the application. For example, in Excel cells A3..A47 could be selected.

Example

This macro gets information from the fields Task and Completed in ObjectVision file TASKLIST.OVD and stores the data in the active notebook. The block command contains an ObjectVision command not available in Quattro Pro:

```
dde_channel         10
command             [@NEXT("TASKS")]
exec_result         0
vision_task         Print out third quarter report
task_complete       Yes
_get_vision_tas     {INITIATE "VISION","TASKLIST.OVD",dde_channel}
k
                    {REQUEST dde_channel,"Task",vision_task}
                    {REQUEST dde_channel "Completed",task_complete}
                    {EXECUTE dde_channel,+command,exec_result}
```

See Also

About DDE Macro Commands

{POKE}

{RESIZE x, y, NewWidth, NewHeight, <VertFlip?>, <HorizFlip?>}

{RESIZE} resizes all selected objects in the active window (dialog or graph window).

See Also
{FLOATSIZE}
{MOVETO}

{RESTART}

{RESTART} changes the current subroutine to the starting routine (or the main routine) by removing all preceding For loops and subroutine calls.

{RESTART} is typically used for error handling. If an application is already nested to near the maximum number of levels and a severe error occurs that requires the macro to end, {RESTART} ensures that additional subroutine calls can be made. If you use the {RESTART} command often, you may want to use {BRANCH} to run subroutines.

See Also

Using Macros

Macro Command Descriptions

{RETURN}

{RETURN} ends the executing subroutine and returns control to the macro that called it. If the macro executing isn't a subroutine, execution stops.

A {RETURN} command at the end of a subroutine is optional, since a macro automatically returns from a subroutine when it reaches a blank cell or a cell containing a value. {RETURN} is usually used with {IF} to return to the main macro if a certain condition is met.

See {*Subroutine*} for an example of using {RETURN}.

See Also
Using Macros
Macro Command Descriptions

{RIGHT <Number>} and {R <Number>}

{RIGHT} and {R} are equivalent to the Right Arrow key. The optional argument *Number* moves the selector the corresponding number of columns to the right. You can also use cell references or block names as arguments.

Example

`{RIGHT}{RIGHT}` moves right two columns.

`{R 6}` moves right six columns.

`{RIGHT D9}` moves right the number of columns specified in cell D9.

`{RIGHT count}` moves right the number of columns specified in the first cell of the block named count.

See Also

Using Macros

Macro Command Descriptions

{ROWCOLSHOW Block, Show?, Row or Col, FirstPane?}

{ROWCOLSHOW} lets you hide or reveal rows and columns (it is equivalent to the block property Reveal/Hide).Use *FirstPane?* when the active window is split into panes (using Window|Panes) or {WindowPanes}.

Examples

`{ROWCOLSHOW A:A..B,1,0,1}` reveals columns A and B on page A.

`{ROWCOLSHOW A:1..7,0,1,1}` hides rows 1 through 7 on page A.

`{ROWCOLSHOW A:1..7,1,1,0}` reveals rows 1 through 7 on page A. If the window is split, the rows are revealed in the right or bottom pane.

See Also

{COLUMNWIDTH}

{ROWHEIGHT}

Using Macros

Macro Command Descriptions

{ROWHEIGHT Block, FirstPane?, Set/Reset, Size}

{ROWHEIGHT} provides two ways to change the height of a row or block of rows (it is equivalent to the block property Row Height). Use *FirstPane?* when the active window is split into panes (using Window|Panes or {WindowPanes}).

Set/Reset specifies how to change the height. To set a row height, use this syntax: {ROWHEIGHT *Block, FirstPane?*, 0, *Size*}

Size is the new row height, in twips. The maximum height is ten inches (14,400 twips).

To reset a row to the default height (determined by font sizes in the row), use this syntax: {ROWHEIGHT *Block, FirstPane?*, 1}

Examples

`{ROWHEIGHT A:1..A:2,1,0,1440}` sets the height of rows 1 and 2 (on page A) to one inch (1,440 twips).

`{ROWHEIGHT A:1..A:2,0,0,2160}` sets the height of rows 1 and 2 (on page A) to one and a half inches (2,160 twips). If the window is split, the top or left pane is affected.

`{ROWHEIGHT A:5,1,1}` resets the height of row 5 (on page A) to the default height.

See Also

{COLUMNWIDTH}

{ROWCOLSHOW}

Using Macros

Macro Command Descriptions

{SAMPLE InBlock, OutBlock, Type, Rate}

{SAMPLE} returns a periodic or random sample from values in InBlock. {SAMPLE} is equivalent to the Sampling analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{SCROLLOFF} and {SCROLLON}

{SCROLLOFF} and {SCROLLON} are equivalent to Scroll Lock off and Scroll Lock on, respectively.

See Also
Using Macros
Macro Command Descriptions

{SELECTBLOCK Block}

{SELECTBLOCK} lets you select a contiguous or noncontiguous block within the active notebook. Parts of a noncontiguous block must be enclosed in parentheses.

Examples

`{SELECTBLOCK A4..B23}` selects the block A4..B23 in the active notebook window.

`{SELECTBLOCK (A:A1..A:B12,B:B13..B:C34)}` selects the noncontiguous block A:A1..A:B12, B:B13..B:C34.

See Also

{SELECTFLOAT}
{SELECTOBJECT}

{SELECTFLOAT ObjectID1, <ObjectID2, ...>}

With {SELECTFLOAT} you can select floating objects in the active notebook window using their names. (To find the name of an object, view it and study its Object Name property.) Use {SELECTOBJECT} to select objects in a graph or dialog window.

Example

`{SELECTFLOAT "Button1"}` selects the SpeedButton in the the active notebook window with the object name Button1.

See Also

About Object Macro Commands

{FLOATMOVE}

{FLOATSIZE}

{SELECTBLOCK}

{SELECTOBJECT ObjectID1, <ObjectID2, ...>}

With {SELECTOBJECT} you can select objects in the active window using their ID numbers or names. (To find the ID number of an object, view it and study its Object ID property. Its name is stored in its Name property.) Since {SELECTOBJECT} is context-sensitive, you can select controls in a dialog window, drawings in a graph window, or icons in the Graphs page.

Example

`{SELECTOBJECT 2,5,7}` selects the objects in the active window with the IDs 2, 5, and 7.

See Also

About Object Macro Commands

Using Macros

Macro Command Descriptions

{SETGRAPHATTR FillColor, BkgColor, FillStyle, BorderColor, BoxType}

{SETGRAPHATTR} lets you quickly set the properties of all selected objects in the active graph window. If one of the arguments specified in the {SETGRAPHATTR} command isn't appropriate for an object, that argument is ignored.

Each color (*FillColor*, *BkgColor*, and *BorderColor*) is in quotes, and specified in RGB format. For *FillStyle*, use any of the strings for that option in the appropriate Object Inspector.

BoxType specifies the new border style for the object; use any *Border Style* property string included in a graph Object Inspector.

See Also

{SETPROPERTY}

{SETOBJECTPROPERTY}

{SPEEDFORMAT}

Using Macros

Macro Command Descriptions

{SETMENUBAR <SystemDefinition>}

{SETMENUBAR} lets you specify which menu system displays on the menu bar. *SystemDefinition* is a block containing the new menu system definition. See Building Menus for a discussion of menu system definitions.

You can use `{SETMENUBAR}` without an argument to restore the Quattro Pro for Windows menu system.

Examples

`{SETMENUBAR "A3..C324"}` makes the system defined in A3..C324 the active menu system.

See Also

{ADDMENU}

{ADDMENUITEM}

{SETOBJECTPROPERTY Object.Property,Value}

{SETOBJECTPROPERTY} can change the property settings of many Quattro Pro objects. Selectable objects such as blocks and annotations can also be changed using {SETPROPERTY}. {SETOBJECTPROPERTY} can affect:

Dialog controls. Use this syntax to specify a control to manipulate in a dialog window: [*Notebook*]*DialogName:ObjectID.Property.* [*Notebook*] is optional. For example, the following macro sets the Fill Color property of the control Rectangle1 in the dialog ColorPick to red:

```
{SETOBJECTPROPERTY "ColorPick:Rectangle1.Fill_Color", "255,0,0"}
```

Graph objects. Use the same syntax as for dialog controls, but substitute the name of the graph in place of *DialogName*. For example, the following macro changes the size of a rectangle named ColorPick in the graph 1QTR92:

```
{SETOBJECTPROPERTY "1QTR92:ColorPick.Dimension", "0,0,25,25"}
```

Menu items. Use the syntax *MenuPath.Property*. See the description of {ADDMENU} for the syntax of *MenuPath*; Changing Command Properties lists the properties available for a menu item.

For example, the following macro disables the Data menu in the active menu system:

```
{SETOBJECTPROPERTY "/Data.Disabled", "Yes"}
```

Value is the new setting for the property. You can also substitute another instance of *Object.Property* for this argument to copy property settings between objects. For example, this macro copies the text color of the active block to the text color of A23:

```
{SETOBJECTPROPERTY "A23.Text_Color","Active_Block.Text_Color"}
```

See Property Reference for a list of properties you can use.

See Also

About Object Macro Commands

{GETPROPERTY}

{GETOBJECTPROPERTY}

{SETPOS FilePosition}

{SETPOS} moves the file pointer of a file previously opened using {OPEN} to the value *FilePosition*. (See {GETPOS} for a discussion of the file pointer.) *FilePosition* refers to the offset, in number of bytes, where you want to position the file pointer. Therefore, the first position in the file is numbered 0, not 1.

If no file is open when {SETPOS} is encountered (or some other problem occurs), macro execution begins with the next command in the same cell as {SETPOS}. If {SETPOS} succeeds, the rest of that cell's commands are ignored, and execution continues in the next row of the macro.

For an example using {SETPOS}, see {READ}.

See Also
{ONERROR}

{SETPROPERTY Property, Setting}

{SETPROPERTY} alters the properties of the active object (use {SELECTBLOCK}, {SELECTFLOAT}, or {SELECTOBJECT} to select objects).

To find *Property*, view the object and use the name of the control that sets the property. If the control name is more than one word, connect the words with underscores (_). See Property Reference for a list of properties you can use.

Examples

`{SETPROPERTY "Text_Color", "3"}` sets the selected block's Text Color property to the third color on the notebook palette (the first color is 0).

The following macro selects the SpeedButton named PushButton1 and renames it Button1.

```
name_float    {SELECTFLOAT "PushButton1"}
              {SETPROPERTY "Object_Name","Button1"}
```

See Also

Using Macros

Macro Command Descriptions

{SHIFT+Key <Num>}

{SHIFT} emulates holding down the Shift key while pressing a keystroke. It's handy for selecting blocks, portions of a text box, or parts of a formula being edited.

You can combine {ALT}, {CTRL}, and {SHIFT}. For example, `{CTRL+SHIFT+D}` is equivalent to pressing Ctrl+Shift+D, the Date keys.

Examples

`{SHIFT+DOWN 5}` emulates pressing the Shift key while moving down five cells.

`{EDIT}{END}{SHIFT+LEFT 3}` edits the active cell, and selects the last three characters of the cell contents.

See Also

Using Macros

Macro Command Descriptions

{SPEEDFILL}

{SPEEDFILL} is equivalent to the SpeedFill button on the SpeedBar. It fills the selected block with sequential data, based on entries in the upper-left portion of the block.

See Also

Using Macros

Macro Command Descriptions

{SPEEDFORMAT FmtName, NumFmt?, Font?, Shading?, TextColor?, Align?, LineDraw?, AutoWidth?, ColHead?, ColTotal?, RowHead?, RowTotal?}

{SPEEDFORMAT} applies the format *FmtName* to the selected block; it is equivalent to the SpeedFormat button on the SpeedBar. The arguments *NumFmt?* through *RowTotal?* each specify a part of the format to apply; use 1 to apply the part or 0 to omit the part.

See Also

{SETOBJECTPROPERTY}

{SETPROPERTY}

Using Macros

Macro Command Descriptions

{SPEEDSUM Block}

{SPEEDSUM} is equivalent to selecting a block and choosing the SpeedSum button from the SpeedBar. *Block* includes rows and/or columns to sum, plus adjacent empty cells to hold the results; the default *Block* is the current selection.

See Also

Using Macros

Macro Command Descriptions

{STEP}

{STEP} is equivalent to the Debug key, Shift+F2.

See Also
Using Macros
Macro Command Descriptions

{STEPOFF}

{STEPOFF} exits Debug mode, which runs the macro in slow-motion for debugging. The macro then runs at normal speed.

See {STEPON} for more information.

See Also

Using Macros

Macro Command Descriptions

{STEPON}

{STEPON} activates Debug mode, which runs macros one step at a time for debugging. When Quattro Pro encounters a {STEPON} command, it pauses and waits for the user to press any key or click the mouse before it runs the next macro command and pauses again. Debug mode continues until {STEPOFF} is encountered or the macro ends.

{STEPON} is like using the Debug key (Shift+F2) to enter Debug mode, but you can embed it within a macro. The Debug key must be used before a macro begins executing.

Example

This example uses {STEPON} and {STEPOFF} to debug the last routine of a macro. Notice that the empty pairs of braces, placed at the beginning and end of the routine, make it easy to add {STEPON} and {STEPOFF} commands when there is a problem, because you can delete them when you want to debug, and insert them when you don't.

```
_clean_up     {; Save notebook to disk and say
                goodbye}
              { }{STEPON}
              {FileSave}
              {BRANCH _quit_msg}

_quit_msg     { }{STEPOFF}
              {QUIT}
```

See Also

Using Macros

Macro Command Descriptions

{Subroutine <ArgumentList>}

{*Subroutine*} calls the subroutine of the specified name, passing along any arguments given in *ArgumentList*. (Those arguments are then defined within the subroutine using the {DEFINE} command.)

Any macro can be a subroutine. The macro commands in the subroutine run until {RETURN}, a blank cell, or a value is encountered. Then Quattro Pro returns control to the calling routine, continuing execution with the next macro command in the same cell. If the subroutine ends with {QUIT}, both the subroutine and the calling macro(s) end.

Example

The following example invokes a subroutine that forces the user to verify printing twice before it begins.

```
_print          {GETLABEL "Do you want to print this (Y/N)?",ans_cell}
                {IF @UPPER(ans_cell)="Y"}{_chk_twice}
                {HOME}

_chk_twice      {; Double-check the print request}
                {GETLABEL "Are you certain (Y/N)?",ans_cell}
                {IF @UPPER(ans_cell)="N"} {RETURN}
                {GETLABEL "Ready to print now (Y/N)?",ans_cell}
                {IF @UPPER(ans_cell)="Y"}{Print.DoPrint}
                {RETURN}

ans_cell        Y
```

See Also

{BRANCH}

{DISPATCH}

{TAB <Number>}

{TAB}, like {BIGRIGHT}, is equivalent to Ctrl+Right Arrow or Tab; it selects the leftmost cell of the screen that's to the right of the current one.

The optional argument *Number* specifies how many times to repeat the operation; for example, `{TAB 2}` is equivalent to pressing Tab twice.

See Also
Using Macros
Macro Command Descriptions

{TABLE}

{TABLE} is equivalent to the Table key, F8, which repeats the last Data|What-If operation.

See Also
Using Macros
Macro Command Descriptions

{TERMINATE DDEChannel}

{TERMINATE} closes down a DDE conversation opened with {INITIATE}.

See Also
Using Macros
Macro Command Descriptions

{TTESTM InBlock1, InBlock2, OutBlock, <Labels(0|1)>, <Alpha>, <Difference>}

{TTESTM} performs a Student's t-Test using two indepependent (rather than paired) samples with equal variances. {TTESTM} is equivalent to the t-Test analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{UNDO}

{UNDO} "takes back" the last command given and restores the previous state for most commands.

See Also

Using Macros

Macro Command Descriptions

{UP <Number>} and {U <Number>}

{UP} and {U} are equivalent to the Up Arrow key. The optional argument *Number* moves the selector up the corresponding number of rows. You can use cell references or block names as arguments.

Example

`{UP}{UP}{UP}`	moves the selector up three rows.
`{UP 4}`	moves the selector up four rows.
`{UP C13}`	moves the selector up the number of rows specified in cell C13.
`{UP temp}`	moves the selector up the number of rows specified in the first cell of the block named temp.

See Also

Using Macros

Macro Command Descriptions

{VLINE Distance}

{VLINE} scrolls the active notebook vertically by *Distance* rows. If the number is positive, it scrolls down; if negative, it scrolls up. {VLINE} doesn't move the selector; only the view of the notebook is altered. Use {DOWN} or {UP} to move the selector vertically.

Examples

`{VLINE 11}`	scrolls the display 11 rows down.
`{VLINE -4}`	scrolls the display 4 columns up.

See Also

{HLINE}
{HPAGE}
{VPAGE}

{VPAGE Distance}

{VPAGE} scrolls the active notebook vertically by *Distance* screens. If the number is positive, it scrolls down; if negative, it scrolls up. {VPAGE} doesn't move the selector; only the view of the notebook is altered. Use the commands {PGDN} or {PGUP} to move the selector vertically.

See Also

{HLINE}
{HPAGE}
{VLINE}

{WAIT DateTimeNumber}

{WAIT} pauses macro execution until the time corresponding to *DateTimeNumber* arrives. *DateTimeNumber* is a standard date/time serial number (both date and time portions must be included). If the current date is already later than the date specified in *DateTimeNumber*, macro execution continues immediately.

While execution is suspended, the macro is inactive, a WAIT indicator displays on the status line, and normal notebook operation can be restored only by pressing Ctrl+Break.

Examples

The following macro beeps, displays a "Go home" message, and suspends all execution until 8:00 a.m. the next day:

```
{BEEP 3}
{PUTCELL "Time to go home!"}
{DOWN}
{WAIT @TODAY+1+@TIMEVALUE("8:00 AM")}
```

This macro waits for one day:

```
{WAIT @NOW+1}
```

This macro uses {WAIT} to alert the user by sounding five short beeps, spaced two seconds apart:

```
\H              {FOR counter,1,5,1,_loop_here}

_loop_here      {BEEP 3}
                {WAIT @NOW+@TIME(0,0,2)}
counter 6
```

See Also

Using Macros

Macro Command Descriptions

{WINDOW<Number>}

{WINDOW} switches to the specified open notebook window (1-9). This command is for compatibility with Quattro Pro for DOS; use {ACTIVATE} to activate windows when developing macros for Quattro Pro for Windows.

The argument *Number* is optional; {WINDOW} without an argument is equivalent to the Pane key, F6.

See Also
{ACTIVATE}
{CHOOSE}

{WINDOWSOFF}

{WINDOWSOFF} disables normal screen updating during macro execution when Quattro Pro's Macro Suppress-Redraw property is set to None. It can significantly speed up execution for most macros because it saves Quattro Pro the time normally needed to redraw the screen each time a cell changes. Quattro Pro cancels it once the macro stops executing, so the user isn't "locked out" of the screen. To cancel its effect within the same macro, use {WINDOWSON}.

Use {WINDOWSOFF} with {PANELOFF} to completely disable normal screen updating.

Caution: After a {WINDOWSOFF} command, avoid letting users point to a block in response to an Edit command. The selector may be in a different cell than the "frozen" display indicates. If users must point to a block, precede it with a {WINDOWSON} command.

Example

The following macro uses {WINDOWSOFF} and {WINDOWSON} to turn off screen updating while Quattro Pro sorts a list of vendors with the block name vendor_name, thereby speeding up the sort operation.

```
sort_blk        vendor_name
key_nm          vendor_name

\W              {QGOTO}sort_message~
                {WINDOWSOFF}
                {_sort vendor_name}
                {WINDOWSON}

_sort           {DEFINE sort_blk}
                {Sort.Block @@(sort_blk)}
                {BlockCopy sort_blk,key_nm}
                {Sort.Key_1 @@(key_nm)}
                {Sort.Order_1 "Ascending"}
                {Sort.Go}

sort_message    SORT IS IN PROGRESS

vendor_name     General Cement Co.
                Alveoli Mfg., Inc.
                Sandab Development
                Consolidated Dust
```

See Also

Using Macros

Macro Command Descriptions

{WINDOWSON}

{WINDOWSON} reenables normal screen updating during macro execution, canceling the effects of a previous {WINDOWSOFF}. However, the screen won't be updated until {CALC} is encountered or the macro ends. If {WINDOWSON} is called when screen updating is already in effect, the command is ignored.

See {WINDOWSOFF} for an example using {WINDOWSON}.

See Also
Using Macros
Macro Command Descriptions

{WRITE String,<String2>,<String3,...>}

{WRITE} copies *String* to a file opened with the {OPEN} command, starting at the location of the file pointer. The file pointer is advanced to the position following the last character written, and the file's size increases if necessary. See {GETPOS} for a discussion of the file pointer.

{WRITE} doesn't place the carriage return and linefeed characters at the ends of lines. To include these characters, use {WRITELN}.

String can be a quoted string or text formula. If using more than one *String*, separate them with commas. For example, to write the label Dear Ms. followed by the contents of B6 into the file, use the following:

```
{WRITE "Dear Ms. ",+B6}
```

You can use an unlimited number of *String* arguments with {WRITE}.

If no file is open, or if there is insufficient room on the disk to increase the file's size, {WRITE} fails. Macro execution then continues with the first command after {WRITE} (in the same cell). If {WRITE} is successful, the rest of that cell's commands are ignored, and execution continues on the next row below.

If you try to reference a cell that doesn't contain a label, an error message displays.

Caution: After you finish accessing a file with {WRITE}, it **must** be closed with {CLOSE}. Otherwise, the file could become corrupted if the computer is turned off or otherwise interrupted.

Example

The following example creates a text file with fixed-length fields that will serve as a phone list database. After the macro runs, the file will look like this:

```
Golden, David 415-555-7774;Hack, Edmund  201-555-3511;Hall, Sue Ann 617-
555-5678
```

See the description of {READ} to see how to read a file like this.

```
\K    {OPEN "A:PHONEDIR.PRN",W}
      {WRITE "Golden, David "}
      {WRITE "415-555-7774;"}
      {WRITE "Hack, Edmund  "}
      {WRITE "201-555-3511;"}
      {WRITE "Hall, Sue Ann "}
      {WRITE "617-555-5678"}
      {CLOSE}
```

See Also

Using Macros

Macro Command Descriptions

{WRITELN String,<String2>,<String3,...>}

{WRITELN} copies *String*(s) to a file opened with {OPEN}, starting at the location of the file pointer, and ends the string(s) with the carriage-return and linefeed characters. The file pointer advances to the position following the last character written, and the file's size increases if necessary. (See {GETPOS} for a discussion of the file pointer.)

To enter a blank line in the file, use {WRITELN ""}. To enter characters without carriage-return/linefeed characters, use {WRITE}.

String can be a quoted string or a text formula. If using more than one *String*, separate them with commas. For example, to write the label `Dear Ms.` followed by the contents of B6 into the file you'd use

```
{WRITELN "Dear Ms. ",+B6}
```

A carriage return and linefeed character are placed at the end of all strings combined, not after each. You can use an unlimited number of *String* arguments with {WRITELN}.

If no file is open, or if there is insufficient room on the disk to increase file's size, {WRITELN} fails. Macro execution then continues with the first command after {WRITELN} in the same cell. If {WRITELN} succeeds, the rest of that cell's commands are ignored, and execution continues on the next row below.

Caution: After you finish accessing a file with {WRITELN}, you **must** close the file with {CLOSE}. Otherwise, the file could become corrupted if the computer is turned off or otherwise interrupted.

Example

The following example shows how {WRITELN} is used to write a line of text to the file TEST.TXT. This line is terminated by the CR (carriage-return) and LF (linefeed) characters.

```
\Z    {OPEN "A:TEST.TXT",W}
      {WRITELN "This is a short line."}
      {CLOSE}
```

See Also

{WRITE}

{ZOOM}

{ZOOM} maximizes and restores the active window.

This command is for compatibility with Quattro Pro for DOS; use {WindowMaximize} and {WindowRestore} when developing macros for Quattro Pro for Windows.

See Also

Using Macros

Macro Command Descriptions

{ZTESTM InBlock1, InBlock2, OutBlock, <Labels(0|1)>, <Alpha>, <Difference>, <Variance1>, <Variance2>}

{ZTESTM} performs a two-sample z-Test for means, assuming known variances for each sample. {ZTESTM} is equivalent to the z-Test analysis tool.

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Descriptions

{Align.*Option*}

Command equivalent	Equivalent to ...
{Align.Bottom}	Dialog|Align|Bottom, Draw|Align|Bottom
{Align.Horizontal_Center}	Dialog|Align| Horizontal Center, Draw|Align|Horizontal Center
{Align.Horizontal_Space *Value*}	Dialog|Align|Horizontal Space
{Align.Left}	Dialog|Align|Left, Draw|Align|Left
{Align.Right}	Dialog|Align|Right, Draw|Align|Right
{Align.Top}	Dialog|Align|Top, Draw|Align|Top
{Align.Vertical_Center}	Dialog|Align|Vertical Center, Draw|Align|Vertical Center
{Align.Vertical_Space *Value*}	Dialog|Align|Vertical Space

{Align.*Option*} is the command equivalent for Draw|Align and Dialog|Align; its action depends on whether a graph or dialog window is active.

Value is the amount of space to leave between controls, in pixels.

See Also

Aligning Objects

Aligning Controls

Using Macros

Macro Command Descriptions

{Application.*Property*}

{Application.Property} is equivalent to Property|Application (the application Object Inspector). The next table lists the possible settings for Property. Items marked with an asterisk (*) appear only in Developer mode.

To display a property description with syntax, choose that property in the following list:

Property	Property|Application option
Current_File	none
Delay_Time	Delay Time
Display	Display
Enable_Inspection	Enable Inspection*
International	International
Macro	Macro
SpeedBar	none
SpeedBars	SpeedBars
Startup	Startup
Title	Title*

See Also

Property|Application
Using Developer Mode
Using Macros
Macro Command Descriptions

{Application.Current_File}

{Application.Current_File} returns the name of the active notebook. This command equivalent is only used with @COMMAND and has no equivalent menu command.

See Also

Application.*Property*

Property|Application

@COMMAND

Using Macros

Macro Command Descriptions

{Application.Delay_Time *Integer*}

{Application.Delay_Time} is equivalent to the application property Delay Time, which sets the delay time after which drag and drop is activated when you click and hold in a selected block. *Integer* is a value in milliseconds; the default is 500.

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.Display<.*Option*>}

{Application.Display<.*Option*>}	Equivalent to Property|Application...
{Application.Display "*Clock, SpeedBar, InputLine, Status, BlockSyntax*"}	...|*Display*
{Application.Display.Clock_Display International|None|Standard}	...|Display|Clock Display
{Application.Display.Range_Syntax "A..B:A1..B2"|"A:A1..B:B2"}	...|Display|3-D Syntax
{Application.Display.Show_InputLine Yes|No}	...|Display|Show Input Line
{Application.Display.Show_StatusLine Yes|No}	...|Display|Show Status Line
{Application.Display.Show_Toolbar Yes|No}	...|Display|Show SpeedBar

{Application.Display.*Option*} is equivalent to Property|Application|Display, which lets you specify block syntax and display parts of the Quattro Pro user interface. The arguments of {Application.Display} (which sets all options of the Display property in one command) use the same syntax as those in the {Application.Display.*Option*} commands. For example,the *Clock* argument can be International, None, or Standard, the same settings that {Application.Display.Clock_Display} accepts.

Examples

The following macro command hides the time, hides the standard SpeedBar, displays the input line and status line, and sets the block syntax to standard.

```
{Application.Display "None,No,Yes,Yes,A..B:A1..B2"}
```

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.Enable_Inspection Yes|No}

{Application.Enable_Inspection} enables (Yes) or disables (No) Object Inspector menus. It is the command equivalent for Property|Application|Enable Inspection, only available in Developer mode.

See Also

Using Developer Mode

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.International<.*Option*>}

{Application.International<.*Option*>}	Equivalent to Property|Application...
{Application.International "*CurSym, Currency, Placement, Negative, Punctuation, DateFmt, TimeFmt, Language, Conversion*"}	...|International
{Application.International.Currency "Windows Default"|"Quattro Pro/Windows"}	...|International|Currency
{Application.International.Currency_Symbol *Symbol*}	...|International|Currency Symbol
{Application.International.Date_Format *String*}	...|International|Date Format
{Application.International.Language *String*}	...|International|Language
{Application.International.LICS_Conversion Yes|No}	...|International|Conversion
{Application.International.Negative Signed| Parenthesis}	...|International|Negative Values
{Application.International.Placement Prefix| Suffix}	...|International|Placement
{Application.International.Punctuation "1 234,56 (a1.a2)" | "1 234,56 (a1;a2)" | "1 234.56 (a1,a2)" | "1 234.56 (a1;a2)" | "1,234.56 (a1,a2)"| "1,234.56 (a1;a2)" | "1.234,56 (a1;a2)" | "1.234,56 (a1.a2)" | "Windows Default"}	...|International|Punctuation
{Application.International.Time_Format *String*}	...|International|Time

{Application.International.*Option*} is equivalent to Property|Application|International, which lets you specify the punctuation, sort order, and numeric formats used by Quattro Pro. The arguments of {Application.International} (which sets all options of the International property in one command) use the same syntax as those in the {Application.International.*Option*} commands. For example, the *Placement* argument can be Prefix or Suffix, the same settings that {Application.International.Placement} accepts.

Examples

The following macro command sets the currency symbol to $, specifies that the Quattro Pro currency format is used, places the currency symbol before values, sets the punctuation, sets the date and time formats to Windows defaults, sets the sort order to English, and disables LICS conversion. The entire string must be enclosed within a set of quotes. Individual items within the string that contain spaces or punctuation must be enclosed in two sets of quotes, as shown in the next example. (Enter all of the example into one cell.)

```
{Application.International "$, ""Quattro Pro/Windows"",Prefix,
Parentheses,""1,234.56 (a1,a2)"", ""MM/DD/YY (MM/DD)"", ""HH:MM:SS
(HH:MM)"", ""Quattro Pro/Windows"", ""English (American)"",No"}
```

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.Macro<.*Option*>}

{Application.Macro<.*Option*>}	Equivalent to Property| Application...
{Application.Macro "*MacSuppress, SlashKey*"}	...|Macro
{Application.Macro.Macro_Redraw Both|None|Panel|Window}	...|Macro|Macro Suppress-Redraw
{Application.Macro.Slash_Key *MenuName*}	...|Macro|Slash Key

{Application.Macro.*Option*} is equivalent to Property|Application|Macro, which lets you control screen updates and display alternative menu systems. The arguments of {Application.Macro} (which sets all options of the Macro property in one command) use the same syntax as those in the {Application.Macro.*Option*} commands.

Example

The following macro command specifies that windows shouldn't display when a macro runs and makes the slash key display the Quattro Pro for DOS menu system.

```
{Application.Macro "Window,Quattro Pro - DOS"}
```

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.SpeedBar <*SpeedBarName1,SpeedBarName2,...*>}

{Application.SpeedBar *SpeedBarName*} displays or removes a custom SpeedBar (below the standard Quattro Pro SpeedBar). To remove the custom SpeedBar, use {Application.SpeedBar ""}.

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.SpeedBars Yes|No}

{Application.SpeedBars Yes|No} saves the SpeedBar arrangement when exiting Quattro Pro.

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.Startup<.*Option*>}

{Application.Startup<.*Option*>}	Equivalent to Property|Application...
{Application.Startup "*StartDir, AutoFile, StartMacro, FileExt,UseBeep, UseUndo, CompatibleKeys, MoveCellSelectorOnEnter*"}	...|Startup
{Application.Startup.Autoload_File *String*}	...|Startup|Autoload File
{Application.Startup.Beep Yes|No}	...|Startup|Use Beep
{Application.Startup.Compatible_Keys Yes|No}	...|Startup| Compatible Keys
{Application.Startup.File_Extension *String*}	...|Startup|File Extension
{Application.Startup.MoveCellOnEnterKey Yes|No}	...|Startup|Move Cell Selector on Enter Key
{Application.Startup.Startup_Directory *String*}	...|Startup|Directory
{Application.Startup.Startup_Macro *String*}	...|Startup|Startup Macro
{Application.Startup.Undo Yes|No}	...|Startup|Undo Enabled

{Application.Startup.*Option*} is equivalent to Property|Application|Startup, which lets you specify startup options for Quattro Pro like the autoload macro name and the initial directory to display in file controls. The arguments of {Application.Startup} (which sets all options of the Startup property in one command) use the same syntax as those in the {Application.Startup.*Option*} commands. For example, the *UseBeep* argument can be Yes or No, the same settings that {Application.Startup.Beep} accepts.

Example

The following macro command sets the starting directory to C:\FILES, sets the autoload file to START.WB1, sets the autoload macro to Init, sets the default file extension to WB1, disables the Quattro Pro beep, enables Undo, disables compatible keys, and doesn't move the cell selector when you press Enter.

```
{Application.Startup "C:\FILES,START.WB1,Init,WB1,No,Yes,No,No"}
```

See Also

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{Application.Title *Title*}

{Application.Title *Title*} sets the title appearing in the Quattro Pro title bar. It is equivalent to Property|Application|Title, only available in Developer mode.

See Also

Using Developer Mode

Application.*Property*

Property|Application

Using Macros

Macro Command Descriptions

{BlockCopy SourceBlock,DestBlock, <ModelCopy?(0|1)>,<Formulas?(0|1)>,<Values?(0|1)>, <Properties?(0|1)>,<Objects?(0|1)>,<Row/Col_Sizes?(0|1)>}

{BlockCopy} is the command equivalent for Block|Copy. It copies the source block to the specified destination. If *ModelCopy?* is 1, absolute references to cells within the copied block adjust to reflect the new location. *Formula?*, *Values?*, *Properties?*, *Object?*, and *Row/Col_Sizes?* apply only if *ModelCopy?* is 1.

See Also

Block|Copy

Using Macros

Macro Command Descriptions

{BlockDelete.*Option*}

Command equivalent	Equivalent to Block|Delete...
{BlockDelete.Columns *Block*, Entire|Partial}	...|Columns
{BlockDelete.Pages *Block*, Entire|Partial}	...|Pages
{BlockDelete.Rows *Block*, Entire|Partial}	...|Rows

{BlockDelete.*Option*} is the command equivalent for Block|Delete. It deletes entire or partial columns, rows, and pages. *Block* is the 2-D or 3-D block where material is deleted.

See Also

Block|Delete

Using Macros

Macro Command Descriptions

{BlockFill.*Option*}

Command equivalent	Equivalent to Block| Fill...
{BlockFill.Block *Block*}	...|Blocks
{BlockFill.Go}	...|OK
{BlockFill.Order Column| Row}	...|Fill|Order
{BlockFill.Series Linear | Growth | Power | Year | Month | Week | Weekday | Day | Hour | Minute | Second}	...|Fill|Series
{BlockFill.Start *Value*}	...|Fill|Start
{BlockFill.Step V*alue}*	...|Fill|Step
{BlockFill.Stop V*alue}*	...|Fill|Stop

{BlockFill.*Option*} is the command equivalent for Block|Fill. It fills *Block* with sequential data. You can use numbers, dates, times, or even formulas for *Value*.

If {BlockFill.Start} is a number or formula, you can enter one of these strings for {BlockFill.Series}:

"Linear" adds the step value to the previous value (defined at first to be the start value).

Growth" multiplies the step value by the previous value.

"Power" uses the step value as the exponent of the previous value.

If {BlockFill.Start} is a date or time, the fill operation is always linear, but you can specify the step unit as "Year", "Month", "Week", "Weekday", "Day", "Hour", "Minute", or "Second". For example, with a start value of 6/20/92, a step value of 2, and "Month" as the {BlockFill.Series *Option*} setting, the second cell in the filled block contains August.

You can enter the date and time directly as a serial number or use one of the date and time @functions.

Example

The following macro uses @DATEVALUE to enter 6/20/92 as the start value. The 3-D block to fill is B..C:B1..D4 with a step value of 2. Fill order is "Row".

```
{BlockFill.Block B:B1..C:D4}
{BlockFill.Start @DATEVALUE("6/20/92")}
{BlockFill.Step 2}
{BlockFill.Stop @DATEVALUE("12/31/2099")}
{BlockFill.Order Row}
{BlockFill.Series Month}
{BlockFill.Go}
```

See Also

Block|Fill

{BlockInsert.*Option*}

Command equivalent	Equivalent to Block| Insert...
{BlockInsert.Columns *Block,* Entire|Partial}	...|Columns
{BlockInsert.File *FileName, BeforeBlock*}	...|File
{BlockInsert.Pages *Block,* Entire|Partial}	...|Pages
{BlockInsert.Rows *Block,* Entire|Partial}	...|Rows

{BlockInsert.*Option*} is the command equivalent for Block|Insert. It inserts entire or partial columns, rows, and pages, or complete files. *Block* is the 2-D or 3-D block where material is inserted. In {BlockInsert.File}, *Filename* is inserted into the active notebook before block *BeforeBlock*.

See Also

Block|Insert

Using Macros

Macro Command Descriptions

{BlockMove SourceBlock,DestBlock}

{BlockMove} is the command equivalent for Block|Move. It moves the contents of a block from one location to another. Any data already in *DestBlock* is overwritten by the contents of *SourceBlock*. Block properties move with the data. Nonabsolute formulas are adjusted.

To move pages, use {BlockMovePages}.

See Also

Block|Move

{BlockMovePages}

Using Macros

Macro Command Descriptions

{BlockMovePages SrcPages,BeforePage}

{BlockMovePages} is the command equivalent for Block|Move Pages. It reorders pages within a notebook. Moved pages appear before *BeforePage*.

See Also

Block|Move Pages

Using Macros

Macro Command Descriptions

{BlockName.*Option*}

Command equivalent	Equivalent to Block|Names...
{BlockName.Autogenerate *Block*,*LabelsTop?*(0|1),*LabelsLeft?*(0|1),*LabelsBottom?*(0|1),*LabelsRight?*(0|1),*Intersection?*(0|1)}	...|Auto Generate
{BlockName.Create *BlockName, Block*}	...|Create
{BlockName.Delete *BlockName*}	...|Delete
{BlockName.Labels *Block*,Left|Right|Up|Down}	...|Labels
{BlockName.MakeTable *Block*}	...|Make Table
{BlockName.Reset}	...|Reset

{BlockName.*Option*} is the command equivalent for Block|Names. It creates, deletes, and displays names for contiguous and noncontiguous blocks.

BlockName is the block name to create or delete. In {BlockName.Create}, *Block* is the block to name; in {BlockName.MakeTable}, *Block* indicates where to create the name table. {BlockName.Reset} clears all block names in the notebook. If you omit the arguments for {BlockName.Create}, {BlockName.Delete}, or {BlockName.MakeTable}, the appropriate dialog box appears.

See Also

Block|Names

Using Macros

Macro Command Descriptions

{BlockReformat Block}

{BlockReformat} is the command equivalent for Block|Reformat. It adjusts word wrapping in a series of label entries (contained in *Block*) as though they were in a paragraph.

See Also

Block|Reformat

Using Macros

Macro Command Descriptions

{BlockTranspose SourceBlock,DestBlock}

{BlockTranspose} is the command equivalent for Block|Transpose. It copies *SourceBlock* to another location and reverses its rows and columns. Existing data in *DestBlock* is overwritten.

See Also

Block|Transpose

Using Macros

Macro Command Descriptions

{BlockValues SourceBlock,DestBlock}

{BlockValues} is the command equivalent for Block|Values. It copies a block to another location and converts its formulas to values. Existing data in *DestBlock* is overwritten.

See Also

Block|Values

Using Macros

Macro Command Descriptions

{ClearContents <fPageOnly? (0|1)>}

{ClearContents} is the command equivalent for Edit|Clear Contents. It erases the contents of the selected block but leaves block property settings intact.

See Also

Edit|Clear Contents

Using Macros

Macro Command Descriptions

{Consolidate.*Option*}

Command equivalent	Equivalent to Tools|Consolidator...
{Consolidate.Add_Source_Block <*Block*>}	...|Add Source Block button
{Consolidate.Destination_Block <*Block*>}	...|Add Destination Block button
{Consolidate.Function SummaryFunction}	...|Operator list
{Consolidate.Go}	...|Consolidate button
{Consolidate.Options *OutputWithFormulas?*(0|1), *LabelsInTopRow?*(0|1), *LabelsInLeftCol?*(0|1)}	...|Options button
{Consolidate.Remove *Name*}	...|Delete Consolidation button
{Consolidate.Remove_Source_Block <*Block*>}	...|Remove Source Block button
{Consolidate.Save *Name*}	...|Save Consolidation button
{Consolidate.Use *Name*}	...|Consolidation Name list

{Consolidate.*Option*} is the command equivalent for Tools|Consolidator. It combines data from multiple blocks into one using your choice of operators. *Block* defaults to the current block selected if the argument isn't supplied.

Example

The following macro adds the values in the source blocks B2..B4, C2..C3, and D2..D4, and returns values in the destination block F2..F4.

```
{Consolidate.Add_Source_Block A:B2..B4}
{Consolidate.Add_Source_Block A:C2..C3}
{Consolidate.Add_Source_Block A:D2..D4}
{Consolidate.Function SUM}
{Consolidate.Destination A:F2..F4}
{Consolidate.Options 1,0,0}
{Consolidate.Go}
{Consolidate.Save CONSOL1}
```

See Also

Using the Consolidator

Using Macros

Macro Command Descriptions

{Controls.*Option*}

Command equivalent	Equivalent to Dialog|Order...
{Controls.Order}	...|Order Controls
{Controls.OrderFrom}	...|Order From
{Controls.OrderTab}	...|Order Tab Controls
{Controls.OrderTabFrom}	...|Order Tab From

{Controls.*Option*} is equivalent to Dialog|Order. It affects selected objects in the dialog window.

See Also

Dialog|Order

Using Macros

Macro Command Descriptions

{DialogView Window}

{DialogView} lets you edit an existing dialog box, *Window*.

See Also

Dialog Box and SpeedBar Overview

Using Macros

Macro Command Descriptions

{DialogWindow.*Property*}

{DialogWindow.*Property*} is equivalent to Property|Dialog (the dialog window Object Inspector).

{DialogWindow} commands affect the active dialog window. The next table lists the possible settings for Property. To display a property description with syntax, choose that property in the following list:

Property	Property|Dialog option
Dimension	Dimension
Disabled	Disabled
Grid_Options	Grid Options
Name	Name
Position_Adjust	Position Adjust
Title	Title
Value	Value

See Also

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Dimension<.*Option*>}

{DialogWindow.Dimension<.*Option*>}	Equivalent to Property|Dialog...
{DialogWindow.Dimension "XPos, YPos, Width, Height"}	...|Dimension
{DialogWindow.Dimension.Height *Height*}	...|Dimension|Height
{DialogWindow.Dimension.Width *Width*}	...|Dimension|Width
{DialogWindow.Dimension.X *XPos*}	...|Dimension|X Pos
{DialogWindow.Dimension.Y *YPos*}	...|Dimension|Y Pos

{DialogWindow.Dimension.*Option*} is equivalent to the dialog window property Dimension, which lets you move and resize the active dialog window. Each argument is specified in pixels. *XPos* and *YPos* specify the distance in pixels from the left side of the Quattro Pro window and bottom of the input line, respectively.

Example

The following macro command positions the active dialog window two pixels from the left edge of the Quattro Pro window, five pixels below the input line, sets the width to 150 pixels, and sets the height to 250 pixels.

```
{DialogWindow.Dimension "2,5,150,250"}
```

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Disabled Yes|No}

{DialogWindow.Disabled Yes|No} disables (Yes) or enables (No) the active dialog box or SpeedBar. This command only works when the user is viewing a dialog box or SpeedBar; it doesn't work when editing one.

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Grid_Options *GridSize, ShowGrid, SnapToGrid*}

{DialogWindow.Grid_Options *GridSize, ShowGrid, SnapToGrid*} sets the grid size of the active dialog window. Use *GridSize* to specify the distance between grid points, in pixels; *ShowGrid* specifies whether the grid is visible; *SnapToGrid* specifies whether the grid is active.

Example

The following macro sets the distance between grid points to 10, hides the grid, and enables it.

```
{DialogWindow.Grid_Options "10,No,Yes"}
```

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Name *Name*}

{DialogWindow.Name *Name*} sets the name of the active dialog window. This name is used by macro commands, @functions, and link commands to identify the dialog box (or SpeedBar).

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Position_Adjust *Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer*}

{DialogWindow.Position_Adjust *Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer*} specifies how the active dialog box resizes when the Quattro Pro window is resized.

See Also

Position AdjustProperty

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Title *String*}

{DialogWindow.Title *String*} specifies the title that appears on the dialog box when the user is viewing it (the title doesn't appear when editing the dialog box).

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

Using Macros

Macro Command Descriptions

{DialogWindow.Value *String*}

{DialogWindow.Value *String*} sets the initial settings of the dialog box (or SpeedBar). You can use it with @COMMAND to find the current settings of the dialog box. *String* is a comma separated list of settings. Each setting sets the initial value of one control. Control values appear in this list if their Process Value property is set to Yes. You can set the order of the settings while editing the dialog box.

Example

The following macro command sets the initial values of a dialog box with three controls. Each setting maps to one control.

```
{DialogWindow.Value "25000,5,1st of month"}
```

See Also

{DialogWindow.*Property*}

Active Dialog Window Properties

@COMMAND

Using Macros

Macro Command Descriptions

{EditClear}

{EditClear} is the command equivalent for Edit|Clear. It erases the contents and properties of the current block, deletes selected objects from dialog and graph windows, and deletes selected floating objects. To erase a block while leaving its properties intact, use {ClearContents}.

See Also

Edit|Clear

{ClearContents}

Using Macros

Macro Command Descriptions

{EditCopy}

{EditCopy} is the command equivalent for Edit|Copy. It copies the selected object to the Clipboard.

See Also

Edit|Copy

Using Macros

Macro Command Descriptions

{EditCut}

{EditCut} is the command equivalent for Edit|Cut. It deletes the selected object and moves it to the Clipboard.

See Also

Edit|Cut

Using Macros

Macro Command Descriptions

{EditGoto Block, <Extend?(0|1)>}

{EditGoto} is the command equivalent for Edit|Goto. It selects and displays *Block* within spreadsheet pages, but not the Graphs page.

See Also

Edit|Goto

Using Macros

Macro Command Descriptions

{EditPaste}

{EditPaste} is the command equivalent for Edit|Paste. It copies data and its properties from the Clipboard into the notebook.

To paste only values or properties, use {PasteSpecial}. {PasteLink} creates a live DDE link, and {PasteFormat} adds many types of data from other applications (including embedded OLE objects).

See Also

Edit|Paste

{PasteSpecial}

{PasteLink}

{PasteFormat}

Using Macros

Macro Command Descriptions

{ExportGraphic Filename, <GrayScale?(0|1)>, <Compression?(0|1)>}

{ExportGraphic} is the command equivalent for Draw|Export. It saves selected graphic objects to one of several file types with optional gray-scaling and compression.

See Also

Draw|Export

Using Macros

Macro Command Descriptions

{FileClose...}

Command equivalent	Equivalent to ...
{FileClose <*DoSave?* (0|1)>}	File|Close
{FileCloseAll <*DoSave?* (0|1)>}	File|Close All

{FileClose} closes all views of the active notebook; {FileCloseAll} closes all open notebooks. The optional argument *DoSave?* indicates whether to display a save prompt before closing files with changes. Use 1, the default, to prompt for changes; 0 suppresses save prompts.

See Also

File|Close

File|Close All

{FileSave}

Using Macros

Macro Command Descriptions

{FileCombine Filename, <Blocks>, Add | Subtract | Multiply | Divide | Copy}

{FileCombine} is the command equivalent for Tools|Combine. If you use the "Copy" option, it copies all or part of a notebook into the active notebook (starting at the selected cell). Omit *Blocks* to combine an entire file. Use "Add", "Subtract", "Multiply", or "Divide" to perform mathematical operations; the incoming data operates on existing data.

See Also

Tools|Combine

Using Macros

Macro Command Descriptions

{FileExit <DoSave? (0|1)>}

{FileExit} is the command equivalent for File|Exit. It closes Quattro Pro. The optional argument *DoSave?* indicates whether to display a save prompt before closing files with changes. Use 1, the default, to prompt for changes; 0 suppresses save prompts.

See Also

File|Exit

{FileSave...}

Using Macros

Macro Command Descriptions

{FileExtract Formulas|Values, Blocks, Filename <,Replace|Backup|Confirm>}

{FileExtract} is the command equivalent for Tools|Extract. It saves part of a notebook to a separate file, leaving the original file intact. Use "Formulas" to retain formulas; use "Values" to convert formulas to values. The optional argument--"Replace", "Backup", or "Confirm"--indicates how to treat an existing file with the same name (without displaying a prompt).

See Also

Tools|Extract

Using Macros

Macro Command Descriptions

{FileImport Filename, "ASCII Text File"|"Comma and "" Delimited File"|"Only Commas"|"Parse Expert"}

{FileImport "*Filename*","ASCII Text File"|"Comma and "" Delimited File"|"Only Commas"|"Parse Expert"} is the command equivalent for Tools|Import. It copies a text or delimited-data file into the active page of a notebook.

See Also

Tools|Import

Using Macros

Macro Command Descriptions

{FileNew}

{FileNew} is the command equivalent for File|New. It opens a blank notebook.

See Also

File|New

Using Macros

Macro Command Descriptions

{FileOpen Filename<,"Open Supporting"|"Update References"|"None">}

{FileOpen} is the command equivalent for File|Open. It opens the specified file.

See Also

File|Open

Using Macros

Macro Command Descriptions

{FileRetrieve Filename<,"Open Supporting"|"Update References"|"None">}

{FileRetrieve} is the command equivalent for File|Retrieve. It loads a notebook into the active notebook, replacing any existing data there.

See Also

File|Retrieve

Using Macros

Macro Command Descriptions

{FileSave...}

Command equivalent	Equivalent to...
{FileSave <Replace|Backup|Confirm>}	File|Save
{FileSaveAll <Replace|Backup|Confirm>}	File|Save All
{FileSaveAs *Filename* <,Replace | Backup | Confirm>}	File|Save As

{FileSave} saves the active notebook, {FileSaveAll} saves all open notebooks, and {FileSaveAs} lets you save the active notebook under another name (*Filename*). The optional argument--"Replace", "Backup", or "Confirm"--indicates how to treat a previous version of the file (without displaying a prompt).

Example

To close all files and save without confirmation, use this macro:

```
{FileSaveAll Replace}
{FileCloseAll 0}
```

See Also

File|Save
File|Save All
File|Save As
Using Macros
Macro Command Descriptions

{FloatOrder.*Option*}

Command equivalent	Equivalent to Block|Object Order...
{FloatOrder.ToBack}	...|Send To Back
{FloatOrder.Backward}	...|Send Backward
{FloatOrder.ToFront}	...|Bring To Front
{FloatOrder.Forward}	...|Bring Forward

{FloatOrder.*Option*} is the command equivalent for Block|Object Order. It works on selected objects to arrange layers of floating graphs and other floating objects.

See Also

Block|Object Order

Using Macros

Macro Command Descriptions

{Frequency.*Option*}

Command equivalent	Equivalent to Data|Frequency...
{Frequency.Bin_Block *Block*}	...|Bin Block
{Frequency.Go}	...|OK
{Frequency.Reset}	...|Reset
{Frequency.Value_Block *Block*}	...|Value Blocks

{Frequency.*Option*} is the command equivalent for Data|Frequency. It counts the number of cases in the value *Block* that fall within each interval specified in the bin *Block*. Use {Frequency.Bin_Block} and {Frequency.Value_Block}, then {Frequency.Go}. You can use {Frequency.Reset} before or after the other commands to clear current settings.

Example

The following macro counts the data in block C1..E13 of page A and groups it according to the intervals given in G1..G7; frequencies display in column H.

```
{Frequency.Value_Block A:C1..E13}
{Frequency.Bin_Block A:G1..G7}
{Frequency.Go}
```

See Also

Data|Frequency

Using Macros

Macro Command Descriptions

{GraphCopy FromGraph, DestGraph, <Style?(0|1)>, <Data?(0|1)>, <Annotations?(0|1)>}

{GraphCopy} is the command equivalent for Graph|Copy. It copies the style, data, and/or annotation objects from one graph to another within a notebook (but not between notebooks).

See Also

Graph|Copy

Using Macros

Macro Command Descriptions

{GraphDelete Name}

{GraphDelete} is the command equivalent for Graph|Delete. It deletes the specified graph from the active notebook.

See Also

Graph|Delete

Using Macros

Macro Command Descriptions

{GraphEdit Name}

{GraphEdit} is the command equivalent for Graph|Edit. It displays the specified graph in a graph window for editing.

See Also

Graph|Edit

Using Macros

Macro Command Descriptions

{GraphNew Name<,UseCurrentBlock?(0|1)>}

{GraphNew} is the command equivalent for Graph|New. It creates a new graph and displays it in a graph window. Any selected data is shown in the graph.

See Also

Graph|New

Using Macros

Macro Command Descriptions

{GRAPHPAGEGOTO}

{GRAPHPAGEGOTO} displays the Graphs page of the active notebook (like pressing Shift+F5 when a spreadsheet page is active). When the Graphs page is active, you can use {SELECTOBJECT} to select icons, and other object commands to manipulate them.

You can use {SELECTBLOCK} to move from the Graphs page to a spreadsheet page.

See Also

{SELECTOBJECT}

{SELECTBLOCK}

{GOTO}

{QGOTO}

Using Macros

Macro Command Descriptions

{GraphSettings.Titles Main, Sub, X-Axis, Y-Axis, Y2-Axis}

{GraphSettings.Titles} is equivalent to Graph|Titles, which sets the titles of the active graph (or selected floating graph or graph icon). Each argument is a string; to reset a title, use an empty string ("").

Example

The following macro command displays the graph Profit99 in a graph window and sets its main title and subtitles. The empty strings ("") indicate that there are no axis titles.

```
{GraphEdit Profit99}
{GraphSettings.Titles "Projected Profits","1999","","",""}
```

See Also

Graph|Titles

Using Macros

Macro Command Descriptions

{GraphSettings.Type *Type*<,*Class*>}

{GraphSettings.Type} is equivalent to Graph|Type, which lets you specify how the data in a graph is displayed. It affects the active graph (or graph icon or floating graph). *Class* is an optional argument that specifies the class of graph type being used. *Class* can be one of five settings: 2-D, 3-D, Combo, Rotate, and Text.

When *Class* is "Text", the only *Type* setting available is also "Text". Use this syntax to specify a text graph: {GraphSettings.Type "Text,Text"}. To display *Type* setting options for the other *Class* settings, choose a setting from this list:

- 2-D
- 3-D
- Combo
- Rotate

Example

`{GraphSettings.Type "Pie"}` and `{GraphSettings.Type "Pie,2-D"}` make the active graph a 2-D pie graph.

See Also

Graph|Type

Using Macros

Macro Command Descriptions

{GraphSettings.Type *Type* <,2-D>}

{GraphSettings.Type} is equivalent to Graph|Type, which lets you specify how the data in a graph is displayed. The following table lists *Type* settings for 2-D graphs:

Command equivalent	Equivalent to Graph|Type...
{GraphSettings.Type "100 Stacked Bar,2-D"}	...|2-D|100% Stacked Bar
{GraphSettings.Type "100 Stacked Line,2-D"}	...|2-D|100% Stacked Bar Comparison
{GraphSettings.Type "Area,2-D"}	...|2-D|Area
{GraphSettings.Type "Bar,2-D"}	...|2-D|Bar
{GraphSettings.Type "Column,2-D"}	...|2-D|Column
{GraphSettings.Type "Doughnut,2-D"}	...|2-D|Doughnut
{GraphSettings.Type "HiLo,2-D"}	...|2-D|High Low
{GraphSettings.Type "Line,2-D"}	...|2-D|Line
{GraphSettings.Type "Pie,2-D"}	...|2-D|Pie
{GraphSettings.Type "Polar Radar,2-D"}	...|2-D|Radar
{GraphSettings.Type "Stacked Bar,2-D"}	...|2-D|Stacked Bar
{GraphSettings.Type "Stacked Line,2-D"}	...|2-D|Stacked Bar Comparison
{GraphSettings.Type "Variance,2-D"}	...|2-D|Variance
{GraphSettings.Type "XY,2-D"}	...|2-D|XY

These are other available settings for *Class* besides "Text" and "2-D":

3-D

Combo

Rotate

Example

`{GraphSettings.Type "Pie"}` and `{GraphSettings.Type "Pie,2-D"}` make the active graph a 2-D pie graph.

See Also

{GraphSettings.Type}

Graph|Type

Using Macros

Macro Command Descriptions

{GraphSettings.Type *Type* <,3-D>}

{GraphSettings.Type} is equivalent to Graph|Type, which lets you specify how the data in a graph is displayed. The following table lists *Type* settings for 3-D graphs:

Command equivalent	Equivalent to Graph|Type...
{GraphSettings.Type "2DHalf Bar,3-D"}	...|3-D|2.5-D Bar
{GraphSettings.Type "3D100 Stacked Bar,3-D"}	...|3-D|100% Stacked Bar
{GraphSettings.Type "3D Area,3-D"}	...|3-D|Area
{GraphSettings.Type "3D Bar,3-D"}	...|3-D|Bar
{GraphSettings.Type "3D Column,3-D"}	...|3-D|Column
{GraphSettings.Type "3D Contour,3-D"}	...|3-D|Contour
{GraphSettings.Type "3D Doughnut,3-D"}	...|3-D|Doughnut
{GraphSettings.Type "3D Pie,3-D"}	...|3-D|Pie
{GraphSettings.Type "3D Ribbon,3-D"}	...|3-D|Ribbon
{GraphSettings.Type "3D ShadedSurface,3-D"}	...|3-D|Shaded Surface
{GraphSettings.Type "3D Stacked Bar,3-D"}	...|3-D|Stacked Bar
{GraphSettings.Type "3D Step,3-D"}	...|3-D|Step
{GraphSettings.Type "3D Surface,3-D"}	...|3-D|Surface
{GraphSettings.Type "3D Unstacked Area,3-D"}	...|3-D|Unstacked Area

These are other available settings for *Class* besides "Text" and "3-D":

2-D

Combo

Rotate

Example

`{GraphSettings.Type "3D Step"}` and `{GraphSettings.Type "3D Step,3-D"}` make the active graph a 3-D step graph.

See Also

{GraphSettings.Type}

Graph|Type

Using Macros

Macro Command Descriptions

{GraphSettings.Type *Type*<,Combo>}

{GraphSettings.Type} is equivalent to Graph|Type, which lets you specify how the data in a graph is displayed. The following table lists *Type* settings for Combo graphs:

Command equivalent	Equivalent to Graph|Type...
{GraphSettings.Type "Area_bar,Combo"}	...|Combo|Area-Bar
{GraphSettings.Type "Hilo_bar,Combo"}	...|Combo|High Low-Bar
{GraphSettings.Type "Line_bar,Combo"}	...|Combo|Line-Bar
{GraphSettings.Type "Multiple 3D columns,Combo"}	...|Combo|3D columns
{GraphSettings.Type "Multiple 3D Pies,Combo"}	...|Combo|3D Pies
{GraphSettings.Type "Multiple Bar,Combo"}	...|Combo|Bar
{GraphSettings.Type "Multiple Columns,Combo"}	...|Combo|Columns
{GraphSettings.Type "Multiple Pies,Combo"}	...|Combo|Multiple Pies

These are other available settings for *Class* besides "Text" and "Combo":

2-D

3-D

Rotate

Example

`{GraphSettings.Type "Multiple Pies"}` and `{GraphSettings.Type "Multiple Pies,Combo"}` make the active graph a combo pie graph.

See Also

{GraphSettings.Type}

Graph|Type

Using Macros

Macro Command Descriptions

{GraphSettings.Type *Type*<,Rotate>}

{GraphSettings.Type} is equivalent to Graph|Type, which lets you specify how the data in a graph is displayed. The following table lists *Type* settings for Rotate (rotated) graphs:

Command equivalent	Equivalent to Graph|Type...
{GraphSettings.Type "R2D Bar,Rotate"}	...|Rotate|Rotated 2-D Bar
{GraphSettings.Type "R2D Stacked Bar,Rotate"}	...|Rotate|Rotated 2-D Stacked Bar
{GraphSettings.Type "R2D Stacked Line,Rotate"}	...|Rotate|Rotated 2-D Comparison
{GraphSettings.Type "R2D100 Stacked Bar,Rotate"}	...|Rotate|Rotated 2-D 100% Stacked Bar
{GraphSettings.Type "R2D100 Stacked Line,Rotate"}	...|Rotate|Rotated 2-D 100% Comparison
{GraphSettings.Type "R2DHalf Bar,Rotate"}	...|Rotate|Rotated 2.5-D Bar
{GraphSettings.Type "R3D bar,Rotate"}	...|Rotate|Rotated 3-D Bar
{GraphSettings.Type "R3D Stacked Bar,Rotate"}	...|Rotate|Rotated 3-D Stacked Bar
{GraphSettings.Type "R3D100 Stacked Bar,Rotate"}	...|Rotate|Rotated 3-D 100% Stacked Bar
{GraphSettings.Type "Rotated Area,Rotate"}	...|Rotate|Rotated Area
{GraphSettings.Type "Rotated Line,Rotate"}	...|Rotate|Rotated Line

These are other available settings for *Class* besides "Text" and "Rotate":

2-D

3-D

Combo

Example

`{GraphSettings.Type "R2D Bar"}` and `{GraphSettings.Type "R2D Bar,Rotate"}` make the active graph a rotated 2-D bar graph.

See Also

{GraphSettings.Type}

Graph|Type

Using Macros

Macro Command Descriptions

{GraphView <GraphName1, GraphName2,...>}

{GraphView} is equivalent to Graph|View, which displays a graph (or series of graphs) full-screen. {GraphView} without an argument displays the active graph (or graph icon or floating graph).

Example

The following macro displays the named graphs Profit90 through Profit94.

```
{GraphView Profit90,Profit91,Profit92,Profit93,Profit94}
```

See Also

Graph|View

Using Macros

Macro Command Descriptions

{GraphWindow.*Property*}

{GraphWindow.*Property*} is equivalent to Property|Graph Window (the graph window Object Inspector). Property can be Aspect_Ratio or Grid.

{GraphWindow.Aspect_Ratio *Option*} sets the aspect ratio of the active graph. *Option* can be one of the following settings: "35mm Slide", "Floating Graph", "Full Extent", "Printer Preview", or "Screen Slide".

{GraphWindow.Grid *GridSize,DisplayGrid,SnapToGrid*} sets the grid size of the active graph window. Use *GridSize* to specify the percent of graph window between grid points; *DisplayGrid* specifies whether the grid is visible; *SnapToGrid* specifies whether the grid is active.

Example

This macro sets up a 10 by 10 grid, displays the grid, and enables it.

```
{GraphWindow.Grid "10,Yes,Yes"}
```

See Also

Property|Graph Window

Using Macros

Macro Command Descriptions

{Group.*Option*}

Command equivalent	Equivalent to Tools|Define Group...
{Group.Define *GroupName*, *StartPage*, *EndPage*}	Tools|Define Group
{Group.Delete GroupName}	...|Delete
{Group.ResetNames}	Clears all group names in the notebook; no equivalent command

{Group.*Option*} is the command equivalent for Tools|Define Group, which creates and deletes page groups.

Once you've defined a page group, you can use `{Notebook.Group_Mode "On"}` to activate Group mode. Use `"Off"` to cancel Group mode.

See Also

Tools|Define Group

Using Macros

Macro Command Descriptions

{GroupObjects}

{GroupObjects} is the command equivalent for Draw|Group. It groups selected objects in a graph window so they can be treated as one object in subsequent operations. Use {UngroupObjects} to treat them independently again.

See Also

Draw|Group

{UngroupObjects}

Using Macros

Macro Command Descriptions

{ImportGraphic Filename}

{ImportGraphic} is the command equivalent for Draw|Import. It imports graphics files into a graph window.

See Also

Draw|Import

Using Macros

Macro Command Descriptions

{InsertBreak}

{InsertBreak} is the command equivalent for Block|Insert Break. It inserts a new line and a hard page break into notebook print blocks at the current selector location.

See Also

Block|Insert Break

Using Macros

Macro Command Descriptions

{InsertObject ObjectType}

{InsertObject} is equivalent to Edit|Insert Object, which inserts an OLE object into the active notebook without using the Clipboard.

Example

This macro inserts a picture created in Paintbrush into the active notebook.

```
{InsertObject "Paintbrush Picture"}
```

See Also

Edit|Insert Object

Using Macros

Macro Command Descriptions

{Invert.*Option*}

Command equivalent	Equivalent to Tools|Advanced Math |...
{Invert.Destination *Block*}	...Invert|Destination
{Invert.Go}	...Invert|OK
{Invert.Source *Block*}	...Invert|Source

{Invert.*Option*} is the command equivalent for Tools|Advanced Math|Invert. It inverts a square matrix (indicated by {Invert.Source *Block*}) and stores the invert matrix in another block (indicated by {Invert.Destination *Block*}). Use {Invert.Go} after the other two matrix-inversion command equivalents to complete the operation.

You can use this command equivalent with {Multiply.Option} to solve sets of linear equations.

See Also

Tools|Advanced Math|Invert

{Multiply.*Option*}

Using Macros

Macro Command Descriptions

{Links.*Option*}

Command equivalent	Equivalent to Tools|Update Links|...
{Links.Change *OldName*, *NewName*}	|Change Link
{Links.Delete *LinkName*|*} (* = all links)	|Delete Links
{Links.Open *LinkName*|*} (* = all links)	...|Open Links
{Links.Refresh *LinkName*|*} (* = all links)	...|Refresh Links

{Links.*Option*} is equivalent to commands on the Tools|Links menu, which refresh, change, or delete links in the active notebook.

LinkName is the name of the file being linked to. You can set *LinkName* to * to affect all links in the active notebook. If *LinkName* is omitted, the dialog box that normally performs the operation appears (and is under macro control; use {PAUSEMACRO} to pass control to the user).

Examples

`{Link.Refresh *}` refreshes all links in the active notebook.

The following macro displays the Open Links dialog box and lets the user select the name of a linked notebook to open.

```
{Links.Open}
{PAUSEMACRO}
```

See Also

Tools|Update Links

{PAUSEMACRO}

Using Macros

Macro Command Descriptions

{Multiply.*Option*}

Command equivalent	Equivalent to Tools|Advanced Math|...
{Multiply.Destination *Block*}	...Multiply|Destination
{Multiply.Go}	...|OK
{Multiply.Matrix_1 *Block*}	...Multiply|Matrix 1
{Multiply.Matrix_2 *Block*}	...Multiply|Matrix 2

{Multiply.*Option*} is the command equivalent for Tools|Advanced Math|Multiply. It multiplies one matrix ({Multiply.Matrix_1 *Block*}) by another ({Multiply.Matrix_2 *Block*}) and stores the product in another block ({Multiply.Destination *Block*}). Use {Multiply.Go} after the other matrix-multiplication command equivalents to complete the operation.

You can use this command equivalent with {Invert.Option} to solve sets of linear equations.

Example

This macro multiplies block C2..D6 by block C18..G19 and stores the results in the block with upper-left cell F1.

```
{Multiply.Matrix_1 A:C2..D6}
{Multiply.Matrix_2 A:C18..G19}
{Multiply.Destination A:F1}
{Multiply.Go}
```

See Also

Tools|Advanced Math|Multiply

{Invert.*Option*}

Using Macros

Macro Command Descriptions

{NamedStyle.*Option*}

Command equivalent	Equivalent to Edit|Define Style...
{NamedStyle.Alignment General|Left|Right|Center|Center Across Block}	...|Included Properties|Alignment
{NamedStyle.Define "*StyleName", Align?, NumericFormat?, Protection?, Lines?, Shading?, Font?, TextColor?*}	...|Define Style For
{NamedStyle.Delete *StyleName*}	...|Delete
{NamedStyle.Font "*FontName, PointSize, Bold, Italic, Underline, Strikeout*"}	...|Included Properties|Font
{NamedStyle.LineDrawing *LeftLine, TopLine, RightLine, BottomLine*}	...|Included Properties|Line Drawing
{NamedStyle.Numeric_Format *NumericFormat*}	...|Included Properties|Format
{NamedStyle.Protection Protected|Unprotected}	...|Included Properties|Protection
{NamedStyle.Shading *ForegroundColor, BackgroundColor, Pattern*}	...|Included Properties|Shading
{NamedStyle.Text_Color 0-15}	...|Included Properties|Text Color

{NamedStyle.*Option*} is equivalent to Edit|Define Style, which lets you create styles in the active notebook.

These command equivalents don't take effect until the command {NamedStyle.Define} is used to create (or modify) a style. The arguments *Align?* through *TextColor?* each specify one property to include in the style; use 1 to include the property, 0 to exclude the property.

{NamedStyle.Font} sets the new typeface and size of text in the cell. *Bold, Italic, Underline* and *Strikeout* can be "Yes" to include that type feature or "No" to omit it.

{NamedStyle.Shading} sets the shading of the cell; *ForegroundColor* and *BackgroundColor* are numbers from 0 to 15; each specifies a color on the notebook palette to use; *Pattern* is a string ("Blend1" through "Blend7").

Example

This macro creates a new style named RedNote, which makes the active block red, and sets a new font.

```
{NamedStyle.Font "Courier,10,Yes,No,No,No"}
{NamedStyle.Text_Color "4"}
{NamedStyle.Define RedNote,0,0,0,0,0,1,1}
```

See Also

Edit|Define Style

Using Macros

Macro Command Descriptions

{Notebook.*Property*}

{Notebook.*Property*} is equivalent to Property|Active Notebook (the notebook Object Inspector). Each command affects the active notebook. The next table lists the possible settings for *Property*. To display a description with syntax, choose that property in the following list:

Property	**Property|Active Notebook option**
Display	Display
Group_Mode	none
Macro_Library	Macro Library
Palette	Palette
Password	none
Password_Level	Password Level
Recalc_Settings	Recalc Settings
System	System
Zoom_Factor	Zoom Factor

See Also

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Display<.*Option*>}

{Notebook.Display<.*Option*>}	**Equivalent to Property| Active Notebook...**
{Notebook.Display "*VertScroll*, *HorizScroll*, *Tabs*"}	...|Display
{Notebook.Display.Show_HorizontalScroller Yes|No}	...|Display|Horizontal Scroll Bar
{Notebook.Display.Show_Tabs Yes|No}	...|Display|Page Tabs
{Notebook.Display.Show_VerticalScroller Yes|No}	...|Display|Vertical Scroll Bar

{Notebook.Display.*Option*} is equivalent to options of the notebook property Display.

Example

This macro command hides the vertical and horizontal scroll bars of the active notebook, and reveals the page tabs.

```
{Notebook.Display "No,No,Yes"}
```

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Group_Mode On|Off}

{Notebook.Group_Mode On|Off} activates or deactivates group mode. This command is equivalent to using the Group button (by the page tabs).

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Macro_Library Yes|No}

{Notebook.Macro_Library Yes|No} is equivalent to options of the notebook property Macro Library. To make the active notebook a macro library, use Yes.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Palette<.*Option*>}

{Notebook.Palette<.*Option*>}	Equivalent to Property| Active Notebook...
{Notebook.Palette *Color1,Color2,...,Color16*}	...|Palette
{NoteBook.Palette.Color_n "*RedValue,GreenValue,BlueValue*"}	...|Palette
{NoteBook.Palette.Color_n.Green *GreenValue*}	...|Palette|Edit Color
{NoteBook.Palette.Color_n.Blue *BlueValue*}	...|Palette|Edit Color
{NoteBook.Palette.Color_n.Red *RedValue*}	...|Palette|Edit Color

{Notebook.Palette.*Option*} is equivalent to the notebook property Palette, which lets you set the colors of the active notebook. The arguments of {Notebook.Palette} (*Color1* through *Color16*) each have three parts, separated by commas: *RedValue, GreenValue*, and *BlueValue*. Each part is a number from 0 to 255. You can also edit a part individually (see the second example).

Examples

`{Notebook.Palette.Color_3 "255,0,255"}` sets the third color on the notebook palette to violet (Red 255, Blue 255).

`{Notebook.Palette.Color_5.Blue "135"}` sets the amount of blue in the fifth color to 135.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Password *Password*}

{Notebook.Password *Password*} sets the password of the active notebook. The next save operation encrypts the file on disk.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Password_Level None|Low|Medium|High}

{Notebook.Password_Level} sets the password level of the active notebook. If you specify a password level of Low, Medium, or High, you must also specify a password using {Notebook.Password}.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Recalc_Settings "*Mode,Order,Iterations,<CompileFormulas?*(0|1)>,<*AuditErrors?*(0|1)>"}

{Notebook.Recalc_Settings} is equivalent to options of the notebook property Recalc Settings. This command equivalent sets the recalculation options of the active notebook. *Mode* options are "Automatic", "Background", and "Manual". *Order* can be "Column-wise", "Row-wise", or "Natural". *Iterations* specifies the number of times formulas are recalculated before calculation is considered complete (relevant only if *Order* is changed, or if you use circular references).

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.System Yes|No}

{Notebook.System Yes|No} makes the active notebook a system notebook.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{Notebook.Zoom_Factor 25-200}

{Notebook.Zoom_Factor 25-200} is equivalent to options of the notebook property Zoom Factor, which sets the zoom factor of the active notebook (from 25% to 200%). This setting is for display only and doesn't affect printed output.

See Also

Notebook.*Property*

Property|Active Notebook

Using Macros

Macro Command Descriptions

{OLE.*Option*}

Command equivalent	Equivalent to...
OLE.Change_Link *FileName*}	OLE Link settings|Change Link
{OLE.Change_To_Picture}	OLE Object settings|Change to Picture
{OLE.Edit}	OLE Link settings|Edit,
	OLE Object settings|Edit
{OLE.Play}	OLE Link settings|Play,
	OLE Object settings|Play
{OLE.Unlink}	OLE Link settings|Unlink
{OLE.Update}	OLE Link settings|Update Now

{OLE.*Option*} is equivalent to commands in the Object Inspector menus of OLE objects. Each command affects the selected OLE object, except {OLE.Update_All_Objects}, which refreshes all OLE objects in the active notebook. The type of OLE object determines what command equivalents affect it:

OLE type	Commands
Embedded	{OLE.Change_To_Picture}, {OLE.Edit}, {OLE.Play}
Linked	{OLE.Edit}, {OLE.Play}, {OLE.Update}, {OLE.Change_Link}, {OLE.Unlink}

Example

This macro selects an OLE object named Embedded1, lets the user edit the data (in the OLE server), and then converts the object into a picture (disabling the OLE link).

```
{SELECTFLOAT Embedded1}
{OLE.Edit}
{OLE.Change_To_Picture}
```

See Also

OLE Object

Using Macros

Macro Command Descriptions

{Optimizer.*Option*}

Command equivalent	Equivalent to Tools|Optimizer...
{Optimizer.Add *Constraint#, Cell, Operator, Constant*}	...|Constraints|Add
{Optimizer.Answer_Reporting *Cell*}	...|Options|Reporting|Answer Report Block
{Optimizer.Auto-scale 0|1}	...|Options|Automatic Scaling
{Optimizer.Change *Constraint#, Cell,* <=|>=|=|Integer, *Constant*}	...|Constraints|Change
{Optimizer.Delete *Constraint#*}	...|Constraints|Delete
{Optimizer.Derivatives Central|Forward}	...|Options|Derivatives
{Optimizer.Detail_Reporting *Cell*}	...|Options|Reporting|Detail Report Block
{Optimizer.Estimates Quadratic|Tangent}	...|Options|Estimates
{Optimizer.Linear 0|1}	...|Optimizer|Options|Assume Linear
{Optimizer.Load_Model}	...|Options|Load Model
{Optimizer.Max_Iters *Value*}	...|Options|Max Iterations
{Optimizer.Max_Time *Value*}	...|Options|Max Time
{Optimizer.Model_Cell *Cell*}	...|Options|Load Model, ...|Options|Save Model
{Optimizer.Precision *Value*}	...|Options|Precision
{Optimizer.Reset}	...|Reset
{Optimizer.Save_Model}	...|Options|Save Model
{Optimizer.Search Conjugate|Newton}	...|Options|Search
{Optimizer.Show_Iters 0|1}	...|Options|Show Iteration Results
{Optimizer.Solution_Cell *SolutionCell*}	...|Goal|Solution Cell
{Optimizer.Solution_Goal Max|Min|None|Target Value}	...|Goal
{Optimizer.Solve}	...|Solve
{Optimizer.Target_Value *Value*}	...|Target Value
{Optimizer.Tolerance *Value*}	...|Options|Tolerance
{Optimizer.Variable_Cells *Cell(s)*}	...|Variable Cells

{Optimizer.*Option*} is the command equivalent for Tools|Optimizer. It performs goal-seeking calculations and solves sets of linear and nonlinear equations and inequalities.

Constraint# refers to a constraint's order in the constraint list. *Constant* may be a value or a cell containing a value. The *Value* for Target_Value may also be a value or a cell. Use {Optimizer.Solve} after the other commands to calculate the solution.

To save an Optimizer model, use {Optimizer.Model_Cell *Cell*} {Optimizer.Save_Model}. To load a model, use {Optimizer.Model_Cell Cell}{Optimizer.Load_Model}

Example

The following macro sets up an Optimizer problem designed to maximize the formula in D6 by varying cells B8..B10. Seven constraints limit the solution. All options have been changed from their default settings. T2 and G13 are the upper-left cells of the report blocks.

```
{Optimizer.Solution_Cell A:D6}
{Optimizer.Solution_Goal Max}
{Optimizer.Variable_Cells A:B8..A:B10}
{Optimizer.Add 1,"A:D8..A:D8",<=,"1000"}
{Optimizer.Add 2,"A:B8..A:B8",>=,"100"}
{Optimizer.Add 3,"A:B9..A:B9",>=,"100"}
{Optimizer.Add 4,"A:B10..A:B10",>=,"100"}
{Optimizer.Add 5,"A:D8..A:D8",>=,"500"}
{Optimizer.Add 6,"A:D9..A:D9",<=,"900"}
{Optimizer.Add 7,"A:D10..A:D10",<=,"110000"}
{Optimizer.Max_Time 50} {Optimizer.Max_Iters 300}
{Optimizer.Precision 5E-05} {Optimizer.Linear 1}
{Optimizer.Show_Iters 1}
{Optimizer.Estimates Quadratic}
{Optimizer.Derivatives Central}
{Optimizer.Search Conjugate}
{Optimizer.Detail_Reporting A:T2..A:T2}
{Optimizer.Answer_Reporting A:G13..A:G13}
{Optimizer.Solve}
```

See Also

Tools|Optimizer

Using Macros

Macro Command Descriptions

{Order.*Option*}

Command equivalent	Equivalent to ...
{Order.Backward}	Dialog|Order|Send Backward, Draw|Send Backward
{Order.Forward}	Dialog|Order|Bring Forward, Draw|Bring Forward
{Order.ToBack}	Dialog|Order|Send to Back, Draw|Send to Back
{Order.ToFront}	Dialog|Order|Bring to Front, Draw|Bring to Front

{Order.*Option*} is equivalent to Draw|Order and Dialog|Order, which reorder overlapping objects in a graph or dialog window. Each command affects selected objects in the active window.

See Also

Dialog|Order
Draw Menu
Using Macros
Macro Command Descriptions
{FloatOrder.*Option*}

{Page.*Property*}

{Page.*Property*} is equivalent to Property|Active Page (the page Object Inspector). Each command affects the active page(s). The next table lists the possible settings for *Property*. To display a property description with syntax, choose the property in the following list:

Property	Property|Active Page option
Borders	Borders
Conditional_Color	Conditional Color
Default_Width	Default Width
Display_Zeros	Display Zeros
Grid_Lines	Grid Lines
Label_Alignment	Label Alignment
Line_Color	Line Color
Name	Name
Protection	Protection
Tab_Color	Tab Color

See Also

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Borders<.*Option*>}

{Page.Borders<.*Option*>}	Equivalent to Property| Active Page...
{Page.Borders "Row, Column"}	...|Borders
{Page.Borders.Column_Borders Yes|No}	...|Borders|Column Borders
{Page.Borders.Row_Borders Yes|No}	...|Borders|Row Borders

{Page.Borders.*Option*} is equivalent to the page property Borders, which lets you hide the row and column borders of the active page.

Example

`{Page.Border "No,Yes"}` hides the row border and leaves the column border.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Conditional_Color<.*Option*>}

{Page.Conditional_Color<.*Option*>}	Equivalent to Property| Active Page...
{Page.Conditional_Color "*Enable, SmallVal, GreatVal, BelowColor, NormalColor, AboveColor, ERRColor*"}	...|Conditional Color
{Page.Conditional_Color.Above_Normal_Color 0-15}	...|Conditional Color|Above Normal Color
{Page.Conditional_Color.Below_Normal_Color 0-15}	...|Conditional Color|Below Normal Color
{Page.Conditional_Color.Enable Yes|No}	...|Conditional Color|Enable
{Page.Conditional_Color.ERR_Color 0-15}	...|Conditional Color|ERR Color
{Page.Conditional_Color.Greatest_Normal_Value *Value*}	...|Conditional Color| Greatest Normal Value
{Page.Conditional_Color.Normal_Color 0-15}	...|Conditional Color|Normal Color
{Page.Conditional_Color.Smallest_Normal_Value *Value*}	...|Conditional Color| Smallest Normal Value

{Page.Conditional_Color.*Option*} is equivalent to the page property Conditional Color, which makes cells change text color (based on the value in the cell). Each color specified in these commands is a number from 0 to 15, corresponding to which color of the notebook palette to use (1 through 16).

Example

The following macro makes negative values red, values greater than 10,000 green, ERR cells cyan, and positive values less than 10,000 black (assuming the default notebook palette is used).

```
{Page.Conditional_Color"Yes,0,10000,4,3,5,7"}
```

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Default_Width *Width*}

{Page.Default_Width *Width*} is equivalent to the page property Default Width. It sets the default column width of the active page. *Width* is the new column width in twips (a twip is 1/1440th of an inch).

Example

`{Page.Default_Width "720"}` makes the default column width a half inch (720 twips).

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Display_Zeros Yes|No}

{Page.Display_Zeros Yes|No} is equivalent to the page property Display Zeros. It specifies whether formulas returning zero are displayed (Yes) in the active page.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Grid_Lines<.*Option*>}

{Page.Grid_Lines<.*Option*>}	Equivalent to Property|Active Page...
{Page.Grid_Lines "Horizontal,Vertical"}	...|Grid Lines
{Page.Grid_Lines.Horizontal Yes|No}	...|Grid Lines|Horizontal
{Page.Grid_Lines.Vertical Yes|No}	...|Grid Lines|Vertical

{Page.Grid_Lines.*Option*} is equivalent to the page property Grid Lines, which hides or reveals grid lines in the active notebook page.

Example

`{Page.Grid_Lines "No,Yes"}` hides the horizontal grid lines of the active page.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Label_Alignment Left | Right | Center}

{Page.Label_Alignment Left|Right|Center} is equivalent to the page property Label Alignment. It sets the default label alignment for the active page.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Line_Color 0-15}

{Page.Line_Color 0-15} is equivalent to the page property Line Color. It sets the color of line drawings on the active page. Each value from 0 to 15 specifies a different color on the notebook palette.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Name *NewName*}

{Page.Name *NewName*} is equivalent to the page property Name. It sets the name of the active page to *NewName*.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Protection Disable|Enable}

{Page.Protection Disable|Enable} is equivalent to the page property Protection. It enables or disables cell protection on the active page.

See Also

Page.*Property*

Property|Active Page

Using Macros

Macro Command Descriptions

{Page.Tab_Color "*Red, Green, Blue*"}

{Page.Tab_Color} changes the tab color of the active page; *Red*, *Green*, and *Blue* are integers from 0 to 255.

See Also

Page.*Property*

Property|Active Page

The Color Property

Using Macros

Macro Command Descriptions

{Parse.*Option*}

Command equivalent	Equivalent to Data|Parse...
{Parse.Create}	...|Create
{Parse.EditLine *NewEditLine*}	...|Edit
{Parse.Go}	...|OK
{Parse.Input *Block*}	...|Input
{Parse.Output *Block*}	...|Output
{Parse.Reset}	...|Reset

{Parse.*Option*} is the command equivalent for Data|Parse. It breaks long text strings into data fields according to a format line.

{Parse.Input} indicates the block to parse. {Parse.Output} indicates the block to hold parsed data. Use {Parse.Create} to build the format line. {Parse.EditLine} lets you specify a new format line, enclosed in quotes. {Parse.Create} and {Parse.EditLine} act on the active block.

Use {Parse.Go} after setting up input and output blocks and creating a format line. {Parse|Reset} clears previous settings.

See Also

Data|Parse

Using Macros

Macro Command Descriptions

{PasteFormat LinkType}

{PasteFormat} is equivalent to Edit|Paste Format, which lets you create OLE links to other applications and paste data in a specific format into a notebook. Use *LinkType* to specify the paste format.

Example

`{PasteFormat Bitmap}` pastes the data in the Clipboard as a bitmap into the active notebook.

See Also

Edit|Paste Format
{EditPaste}
{PasteSpecial}
{PasteLink}
Using Macros
Macro Command Descriptions

{PasteLink}

{PasteLink} is equivalent to Edit|Paste Link, which lets you set up a DDE link to another application.

See Also

Edit|Paste Link
{EditPaste}
{PasteSpecial}
{PasteFormat}
Using Macros
Macro Command Descriptions

{PasteSpecial Properties, Contents, Transpose, Blanks}

{PasteSpecial} is equivalent to Edit|Paste Special, which pastes certain aspects of data from the Clipboard.

Example

`{PasteSpecial "",Values,"",NoBlanks}` pastes formula results from the Clipboard, and skips any blank cells in the Clipboard.

See Also

Edit|Paste Special

{EditPaste}

{PasteLink}

{PasteFormat}

Using Macros

Macro Command Descriptions

{Preview}

{Preview} is equivalent to File|Print Preview, which lets you preview a printout onscreen.

See Also

File|Print Preview

{Print.*Option*}

Using Macros

Macro Command Descriptions

{Print.*Option*}

{Print.*Option*} is equivalent to the menu commands in the following list. To display specific command equivalents, choose one of the following.

Command equivalents for...

File|Named Settings

File|Page Setup

File|Print

File|Print|Options

The command equivalent {Print.PrintReset} resets print settings in all the dialog boxes displayed by these commands.

See Also
Using Macros

Macro Command Descriptions

File | Named Settings Command Equivalents

File|Named Settings command equivalent	Equivalent to File| Named Settings...
{Print.Create *NamedSetting*}	...|Create, ...|Update
{Print.Delete *NamedSetting*}	...|Delete
{Print.Use *NamedSetting*}	...|OK

These command equivalents are equivalent to File|Named Settings. *NamedSetting* is the named print setting to affect. To update an existing named setting, use {Print.Create}. {Print.Delete} removes a named setting from the active notebook. {Print.Use} sets the current print settings to those stored under the name.

See Also

File|Named Settings

Using Macros

Macro Command Descriptions

File | Page Setup Command Equivalents

File|Page Setup command equivalent	Equivalent to File|Page Setup...
{Print.Bottom_Margin *Value*}	...|Margins|Bottom
{Print.Center_Block Yes|No}	...|Options|Center Blocks
{Print.Footer *FooterString*}	...|Footer
{Print.Footer_Margin *Value*}	...|Margins|Footer
{Print.Header *HeaderString*}	...|Header
{Print.Header_Margin *Value*}	...|Margins|Header
{Print.Headers_Font "*Typeface, PointSize, Bold* (Yes|No), *Italic* (Yes|No), *Underline* (Yes|No), *Strikeout* (Yes|No)"}	...|Header Font
{Print.Left_Margin *Value*}	...|Margins|Left
{Print.Orientation Landscape|Portrait}	...|Print Orientation
{Print.Page_Breaks Yes|No}	...|Options|Break Pages
{Print.PageSetupReset}	...|Reset Defaults
{Print.Paper_Type *PaperSize*}	...|Paper Type
{Print.Print_To_Fit Yes|No}	...|Options|Print to Fit
{Print.Right_Margin *Value*}	...|Margins|Right
{Print.Scaling 1-1000}	...|Scaling
{Print.Top_Margin *Value*}	...|Margins|Top

These command equivalents are equivalent to File|Page Setup. When specifying a margin, the default measurement system is used (set in the Windows Control Panel). To use a specific measurement system, place in (for inches) or cm (for centimeters) after the new margin setting (see the example). The new setting is converted into the default measurement system.

Example

This macro sets the top and bottom margins to three centimeters, specifies landscape orientation, and sets the paper size to Legal.

```
{Print.Top_Margin "3 cm"}
{Print.Bottom_Margin "3 cm"}
{Print.Orientation Landscape}
{Print.Paper_Type "Legal 8 1/2 x 14 inch"}
```

See Also

File|Page Setup

Using Macros

Macro Command Descriptions

File | Print Command Equivalents

File|Print command equivalent	Equivalent to File|Print...
{Print.All_Pages Yes|No}	...|Print Pages|All Pages
{Print.Block *Block*}	...|Print Blocks
{Print.Copies *Value*}	...|Copies
{Print.DoPrint}	...|Print
{Print.DoPrintGraph}	...|Print (selected graph)
{Print.End_Page_Number *Value*}	...|Print Pages|To
{Print.Start_Page_Number *Value*}	...|Print Pages|From

These command equivalents are equivalent to File|Print (the commands available for File|Print|Options are discussed next). {Print.DoPrint} prints the active notebook (or active graph) using current print settings. {Print.DoPrintGraph} provides a quick way to print a graph. If a floating graph is selected, {Print.DoPrintGraph} prints the graph being shown; if a graph icon is selected, {Print.DoPrintGraph} prints the graph represented by that icon; if a graph window is active, {Print.DoPrintGraph} prints the graph shown.

Examples

This macro selects an icon on the Graphs page named Report3 and prints the graph it represents.

```
{GRAPHPAGEGOTO}
{SELECTOBJECT Report3}
{Print.DoPrintGraph}
```

This macro prints pages seven through twelve of a document. The print block is A3..C234.

```
{Print.Block A3..C234}
{Print.All_Pages No}
{Print.Start_Page_Number 7}
{Print.End_Page_Number 12}
{Print.DoPrint}
```

See Also

File|Print

Using Macros

Macro Command Descriptions

File | Print | Options Command Equivalents

File|Print|Options command equivalent	Equivalent to File|Print| Options...
{Print.Between_Block_Formatting "Lines"|"Page Advance"}	...|Print Between Blocks
{Print.Between_Page_Formatting "Lines"|"Page Advance"}	...|Print Between 3D Pages
{Print.Cell_Formulas Yes|No}	...|Print Options|Cell formulas
{Print.Left_Heading Block}	...|Left Heading
{Print.Lines_Between_Blocks *Value*}	...|Print Between Blocks|Lines
{Print.Lines_Between_Pages *Value*}	...|Print|Between 3D Pages|Lines
{Print.Print_Borders Yes|No}	...|Print Options|Row/Column Borders
{Print.Print_Gridlines Yes|No}	...|Print Options|Gridlines
{Print.PrintOptionsReset}	...|Reset Defaults
{Print.PrintReset}	Resets all print settings
{Print.Top_Heading *Block*}	...|Top Heading

These command equivalents are equivalent to the Options command (in the File|Print dialog box). {Print.Between_Page_Formatting} and {Print.Lines_Between_Pages} control the amount of space left between notebook pages (if the print block spans multiple pages).

{Print.Between_Block_Formatting} and {Print.Lines_Between_Blocks} control space between the subblocks that make up a noncontiguous print block.

Example

This macro specifies that three lines should be printed between each notebook page (if the print block spans multiple pages), and that row and column borders should print.

```
{Print.Between_Page_Formatting "Lines"}
{Print.Lines_Between_Pages 3}
{Print.Print_Borders Yes}
```

See Also

File|Print

Using Macros

Macro Command Descriptions

{PrinterSetup Printer, Port, PrintToFile?, Filename, ReplaceOption}

{PrinterSetup} is equivalent to File|Printer Setup, which specifies the destination for Quattro Pro print jobs. To create a binary file, set *PrintToFile* to 1, *FileName* to the name of the binary file, and use *ReplaceOption* to tell Quattro Pro what to do if the binary file already exists. The following table lists the possible settings for *ReplaceOption*.

Setting	Description
0	Cancels the operation. The file won't be overwritten.
1	Overwrites the file.
2	Makes a backup copy of the file before overwriting it.
3	Appends the new printing to the end of the existing file. This option is only available when printing plain text files.

Example

The following macro creates a binary file using current print settings. The binary file is named REPORT.PRN. If the binary file already exists, it's overwritten.

```
{PrinterSetup "Epson LQ-2500","LPT1:",1,"REPORT.PRN",1}
{Print.DoPrint}
{PrinterSetup "Epson LQ-2500","LPT1:",0,"",1}
```

See Also

File|Printer Setup

Using Macros

Macro Command Descriptions

{Query.*Option*}

Command equivalent	Equivalent to Data|Query...
{Query.Assign_Names}	...|Field Names
{Query.Criteria_Table *Block*}	...|Criteria Table
{Query.Database_Block *Block*}	...|Database Block
{Query.Delete}	...|Delete
{Query.Extract}	...|Extract
{Query.Locate}	...|Locate
{Query.Locate} enters FIND mode	
{Query.EndLocate} returns the user to the dialog box	
{Query.Output_Block *Block*}	...|Output Block
{Query.Reset}	...|Reset
{Query.Unique}	...|Extract Unique

{Query.*Option*} is equivalent to Data|Query, which lets you set up a Quattro Pro database and search for records in that database. {Query.Locate} enters FIND mode and stays under macro control until {PAUSEMACRO} is used or {Query.EndLocate}, which exits FIND mode.

Examples

The following macro sets up a database block and criteria table (A2..G37 and H1..H2), searches for records using the criteria table, and copies the first record found to A50.

```
{Query.Database_Block A2..G37}
{Query.Criteria_Table H1..H2}
{Query.Locate}
{BlockCopy []C(0)R(0)..C(6)R(0),A50}
{Query.EndLocate}
```

The following macro expands on the database set up in the previous example. It sets up an output block at J2..P2, and copies any records found by the previous search there.

```
{Query.Output_Block J2..P2}
{Query.Extract}
```

See Also

Data|Query

{PAUSEMACRO}

Using Macros

Macro Command Descriptions

{Regression.*Option*}

Command equivalent	Equivalent to Tools|Advanced Math|...
{Regression.Dependent Block}	...Regression|Dependent
{Regression.Go}	...Regression|OK
{Regression.Independent *Block*}	...Regression|Independent
{Regression.Output *Block*}	...Regression|Output
{Regression.Reset}	...Regression|Reset
{Regression.Y_Intercept Compute|Zero}	...Regression|Y Intercept

{Regression.*Option*} is the command equivalent for Tools|Advanced Math|Regression. It performs a regression analysis to show the relationship between a set of independent variables and a dependent variable.

{Regression.Dependent} indicates the dependent-variable block. {Regression.Independent} defines the independent variables. In {Regression.Independent}, *Block* can be noncontiguous with one variable to a column. The dependent and independent blocks must all have the same number of rows.

{Regression.Output} indicates where to store the table of regression results. {Regression.Y_Intercept} specifies whether to compute the Y-intercept, or set it to zero. You can use {Regression.Reset} to clear all settings. Use {Regression.Go} after the other command equivalents to perform the regression analysis. If data changes within the independent or dependent data blocks, use {Regression.Go} again to calculate a new regression table.

Example

The following macro sets these data blocks: Independent, B2..D16; Dependent, F2..F16. The last statement performs the regression analysis and stores the results in the block with upper-left cell H2.

```
{Regression.Independent_Block A:B2..A:D16}
{Regression.Dependent_Block A:F2..A:F16}
{Regression.Output_Block A:H2}
{Regression.Go}
```

See Also

Tools|Advanced Math|Regression

Using Macros

Macro Command Descriptions

{ResizeToSame}

{ResizeToSame} is equivalent to Dialog|Align|Resize to Same, which lets you resize selected objects in the dialog window to the same size as the first object selected.

See Also

Dialog|Align

Using Macros

Macro Command Descriptions

{RestrictInput.*Option*}

Command equivalent	Equivalent to...
{RestrictInput.Enter *Block*}	Data|Restrict Input (enters Input mode)
{RestrictInput.Exit}	Any operation that ends INPUT mode

{RestrictInput.*Option*} is equivalent to Data|Restrict Input, which confines selector movement to a specific block of unprotected cells.

{RestrictInput.Enter} enters INPUT mode and stays under macro control until {PAUSEMACRO} is used or {RestrictInput.Exit}, which exits INPUT mode.

See Also

Data|Restrict Input

{PAUSEMACRO}

Using Macros

Macro Command Descriptions

{Scenario.*Option*}

Command equivalent	Equivalent to Tools|Scenario Manager...
{Scenario.AddCells <*Block*>}	Add Scenario Cells button
{Scenario.Capture *ScenarioName*}	Capture Scenario button
{Scenario.CaptureArea *Area*,*Block*}	Group button|Capture Area
{Scenario.Close}	SpeedBar Control Menu|Remove
{Scenario.DeleteGroup *GroupName*}	Group button|Delete button
{Scenario.Find}	Find Scenario Cells button
{Scenario.Highlight *Highlight?*(0|1)}	Toggle Highlights button
{Scenario.NewGroup *GroupName*}	Group button|New button
{Scenario.Open}	SpeedBar Control Menu|Append or Insert
{Scenario.Remove *ScenarioName*}	Delete Scenario button
{Scenario.RemoveCells <*Block*>}	Remove Scenario Cells button
{Scenario.RenameGroup *OldGroupName*,*NewGroupName*}	Group button|Rename button
{Scenario.Report *AllGroups*(0|1), *LeftLabels*(0|1), *TopLabels*(0|1), <*Block*>}	Report button
{Scenario.Show *ScenarioName*}	Scenario Name list
{Scenario.Update_On_Block *Update?* (0|1)}	Group button|Options button
{Scenario.UseGroup *GroupName*}	Group Name list

{Scenario.*Option*} is the command equivalent for Tools|Scenario Manager. It lets you change values in a model, saving the conditions and results for different scenarios. {Scenario.Open} must be used prior to using other {Scenario.Option} commands; use {Scenario.Close} when you're finished using the Scenario Manager. For {Scenario.AddCells} and {Scenario.RemoveCells}, *Block* defaults to the currently selected block. For {Scenario.Report}, *Block* defaults to the first empty page in the notebook.

Example

The following macro captures the base scenario and two additional scenarios for a car loan.

```
{Scenario.Open}
{Scenario.Capture Base Scenario}
{Scenario.Update}
{SelectBlock A:F4}
{PutCell ".096"}
{SelectBlock A:C5}
{PutCell "60"}
{Scenario.Find}
{Scenario.Highlight 1}
{Scenario.Capture APR96-60}
{Scenario.Update}
{SelectBlock A:F4}
{PutCell ".085"}
```

```
{SelectBlock A:C5}
{PutCell "48"}
{Scenario.Find}
{Scenario.Highlight 1}
{Scenario.Capture APR85-48}
{Scenario.Update}
{Scenario.Report 0,1,1}
{FileSaveAs "C:\QPW\CARS.WB1"}
{Scenario.Close}
```

See Also

Using the Scenario Manager

Using Macros

Macro Command Descriptions

{Search.*Option*}

Command equivalent	Equivalent to Edit| Search and Replace...
{Search.Block Block}	...|Blocks
{Search.Case Any|Exact}	...|Case Sensitive
{Search.Direction Column|Row}	...|Columns First
{Search.Find String}	...|Find
{Search.Look_In Condition| Formula|Value}	...|Look In
{Search.Match Part|Whole}	...|Match Whole
{Search.Next}	...|Next
{Search.Previous}	...|Previous
{Search.Replace}	...|Replace button
{Search.ReplaceAll}	...|Replace All
{Search.ReplaceBy String}	...|Replace edit field
{Search.Reset}	...|Reset

{Search.*Option*} is equivalent to Edit|Search and Replace, which searches for strings in the active page. Use {Search.ReplaceBy} to specify the replacement string; {Search.Replace} replaces the string.

Example

The following macro searches the active page for 1993 in formulas and replaces it with 1994.

```
{Search.Reset}
{Search.Block ""}
{Search.Look_In Formula}
{Search.Match Part}
{Search.Find "1993"}
{Search.ReplaceBy "1994"}
{Search.ReplaceAll}
```

See Also

Edit|Search and Replace

Using Macros

Macro Command Descriptions

{Series.*Option*}

Command equivalent	Equivalent to Graph|Series...
{Series.Data_Range *SeriesNumber* | "XAxisLabelSeries" | "LegendSeries", *Block* <,*CreateIfNotExist?* (0|1)>}	...|Number or X-Axis or Legend
{Series.Delete *SeriesNumber* <,*AndAllSeriesFollowing?*>}	...|Delete
{Series.Go}	...|OK
{Series.Insert *SeriesNumber, Block*}	...|Add
{Series.Label_Range *SeriesNumber*, *Block* <,*CreateIfNotExist?* (0|1)>}	[series Object Inspector]|Label Series
{Series.Legend *SeriesNumber*, *LegendText*}	[series Object Inspector]| Legend
{Series.Reverse_Series 1|0}	...|Reverse Series
{Series.Swap_Row_Col 1|0}	...|Row/Column Swap

{Series.*Option*} is equivalent to Graph|Series and options in a series Object Inspector, which create or delete graph series.

When you manipulate a series using command equivalents, the changes aren't made until the command {Series.Go} is used. In all the commands, *SeriesNumber* is the number of the series to affect (1 for the first series, 2 for the second, and so on).

{Series.Data_Range} changes the values of an existing series. *Block* is the new block the series should take values from. If you're not sure whether the series exists, set *CreateIfNotExist?* to 1. Then the series will be created if it doesn't already exist. You can also use {Series.Data_Range} to set the x-axis series (use "XAxisLabelSeries") or set the legend series (use "LegendSeries").

{Series.Delete} removes an existing series. Set *AndAllSeriesFollowing?* to 1 if you also want to remove all series following *SeriesNumber*.

{Series.Insert} creates a new series. The series is inserted at the position specified by *SeriesNumber*. *Block* is the block containing the new series' data.

{Series.Label_Range} sets up the labels for each value in a series. *Block* is the block containing the labels. If you're not sure whether the series exists, set *CreateIfNotExist?* to 1. Then the series will be created if it doesn't already exist.

{Series.Legend} sets the legend text for a series (*LegendTex*t is the new text).

Example

The following macro creates a graph named Profit99 with two series. The series values are in A:A1..A27 and A:C1..C27. The series labels are in A:B1..B27 and A:D1..D27. The x-axis is stored in A:E1..E27.

```
{GraphNew Profit99}
{GraphEdit Profit99}
{Series.Data_Range 1,A:A1..A27,1}
{Series.Data_Range 2,A:C1..C27,1}
{Series.Label_Range 1,A:B1..B27}
```

```
{Series.Label_Range 2,A:D1..D27}
{Series.Data_Range "XAxisLabelSeries",A:E1..E27}
{Series.Go}
```

The following macro inserts a new series between the two series in the last example.

```
{GraphEdit Profit99}
{Series.Insert 2,A:G1..G27}
{Series.Go}
```

See Also

Graph|Series

Using Macros

Macro Command Descriptions

{Slide.*Option*}

Command equivalent	Action
{Slide.Effect *Effect*}	Specifies the transition effect to use when displaying the next slide in a slide show; no equivalent command
{Slide.Goto *SlideName*}	Takes the active slide show directly to the slide SlideName; no equivalent command
{Slide.Next}	Advances the active slide show to the next slide; no equivalent command
{Slide.Previous}	Returns the active slide show to the previous slide; no equivalent command
{Slide.Run *SlideShowName*}	Graph|Slide Show
{Slide.Speed 0-15}	Specifies the transition speed to use when displaying the next slide in a slide show; no equivalent command
{Slide.Time *Time*}	Specifies the time in seconds to display the next slide in a slide show; no equivalent command

{Slide.*Option*} lets you build, edit, and present graphic slide show sequences. *Effect*, *Speed*, and *Time* are the same options offered in the Light Table dialog box. {Slide.Effect}, {Slide.Speed}, {Slide.Time}, {Slide.Goto}, {Slide.Next}, and {Slide.Previous} can be in the spreadsheet macro which started the slide show, in a spreadsheet macro run from a graph button, or attached directly to a SpeedButton or custom dialog box button.

See Also

Graph|Slide Show

Using Macros

Macro Command Descriptions

{SolveFor.*Option*}

Command equivalent	Equivalent to Tools|Solve For...
{SolveFor.Accuracy *Value*}	...|Accuracy
{SolveFor.Variable_Cell *Cell*}	...|Variable Cell
{SolveFor.Formula_Cell *Cell*}	...|Formula Cell
{SolveFor.Go}	...|OK
{SolveFor.Max_Iters *Value*}	...|Max Iterations
{SolveFor.Reset}	Clears all Solve For settings; no equivalent menu command
{SolveFor.Target_Value *Value*}	...|Target Value

{SolveFor.*Option*} is the command equivalent for Tools|Solve For. It solves goal-seeking problems with one variable.

{SolveFor.Formula_Cell} indicates the location of the formula to evaluate. {SolveFor.Target_Value} is the goal to reach, either a number or a cell containing a number. {SolveFor.Variable_Cell} indicates the formula variable (a referenced cell) that can change to reach the target value.

{SolveFor.Max_Iters} and {SolveFor.Accuracy} control how many calculation passes to make and how closely the solution must match the target value. Use {SolveFor.Go} after the other commands. {SolveFor.Reset} clears previous settings.

See Also

Tools|Solve For

Using Macros

Macro Command Descriptions

{Sort.*Option*}

Command equivalent	Equivalent to Data|Sort...
{Sort.Block *Block*}	...|Block
{Sort.Data "Labels First"|"Numbers First"}	...|Data
{Sort.Go}	...|OK
{Sort.Key_*1-5 Block*}	...|Column|1st - 5th
{Sort.Labels "Character Code"|"Dictionary"}	...|Labels
{Sort.Order_*1-5* Ascending|Descending}	...|Ascending|1st - 5th
{Sort.Reset}	...|Reset

{Sort.*Option*} is equivalent to Data|Sort, which sorts the entries in a block. To perform the sort, use {Sort.Go} after the other sort command equivalents.

Example

The following macro sorts the block A3..C35 using two sort keys (columns A and C). The sort is in ascending order, and values in a column are placed in a group before labels in the column. The labels are sorted in dictionary order.

```
{Sort.Reset}
{Sort.Block A3..C35}
{Sort.Key_1 A25}
{Sort.Order_1 Ascending}
{Sort.Key_2 C23}
{Sort.Order_2 Ascending}
{Sort.Data "Numbers First"}
{Sort.Labels Dictionary}
{Sort.Go}
```

See Also

Data|Sort

Using Macros

Macro Command Descriptions

{SpeedBar.*Option*}

Command equivalent	Equivalent to ***SpeedBar Control Menu*|...**
{SpeedBar.Append *BarName*,<*Position*>}	...|Append
{SpeedBar.Insert *BarName*,<*Position*>}	...|Insert
{SpeedBar.Remove *BarName*}	...|Remove
{SpeedBar.Replace *BarName*,*Position*}	...|Replace

{SpeedBar.*Option*} is the command equivalent for the SpeedBar Control Menu. To display the SpeedBar Control Menu, right-click any blank area of a SpeedBar. *Position* can be a bar name or number. For {SpeedBar.Insert}, the default *Position* is 1. For {SpeedBar.Append}, the default *Position* is below all other SpeedBars.

See Also

Using Macros

Macro Command Descriptions

{SpellCheck.*Option*}

These command equivalents are for features in the Workgroup edition of Quattro Pro.

Command equivalent	Equivalent to Tools|Spell Check...
{SpellCheck <*SourceBlock*>}	...|Start button
{SpellCheck.Dictionary *DictName*}	...|Options button|Personal Dictionary
{SpellCheck.IgnoreAcronym Yes|No}	...|Options button|Spelling options|Ignore acronym errors
{SpellCheck.IgnoreCapErr Yes|No}	...|Options button|Spelling options|Ignore capitalization errors
{SpellCheck.IgnoreNumWord Yes|No}	...|Options button|Spelling options|Ignore words with numbers
{SpellCheck.IgnoreRepeatWord Yes|No}	...|Options button|Spelling options|Ignore repeated words
{SpellCheck.IgnoreUppercase Yes|No}	...|Options button|Spelling options|Ignore UPPERCASE words
{SpellCheck.Language *Name*}	...|Options button|Language

{SpellCheck.*Option*} is the command equivalent for Tools|Spell Check in the Workgroup edition of Quattro Pro. It checks the spelling of text in *SourceBlock* (or the current selection if *SourceBlock* isn't supplied). *Option* corresponds to the options available when you click the Options button in the Spell Checker SpeedBar.

See Also

Using the Spell Checker

Using Macros

Macro Command Descriptions

{TableQuery.*Option*}

Command equivalent	Equivalent to Data|Table Query...
{TableQuery.Destination *Block*}	...|Destination
{TableQuery.FileQuery Yes|No}	...|Query in File
{TableQuery.Go}	...|OK
{TableQuery.QueryBlock *Block*}	...|QBE Block
{TableQuery.QueryFile *Filename*}	...|QBE File

{TableQuery.*Option*} is equivalent to Data|Table Query, which lets you search external databases for records. The query isn't performed until {TableQuery.Go} is used.

Examples

The following macro searches the external table TASKLIST.DB using the query file TASKLIST.QBE. The results of the search are stored in A:A2.

```
{TableQuery.FileQuery Yes}
{TableQuery.QueryFile TASKLIST.QBE}
{TableQuery.Destination A:A2}
{TableQuery.Go}
```

The next macro searches the same database, but uses the query defined in the named block task_query.

```
{TableQuery.FileQuery No}
{TableQuery.QueryBlock task_query}
{TableQuery.Destination A:A2}
{TableQuery.Go}
```

See Also

Data|Table Query

{Query.*Option*}

Using Macros

Macro Command Descriptions

{TableView}

{TableView} is equivalent to Data|Database Desktop, which launches the Database Desktop.

See Also

Data|Table Query

Using Macros

Macro Command Descriptions

{UngroupObjects}

{UngroupObjects} is the command equivalent for Draw|Ungroup. It separates the selected group of graph annotation objects so each can be moved or modified without affecting the others.

See Also

Draw|Ungroup

Using Macros

Macro Command Descriptions

{WhatIf.*Option*}

Command equivalent	Equivalent to Data|What-If...
{WhatIf.Block Block}	...|Data Table
{WhatIf.Input_Cell_1 *Cell*}	...|Input Cell (Column Input Cell)
{WhatIf.Input_Cell_2 *Cell*}	...|Row Input Cell
{WhatIf.One_Way}	...|One Free Variable, ...|Generate
{WhatIf.Reset}	...|Reset
{WhatIf.Two_Way}	...|Two Free Variables, ...| Generate

{WhatIf.*Option*} is the command equivalent for Data|What-If. It builds one- or two-variable "what-if" tables that display a range of results for different conditions.

If you're creating a one-variable table, use these command equivalents: {WhatIf.Input_Cell_1}, {WhatIf.Block}, {WhatIf.One_Way}. For two-variable tables, use {WhatIf.Input_Cell_2} after indicating the first input cell; use {WhatIf.Two_Way} instead of {WhatIf.One_Way}.

Example

The following macro defines A4..H18 as the "what-if" block, B1 as Input Cell 1, B2 as Input Cell 2, and builds a two-variable table.

```
{Whatif.Block A:A4..A:H18}
{Whatif.Input_cell_1 A:B1}
{Whatif.Input_cell_2 A:B2}
{Whatif.Two_Way}
```

See Also

Data|What-If

Using Macros

Macro Command Descriptions

{WindowArrIcon}

{WindowArrIcon} is equivalent to Window|Arrange Icons, which lines up minimized windows on the Quattro Pro desktop or icons on the Graphs page.

See Also

Window|Arrange Icons

Using Macros

Macro Command Descriptions

{WindowCascade}

{WindowCascade} is equivalent to Window|Cascade, which rearranges all open windows on the Quattro Pro desktop.

See Also

Window|Cascade

Using Macros

Macro Command Descriptions

{WindowClose}

{WindowClose} is equivalent to Close in a Control menu, which closes the active window (if the active window isn't saved, a prompt appears to confirm the operation).

See Also

Control Menu|Close

{FileClose}.

Using Macros

Macro Command Descriptions

{WindowHide}

{WindowHide} is equivalent to Window|Hide, which conceals the active notebook window.

See Also

Control Menu|Close

Using Macros

Macro Command Descriptions

{WindowMaximize}

{WindowMaximize} is equivalent to Maximize in a Control menu, which enlarges the active window so it fills the screen.

See Also

Control Menu|Maximize

Using Macros

Macro Command Descriptions

{WindowMinimize}

{WindowMinimize} is equivalent to Minimize in a Control menu, which shrinks the active window to an icon on the Quattro Pro desktop.

See Also

Control Menu|Minimize

Using Macros

Macro Command Descriptions

{WindowMove UpperLeftX, UpperLeftY}

{WindowMove} is equivalent to Move in a Control menu, which lets you move the active window. *UpperLeftX* and *UpperLeftY* are the new coordinates of the upper-left corner of the window.

See Also

Control Menu|Move

Using Macros

Macro Command Descriptions

{WindowNewView}

{WindowNewView} is the command equivalent for Window|New View. It displays a duplicate copy of the active notebook in a new window.

See Also

Window|New View

Using Macros

Macro Command Descriptions

{WindowNext}

{WindowNext} is equivalent to choosing Next in a Control menu or pressing Ctrl+F6. It makes the next window active.

See Also

Window|New View

Using Macros

Macro Command Descriptions

{WindowPanes Horizontal | Vertical | Clear, Synch?(0|1), Dim1, Dim2}

{WindowPanes} is the command equivalent for Window|Panes. It splits a notebook window into two horizontal or vertical panes; use Clear to restore a single pane.

Dim1 and *Dim2* indicate the ratio relationship between the panes.

Example

`{WindowPanes Vertical,0,2,1}` splits the notebook window into two vertical panes, not synchronized. The first pane is twice as wide as the second.

See Also

Window|Panes

Using Macros

Macro Command Descriptions

{WindowRestore}

{WindowRestore} is equivalent to Restore on the Control Menu. It restores minimized windows to their original size.

See Also

Control Menu|Restore

Using Macros

Macro Command Descriptions

{WindowShow Name}

{WindowShow} is the command equivalent for Window|Show. It shows hidden window *Name* and makes it active.

See Also

Window|Show

Using Macros

Macro Command Descriptions

{WindowSize X, Y}

{WindowSize} is equivalent to Size in the Control menu. It sizes the active window to the specified width and height.

See Also

Control Menu|Size

Using Macros

Macro Command Descriptions

{WindowTile}

{WindowTile} is the command equivalent for Window|Tile. It displays all open windows without overlapping them.

See Also

Window|Tile

Using Macros

Macro Command Descriptions

{WindowTitles Horizontal | Vertical | Both | Clear}

{WindowTitles Horizontal|Vertical|Both|Clear} is the command equivalent for Window|Locked Titles. It locks specific rows and/or columns of a spreadsheet page onscreen as titles. When you scroll, the titles remain fixed onscreen while the rows below (or columns to the right) scroll as usual. "Horizontal" locks rows above the active cell, "Vertical" locks columns to the left of the active cell, and "Both" locks both rows and columns. Use "Clear" to unlock the titles.

See Also

Window|Locked Titles

Using Macros

Macro Command Descriptions

{WorkSpace.*Option*}

Command equivalent	Equivalent to File|Workspace...
{Workspace.Restore *Filename*}	...|Restore
{Workspace.Save *Filename*}	...|Save

{Workspace.Save} saves all open notebooks as a group with the specified *Filename* (Quattro Pro's default file extension for workspaces is .WSB). {Workspace.Restore} opens the specified file.

See Also

File|Workspace

Using Macros

Macro Command Descriptions

Keyboard Macro Commands

When you record a macro, Quattro Pro translates special keystrokes into keyboard commands. When you type in a macro, use these commands in place of keystrokes. There are four categories:

Movement Keys	Home, End, and arrows
Function Keys	F1 through F12
Status Keys	Caps Lock, Scroll Lock, and other indicator keys
Other Keys	Break, Esc, Del, plus other keys and key combinations.

Keyboard commands emulate the keys of the keyboard. You can repeat many of these macro commands by specifying a repeat number as an argument. For example, to move the selector down five cells, use `{DOWN 5}`.

You can combine key commands for a variety of purposes. For example, the following macro replaces a formula in the active cell with its result and then moves down one cell:

```
{EDIT}{CALC}~{DOWN}
```

The following macro deletes the last three characters of whatever is in the active cell:

```
{EDIT}{END}{SHIFT+LEFT 3}{DEL}{CR}
```

You can also use keystroke commands to choose menu commands. Keystrokes step through the menu commands one key at a time. For example, the keystroke macro for copying A1 to A2..A46 is

```
{ALT+B}CA1~A2..A46~
```

See Also
Using Macros

Movement Key Macro Commands

Use these macro commands in place of arrows and other movement keys:

{HOME}	Home
{END}	End
{LEFT <*Number*>} or {L <*Number*>}	Left arrow
{RIGHT <*Number*>} or {R <*Number*>}	Right arrow
{UP <*Number*>} or {U <*Number*>}	Up arrow
{DOWN <*Number*>} or {D <*Number*>}	Down arrow
{PGUP <*Number*>}	PgUp
{PGDN <*Number*>}	PgDn
{BIGLEFT <*Number*>} or {BACKTAB <*Number*>}	Ctrl+Left arrow or Shift+Tab
{BIGRIGHT <*Number*>} or {TAB <*Number*>}	Ctrl+Right arrow or Tab

See Also

Using Macros
Function Key Macro Commands
Status Key Macro Commands
Macro Commands for Other Keys

Function Key Macro Commands

These macro commands are equivalent to Quattro Pro function keys and key combinations:

{ABS}	F4
{CALC}	F9
{EDIT}	F2
{FUNCTIONS}	Alt+F3
{GRAPH}	F11
{GRAPHPAGEGOTO}	Shift+F5
{HELP}	F1
{INSPECT}	F12
{MACROS}	Shift+F3
{MARK}	Shift+F7
{MENU}	F10
{NAME}	F3
{NEXTPANE}	F6
{NEXTTOPWIN}	Ctrl+F6
{NEXTWIN}	Shift+F6
{QGOTO}	F5
{QUERY}	F7
{STEPOFF}	Shift+F2
{STEPON}	Shift+F2
{TABLE}	F8

See Also

Using Macros
Movement Key Macro Commands
Status Key Macro Commands
Macro Commands for Other Keys

Status Key Macro Commands

These macro commands are equivalent to the status indicator keys:

{CAPOFF}	Caps Lock off
{CAPON}	Caps Lock on
{INS} or {INSERT}	Toggles Ins on or off
{INSOFF}	Ins off
{INSON}	Ins on
{NUMOFF}	Num Lock off
{NUMON}	Num Lock on
{SCROLLOFF}	Scroll Lock off
{SCROLLON}	Scroll Lock on

See Also

Using Macros
Movement Key Macro Commands
Function Key Macro Commands
Macro Commands for Other Keys

Macro Commands for Other Keys

These macro commands are used for building key combinations and for entering or editing data:

{ALT+Key <Number>}	The keystroke Key while Alt is held down
{BACKSPACE <Number>} or {BS <Number>}	Backspace
{BREAK}	Returns Quattro Pro to Ready mode
{CLEAR}	Ctrl+Backspace (Clears any existing entry from a prompt line or from the input line in Edit mode)
{CR} or ~	Enter
{CTRL+Key <Number>}	The keystroke Key while Ctrl is held down
{DATE}	Ctrl+Shift+D
{DEL} or {DELETE}	Del
{ESC} or {ESCAPE}	Esc
{SHIFT+Key <Number>}	The keystroke Key while Shift is held down

See Also

Using Macros

Movement Key Macro Commands

Function Key Macro Commands

Status Key Macro Commands

Screen Macro Commands

These macro commands affect the display:

{BEEP <Number>}	Sounds the computer's bell
{INDICATE String}	Sets the mode indicator to display String
{PANELOFF}	Suppresses menus and prompts
{PANELON}	Restores menus and prompts disabled by {PANELOFF}
{WINDOWSOFF}	Prevents screen updating
{WINDOWSON}	Restores screen updating disabled by {WINDOWSOFF}
{ZOOM}	Maximizes or restores the active window

Screen commands can control the status line, suppress screen redraws, or sound the computer's bell. For example, `{INDICATE "WAIT"}` changes the status line indicator to WAIT; `{INDICATE}` resets the status line to its default.

See Also
Using Macros

Cell Macro Commands

These macro commands affect the data stored in specific cells. They can enter and read data, recalculate specific blocks, or convert values into formatted labels:

{BLANK Location}	Erases a cell or block
{CONTENTS Dest,Source, <Width#>,<Format#>}	Copies the contents of one cell to another as a formatted label
{GETDIRECTORYCONTENTS Block,<Path>}	Enters into Block an alphabetized list of files in the directory specified by Path; if Path is not included, lists the current directory
{GETWINDOWLIST Block}	Lists open windows in Block
{LET Location,Value<:Type>}	Places a number or string in the given location; Type lets you store Value as a number or string
{PUT Location,Column#, Row#,Value<:Type>}	Places a number or string in a given location offset by the specified number of columns and rows; Type lets you store Value as a number or string
{PUTBLOCK Value,<Block>}	Stores Value in each cell of Block
{PUTCELL Data}	Stores Data in the active cell
{RECALC Location, <Condition>,<Iteration#>}	Recalculates formulas in Location in rowwise order for a specified number of times, or until Condition is reached
{RECALCCOL Location, <Condition>,<Iteration#>}	Recalculates formulas in Location in columnwise order for a specified number of times, or until Condition is reached
{SPEEDFILL}	Fills the selected block with sequential data
{SPEEDFORMAT Options}	Applies a standard format to the selected block
{SPEEDSUM Block}	Sums filled cells in Block, writes results in adjacent empty cells (right/bottom)

See Also
Using Macros

Interactive Macro Commands

Interactive commands let you create macros that accept users' input. These commands can display dialog boxes and pop-up menus, pause the macro and request input, and prevent users from breaking out of the macro during a crucial period:

{?}	Pauses the macro and accepts input from the keyboard until you press Enter
{ACTIVATE WindowName}	Makes the specified window active
{BREAKOFF}	Disables the Ctrl+Break key used to cancel a macro
{BREAKON}	Restores the Ctrl+Break key disabled by {BREAKOFF}
{CHOOSE}	Displays a pick list of open windows
{DODIALOG DialogName, OKExit?,<Arguments>, <MacUse?>}	Displays the dialog box DialogName for use
{GET Location}	Pauses the macro, accepts one keystroke and stores it in Location
{GETLABEL Prompt,Location}	Pauses the macro, displays Prompt, and stores the subsequent keystrokes as a label in Location
{GETNUMBER Prompt, Location}	Pauses the macro, displays Prompt, and stores the subsequent keystrokes as a numeric value in Location
{GOTO}	Moves the selector to a specified location
{GRAPHCHAR Location}	Stores the character a user presses to leave a graph or message in Location
{IFKEY String}	Returns true if String is the macro name for any key macro command
{LOOK Location}	Places the first keystroke typed in Location
{MENUBRANCH Location}	Passes macro control to a custom menu at Location
{MENUCALL Location}	Pauses the macro, and runs a customized menu as a subroutine
{MESSAGE Block,Left,Top, Time}	Displays contents of Block in a pop-up window for Time seconds
{PAUSEMACRO}	Pauses the macro for dialog box entries, then resumes the macro when the user closes the dialog box
{STEPOFF}	Exits Debug mode, executing the macro at normal pace
{STEPON}	Enters Debug mode, in which Quattro Pro runs the macro step by step, advancing when you press the spacebar
{UNDO}	"Takes back" the last command; cancels its effects
{WAIT DateTimeNumber}	Pauses the macro until the time specified in DateTimeNumber arrives
{WINDOW}	Makes the next pane active
{WINDOW<Number>}	Makes the specified window active

See Also
Using Macros

Program Flow Macro Commands

Program flow commands let you branch and loop in a macro. Use these commands to create macros that behave differently based on logical decisions, to break down complicated macros, or to consolidate repetitious macros into subroutines:

{BRANCH Location}	Passes execution control to another macro at Location
{DEFINE Location1<:Type1>,...}	Defines the type of arguments passed to a subroutine and tells where to store them
{DISPATCH Location}	Branches indirectly to an alternate macro branch
{FOR CounterLoc,Start#, Stop#,Step#,StartLoc}	Runs a subroutine the specified number of times
{FORBREAK}	Terminates execution of a {FOR} command
{IF Condition}	Checks to see if a condition is TRUE or FALSE before continuing execution
{ONERROR BranchLocation, <MessageLocation>, <ErrorLocation>}	Continues execution at a specified location after Quattro Pro detects an error
{QUIT}	Terminates macro execution and returns keyboard control
{RESTART}	Changes the current subroutine to the starting or main routine
{RETURN}	Terminates the current subroutine and returns control to main routine
{*Subroutine* <ArgumentList>}	Calls the specified subroutine, and passes any given arguments

See Also

Using Macro Subroutines

Using Macros

File Macro Commands

File macro commands work with data in files other than the active notebook. These commands can open files and read their data character by character, or create custom data files. If a file command succeeds, the macro command in the next cell is run, ignoring any other commands in the same cell as the file command. Commands in the same cell as the file command run only if the file command fails. Use this feature to trap errors such as the disk being full or reading data past the end of the file:

{ANSIREAD #Bytes,Location}	Reads the specified number of bytes and stores them in Location, without any character mapping
{ANSIREADLN Location}	Reads a line of characters and stores it in Location, without any character mapping
{ANSIWRITE String,<String2,...>}	Writes a string of characters to the current file, without any character mapping
{ANSIWRITELN String,<String2,...>}	Writes a string of characters to the current file and ends it with a carriage return/linefeed, without any character mapping
{CLOSE}	Closes a file opened by {OPEN}
{FILESIZE Location}	Calculates the number of bytes in the current file, and stores the value in Location
{GETPOS Location}	Places the position of the file pointer in Location
{OPEN Filename,AccessMode}	Opens a file for reading, writing, modifying, or appending
{READ #Bytes,Location}	Reads the specified number of bytes and stores them in Location
{READLN Location}	Reads a line of characters and stores it in Location
{SETPOS FilePosition}	Sets the file pointer to the value of FilePosition
{WRITE String,<String2,...>}	Writes a string of characters to the current file
{WRITELN String,<String2,...>}	Writes a string of characters to the current file and ends it with a carriage return/linefeed

See Also
Using Macros

DDE Macro Commands

DDE macro commands communicate with other Windows applications:

{EXEC AppName, WindowMode, <ResultLoc>}	Runs the application AppName
{EXECUTE DDEChannel, Macro,<ResultLoc>}	Runs the macro Macro in a server application
{INITIATE AppName, Topic, ChannelLoc}	Initiates a conversation with a DDE application
{POKE DDEChannel, Destination, DataToSend}	Sends data to a DDE application
{REQUEST DDEChannel, DataToReceive,DestBlock}	Requests data from a DDE application
{TERMINATE DDEChannel}	Ends a DDE conversation

For more about these commands and how they interact, see About DDE Macro Commands.

See Also

Using Macros

UI Building Macro Commands

UI building commands can add or remove menu commands from the active menu bar:

{ADDMENU MenuPath,MenuBlock}	Adds the menu MenuBlock to the menu system at MenuPath
{ADDMENUITEM MenuPath, Name, <*Options*>}	Adds the menu command Name to the menu system at MenuPath
{DELETEMENU MenuPath}	Deletes the menu MenuPath from the active menu system
{DELETEMENUITEM MenuPath}	Deletes the menu item MenuPath from the active menu system
{SETMENUBAR MenuDefinition}	Sets the active menu system

See Also
Using Macros

Object Macro Commands

Object commands let you move and resize objects on the Graphs page either in a graph window, in a notebook window, or in a dialog window. These commands can also create objects or change the property settings of Quattro Pro objects. Object commands can affect objects in any Quattro Pro window. See Property Reference for a list of objects and properties you can manipulate with Object macro commands:

{COLUMNWIDTH Block, FirstPane?, Set/Reset/Auto, Size}	Sets the width of columns in Block
{CREATEOBJECT ObjectName, x1,y1,x2,y2,<x3, y3,...>}	Creates graph or dialog objects
{FLOATCREATE Type,UpperCell, xoffset,yoffset,LowerCell,xoffset2, yoffset2,Text}	Creates a floating graph or SpeedButton
{FLOATMOVE UpperCell, xoffset,yoffset}	Moves a floating object in the active notebook window
{FLOATSIZE UpperCell,xoffset,yoffset, LowerCell,xoffset2,yoffset2}	Resizes a selected floating object in the active notebook window
{GETOBJECTPROPERTY Cell, Object.Property}	Gets property setting from an object
{GETPROPERTY Cell, Property}	Gets property setting from the object currently selected
{MOVETO x,y}	Moves selected objects to coordinates x, y
{RESIZE x,y,NewWidth,NewHeight, <VertFlip?>,<HorizFlip?>}	Resizes selected objects in the active window
{ROWCOLSHOW Block,Show?, Row or Col,FirstPane?}	Shows or hides rows or columns in Block
{ROWHEIGHTBlock,FirstPane?, Set/Reset,Size}	Sets the height of rows in Block
{SELECTBLOCK Block}	Selects a contiguous or noncontiguous block within the active notebook
{SELECTFLOAT ObjectID1 <,ObjectID2, ...>}	Selects floating objects in the active notebook window using their object names
{SELECTOBJECT ObjectID1 <,ObjectID2, ...>}	Selects objects in active window
{SETGRAPHATTR FillColor,BkgColor, FillStyle,BorderColor,BoxType}	Equivalent to setting Graph Setup and Background properties (colors are comma-separated RGB triplets in quotes)
{SETOBJECTPROPERTY Object.Property,Value}	Sets properties of objects

{SETPROPERTY Property,Setting} Sets properties of selected object

See Also
Using Macros

Miscellaneous Macro Commands

These commands insert characters, comments, and perform a variety of other tasks. The slide commands at the end of the list are used to build and run slide shows:

} or { } }	Inserts a right brace, }
{ { }	Inserts a left brace, {
{ ~ }	Inserts a tilde, ~
{_}	Does nothing; lets you reserve space in a macro without stopping macro execution
{; String}	Lets you enter explanatory remarks in a macro
{HLINE Distance}	Scrolls window horizontally Distance lines
{HPAGE Distance}	Scrolls window horizontally Distance pages
{VLINE Distance}	Scrolls window vertically Distance lines
{VPAGE Distance}	Scrolls window vertically Distance pages

For slide show commands, see {Slide.*Option*}.

See Also
Using Macros

Command Equivalent Macros

Use the following command equivalent macros in place of menu commands. When used with @COMMAND, they return current command settings. For a list of command equivalents sorted by menu command, see Command Equivalents by Command.

Command Equivalent Macro	Menu Command
{Align.*Option*}	Dialog|Align
{Application.*Property*}	Property|Application
{BlockCopy *Argument(s)*}	Block|Copy
{BlockDelete.*Option*}	Block|Delete
{BlockFill.*Option*}	Block|Fill
{BlockInsert.*Option*}	Block|Insert
{BlockMove *Argument(s)*}	Block|Move
{BlockMovePages *Argument(s)*}	Block|Move Pages
{BlockName.*Option*}	Block|Names
{BlockReformat *Argument(s)*}	Block|Reformat
{BlockTranspose *Argument(s)*}	Block|Transpose
{BlockValues *Argument(s)*}	Block|Values
{ClearContents *Argument(s)*}	Edit|Clear Contents
{Consolidate.*Option*}	Tools|Consolidator
{Controls.*Option*}	Dialog|Order
{DataModel *Arguments*}	Data|Data Modeling Desktop
{DialogView *Window*}	Tools|UI Builder
{DialogWindow.*Property*}	Property|Dialog Box
{EditClear}	Edit|Clear
{EditCopy}	Edit|Copy
{EditCut}	Edit|Cut
{EditGoto *Argument(s)*}	Edit|Goto
{EditPaste}	Edit|Paste
{ExportGraphic *Argument(s)*}	Draw|Export
{FileClose...}	File|Close, File|Close All
{FileCombine *Argument(s)*}	Tools|Combine
{FileExit *Argument(s)*}	File|Exit
{FileExtract *Argument(s)*}	Tools|Extract
{FileImport *Argument(s)*}	Tools|Import
{FileNew}	File|New
{FileOpen *Argument(s)*}	File|Open
{FileRetrieve *Argument(s)*}	File|Retrieve
{FileSave...}	File|Save, File|Save As, File|Save All

{FloatOrder.*Option*}	Block|Object Order
{Form <*Block*>}	Data|Form
{Frequency.*Option*}	Data|Frequency
{GraphCopy *Argument(s)*}	Graph|Copy
{GraphDelete *Argument(s)*}	Graph|Delete
{GraphEdit *Argument(s)*}	Graph|Edit
{GraphNew *Argument(s)*}	Graph|New
{GraphSettings.Titles *Argument(s)*}	Graph|Titles
{GraphSettings.Type *Argument(s)*}	Graph|Type
{GraphView}	Graph|View
{GraphWindow.*Property*}	Property|Graph Window
{Group.*Option*}	Tools|Define Group
{GroupObjects}	Draw|Group
{ImportGraphic *Argument(s)*}	Draw|Import
{InsertBreak}	Block|Insert Break
{InsertObject *Argument(s)*}	Edit|Insert Object
{Invert.*Option*}	Tools|Advanced Math|Invert
{Links.*Option*}	Tools|Update Links
{Multiply.*Option*}	Tools|Advanced Math|Multiply
{NamedStyle.*Option*}	Edit|Define Style
{Notebook.*Property*}	Property|Active Notebook
{OLE.*Option*}	OLE Object
{Optimizer.*Option*}	Tools|Optimizer
{Order.*Option*}	Dialog|Order and Draw Menu (Forward/To Front and Send Backward/To Back commands only)
{Page.*Property*}	Property|Active Page
{Parse.*Option*}	Data|Parse
{PasteFormat *Argument(s)*}	Edit|Paste Format
{PasteLink}	Edit|Paste Link
{PasteSpecial *Argument(s)*}	Edit|Paste Special
{Preview}	File|Print Preview
{Print.*Option*}	File|Page Setup, and File|Print, and File|Named Settings
{PrinterSetup *Argument(s)*}	File|Printer Setup
{Query.*Option*}	Data|Query
{Regression.*Option*}	Tools|Advanced Math|Regression
{ResizeToSame}	Dialog|Align
{RestrictInput.*Option*}	Data|Restrict Input or any operation that ends INPUT mode
{Scenario.*Option*}	Tools|Scenario Manager
{Search.*Option*}	Edit|Search and Replace
{Series.*Option*}	Graph|Series

{Slide.*Option*}	Graph|Slide Show
{SolveFor.*Option*}	Tools|Solve For
{Sort.*Option*}	Data|Sort
{SpeedBar.*Option*}	SpeedBar Control Menu
{SpellCheck.*Option*}	Tools|Spell Check (Workgroup edition only)
{TableQuery.*Option*}	Data|Table Query
{TableView}	Data|Database Desktop
{UngroupObjects}	Draw|Ungroup
{WhatIf.*Option*}	Data|What-If
{WindowArrIcon}	Window|Arrange Icons
{WindowCascade}	Window|Cascade
{WindowClose}	Control Menu|Close
{WindowHide}	Window|Hide
{WindowMaximize}	Control Menu|Maximize
{WindowMinimize}	Control Menu|Minimize
{WindowMove *Argument(s)*}	Control Menu|Move
{WindowNewView}	Window|New View
{WindowNext}	Control Menu|Next
{WindowPanes}	Window|Panes
{WindowRestore}	Control Menu|Restore
{WindowShow *Argument(s)*}	Window|Show
{WindowSize *Argument(s)*}	Control Menu|Size
{WindowTile}	Window|Tile
{WindowTitles *Argument(s)*}	Window|Locked Titles
{WorkSpace.*Option*}	File|Workspace

See Also
Using Macros

Command Equivalents by Command

The following list shows the command equivalent macro for each menu command. Use the command equivalents in place of menu commands. When used with @COMMAND, they return current command settings.

Menu Command	Command Equivalent Macro
Block|Copy	{BlockCopy *Argument(s)*}
Block|Delete	{BlockDelete.*Option*}
Block|Fill	{BlockFill.*Option*}
Block|Insert	{BlockInsert.*Option*}
Block|Insert Break	{InsertBreak}
Block|Move	{BlockMove *Argument(s)*}
Block|Move Pages	{BlockMovePages *Argument(s)*}
Block|Names	{BlockName.*Option*}
Block|Object Order	{FloatOrder.*Option*}
Block|Reformat	{BlockReformat *Argument(s)*}
Block|Transpose	{BlockTranspose *Argument(s)*}
Block|Values	{BlockValues *Argument(s)*}
Control Menu|Close	{WindowClose}
Control Menu|Maximize	{WindowMaximize}
Control Menu|Minimize	{WindowMinimize}
Control Menu|Move	{WindowMove *Argument(s)*}
Control Menu|Next	{WindowNext}
Control Menu|Restore	{WindowRestore}
Control Menu|Size	{WindowSize *Argument(s)*}
Data|Database Desktop	{TableView}
Data|Data Modeling Desktop	{DataModel *Arguments*}
Data|Form	{Form <*Block*>}
Data|Frequency	{Frequency.*Option*}
Data|Parse	{Parse.*Option*}
Data|Query	{Query.*Option*}
Data|Restrict Input	{RestrictInput.*Option*}
Data|Sort	{Sort.*Option*}
Data|Table Query	{TableQuery.*Option*}
Data|What-If	{WhatIf.*Option*}
Data|Workgroup Desktop	None
Dialog|Align	{Align.*Option*}
Dialog|Align	{ResizeToSame}
Dialog|Order	{Controls.*Option*}

Dialog|Order	{Order.*Option*}
Draw|Bring Forward	{Order.*Option*}
Draw|Bring to Front	{Order.*Option*}
Draw|Export	{ExportGraphic *Argument(s)*}
Draw|Group	{GroupObjects}
Draw|Import	{ImportGraphic *Argument(s)*}
Draw|Send Backward	{Order.*Option*}
Draw|Sent to Back	{Order.*Option*}
Draw|Ungroup	{UngroupObjects}
Edit|Clear	{EditClear}
Edit|Clear Contents	{ClearContents *Argument(s)*}
Edit|Copy	{EditCopy}
Edit|Cut	{EditCut}
Edit|Define Style	{NamedStyle.*Option*}
Edit|Goto	{EditGoto *Argument(s)*}
Edit|Insert Object	{InsertObject *Argument(s)*}
Edit|Paste	{EditPaste}
Edit|Paste Format	{PasteFormat *Argument(s)*}
Edit|Paste Link	{PasteLink}
Edit|Paste Special	{PasteSpecial *Argument(s)*}
Edit|Search and Replace	{Search.*Option*}
File|Close	{FileClose...}
File|Close All	{FileClose...}
File|Exit	{FileExit *Argument(s)*}
File|Named Settings	{Print.*Option*}
File|New	{FileNew}
File|Open	{FileOpen *Argument(s)*}
File|Page Setup	{Print.*Option*}
File|Print	{Print.*Option*}
File|Print Preview	{Preview}
File|Printer Setup	{PrinterSetup *Argument(s)*}
File|Retrieve	{FileRetrieve *Argument(s)*}
File|Save	{FileSave...}
File|Save All	{FileSave...}
File|Save As	{FileSave...}
File|Workspace	{WorkSpace.*Option*}
Graph|Copy	{GraphCopy *Argument(s)*}
Graph|Delete	{GraphDelete *Argument(s)*}
Graph|Edit	{GraphEdit *Argument(s)*}
Graph|New	{GraphNew *Argument(s)*}

Graph|Series	{Series.*Option*}
Graph|Slide Show	{Slide.*Option*}
Graph|Titles	{GraphSettings.Titles *Argument(s)*}
Graph|Type	{GraphSettings.Type *Argument(s)*}
Graph|View	{GraphView}
Property|Active Notebook	{Notebook.*Property*}
Property|Active Page	{Page.*Property*}
Property|Application	{Application.*Property*}
Property|Dialog Box	{DialogWindow.*Property*}
Property|Graph Window	{GraphWindow.*Property*}
SpeedBar Control Menu|Append	{SpeedBar.Append *Argument(s)*}
SpeedBar Control Menu|Insert	{SpeedBar.Insert *Argument(s)*}
SpeedBar Control Menu|Replace	{SpeedBar.Replace *Argument(s)*}
SpeedBar Control Menu|Remove	{SpeedBar.Remove *BarName*}
Tools|Advanced Math|Invert	{Invert.*Option*}
Tools|Advanced Math|Multiply	{Multiply.*Option*}
Tools|Advanced Math|Regression	{Regression.*Option*}
Tools|Analysis Tools	See Analysis Tools Macro Commands
Tools|Combine	{FileCombine *Argument(s)*}
Tools|Consolidator	{Consolidate.*Option*}
Tools|Define Group	{Group.*Option*}
Tools|Extract	{FileExtract *Argument(s)*}
Tools|Import	{FileImport *Argument(s)*}
Tools|Optimizer	{Optimizer.*Option*}
Tools|Scenario Manager	{Scenario.*Option*}
Tools|Solve For	{SolveFor.*Option*}
Tools|Spell Check	{SpellCheck.*Option*} (Workgroup edition only)
Tools|UI Builder	{DialogView *Window*}
Tools|Update Links	{Links.*Option*}
Window|Arrange Icons	{WindowArrIcon}
Window|Cascade	{WindowCascade}
Window|Hide	{WindowHide}
Window|Locked Titles	{WindowTitles *Argument(s)*}
Window|New View	{WindowNewView}
Window|Panes	{WindowPanes}
Window|Show	{WindowShow *Argument(s)*}
Window|Tile	{WindowTile}

See Also
Using Macros

Analysis Tools Macro Commands

Analysis Tools commands let you perform analyses of numerical data. They are equivalent to the tools available when you choose Tools|Analysis Tools.

{ANOVA1 *InBlock*,*OutBlock*,<*Grouped*>, <*Labels*(0|1)>,<*Alpha*>}	One-way analysis of variance
{ANOVA2 *InBlock*,*OutBlock*, *SampleRows*,<*Alpha*>}	Two-way analysis of variance with replication
{ANOVA3 *InBlock*,*OutBlock*, <*Labels*(0|1)>,<*Alpha*>}	Two-way analysis of variance without replication
{DESCR *InBlock*,*OutBlock*,<*Grouped*>, <*Labels*(0|1)>,<*Summary*(0|1)>, <*Largest*>,<*Smallest*>, <*Confidence*>}	Creates a table of descriptive statistics that characterize one or more samples
{EXPON *InBlock*,*OutBlock*,<*Damping*>, <*StdErrs*(0|1)>}	Exponentially smooths a series of values
{FOURIER *InBlock*,*OutBlock*, <*Inverse*(0|1)>}	Performs a fast Fourier transformation on a block of data
{FTESTV *InBlock1*,*InBlock2*,*OutBlock*, <*Labels*(0|1)>}	Performs a two-sample F-Test to compare population variances
{HISTOGRAM *InBlock*,*OutBlock*, <*BinBlock*>,<*Pareto*(0|1)>,<*Cum*(0|1)>}	Calculates the probability and cumulative distributions for a sample based on a series of bins
{MCORREL *InBlock*,*OutBlock*, <*Grouped*>,<*Labels*(0|1)>}	Returns the correlation matrix between two or more data sets
{MCOVAR *InBlock*,*OutBlock*,<*Grouped*>, <*Labels*(0|1)>}	Returns the covariance matrix between two or more data sets
{MOVEAVG *InBlock*,*OutBlock*,<*Interval*>, <*StdErrs*(0|1)>}	Returns a moving average for the values in a sample
{MTGAMT <*OutBlock*>,<*Rate*>,<*Term*>, <*OrigBal*>,<*EndBal*>,<*LastYear*>}	Generates an amortization schedule for a mortgage
{MTGREFI <*OutBlock*>,<*CurrBal*>, <*CurrRate*>,<*RemTerm*>, <*CandPctFees*>,<*CandRate*>}	Generates a table of information relating to refinancing a mortgage
{PTTESTM *InBlock1*,*InBlock2*,*OutBlock*,	

{PTTESTV *InBlock1*,*InBlock2*,*OutBlock*, <*Labels*(0|1)>,<*Alpha*>,<*Difference*>}	Performs a paired two-sample Student's t-Test for means
{PTTESTV *InBlock1*,*InBlock2*,*OutBlock*, <*Labels*(0|1)>,<*Alpha*>}	Performs a Student's t-Test using two independent samples with unequal variances
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*LowerBound*, *UpperBound*}	Returns random values from a uniform distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*Mean*,*SDev*}	Returns random values from a normal distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*Prob*}	Returns random values from a Bernoulli distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*Prob*,*Trials*}	Returns random values from a binomial distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*Lambda*}	Returns random values from a Poisson distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*LowerBound*, *UpperBound*,*Step*,*RepeatNumber*, *RepeatSequence*}	Returns random values from a patterned distribution
{RANDOM *OutBlock*,*Columns*,*Rows*, *Distribution*,*Seed*,*InBlock*}	Returns random values from a discrete distribution
{RANKPERC *InBlock*,*OutBlock*, <*Grouped*>,<*Labels*(0|1)>}	Computes the ordinal and percent rank of each value in a sample
{REGRESS *InBlockY*,*InBlockX*, *YIntZero*(0|1),*Labels*(0|1),*Confidence*, *SumOutBlock*,*Residuals*(0|1), *StdResidual*(0|1),<*ResidualOutBlock*>, <*ProbOutBlock*>}	Performs multiple linear regression analysis
{SAMPLE *InBlock*,*OutBlock*,*Type*, *Rate*}	Returns a periodic or random sample of values from *InBlock*
{TTESTM *InBlock1*,*InBlock2*,*OutBlock*, <*Labels*(0|1)>,<*Alpha*>,<*Difference*>}	Performs a Student's t-Test using two independent samples with equal variances
{ZTESTM *InBlock1*,*InBlock2*,*OutBlock*, <*Labels*(0|1)>,<*Alpha*>,<*Difference*>, <*Variance1*>,<*Variance2*>}	Performs a two-sample z-Test for means, assuming known variances for each sample

See Also

Using the Advanced Analysis Tools

Using Macros

Macro Command Index

{BREAKOFF}
{BREAKON}
{BS}
{CALC}
{CAPOFF}
{CAPON}
{CHOOSE}
{CLEAR}
{ClearContents}
{CLOSE}
{COLUMNWIDTH}
{Consolidate.*Option*}
{CONTENTS}
{Controls.*Option*}
{CR}
{CREATEOBJECT}
{CTRL}
{DATAMODEL}
{DATE}
{DEFINE}
{DEL}
{DELETE}
{DELETEMENU}
{DELETEMENUITEM}
{DESCR}
{DialogView *Window*}
{DialogWindow.*Property*}
{DISPATCH}
{DLL}
{DLL.Load}
{DODIALOG}
{DOWN}
{D}
{EDIT}
{EditClear}
{EditCopy}
{EditCut}
{EditGoto}

{EditPaste}
{END}
{ESC}
{ESCAPE}
{EXEC}
{EXECUTE}
{EXPON}
{ExportGraphic}
{FileClose...}
{FileCombine}
{FileExit}
{FileExtract}
{FileImport}
{FileNew}
{FileOpen}
{FileRetrieve}
{FILESIZE}
{FileSave...}
{FLOATCREATE}
{FLOATMOVE}
{FloatOrder.*Option*}
{FLOATSIZE}
{FOR}
{FORBREAK}
{FORM}
{FOURIER}
{Frequency.*Option*}
{FTESTV}
{FUNCTIONS}
{GET}
{GETDIRECTORYCONTENTS}
{GETLABEL}
{GETNUMBER}
{GETOBJECTPROPERTY}
{GETPOS}
{GETPROPERTY}
{GETWINDOWLIST}
{GOTO}

{GRAPH}
{GRAPHCHAR}
{GraphCopy}
{GraphDelete}
{GraphEdit}
{GraphNew}
{GRAPHPAGEGOTO}
{GraphSettings.Titles}
{GraphSettings.Type}
{GraphView}
{GraphWindow.*Property*}
{Group.*Option*}
{GroupObjects}
{HELP}
{HISTOGRAM}
{HLINE}
{HOME}
{HPAGE}
{IF}
{IFKEY}
{IMFORMAT}
{ImportGraphic}
{INDICATE}
{INITIATE}
{INS}
{INSERT}
{InsertBreak}
{InsertObject}
{INSOFF}
{INSON}
{INSPECT}
{Invert.*Option*}
{L}
{LEFT}
{LET}
{Links.*Option*}
{LOOK}
{MACROS}
{MARK}

{MCORREL}
{MCOVAR}
{MENU}
{MENUBRANCH}
{MENUCALL}
{MESSAGE}
{MOVEAVG}
{MOVETO}
{MTGAMT}
{MTGREFI}
{Multiply.*Option*}
{NAME}
{NamedStyle.*Option*}
{NEXTPANE}
{NEXTTOPWIN}
{NEXTWIN}
{Notebook.*Property*}
{NUMOFF}
{NUMON}
{OLE.*Option*}
{ONERROR}
{OPEN}
{Optimizer.*Option*}
{Order.*Option*}
{Page.*Property*}
{PANELOFF}
{PANELON}
{Parse.*Option*}
{PasteFormat}
{PasteLink}
{PasteSpecial}
{PAUSEMACRO}
{PGDN}
{PGUP}
{POKE}
{Preview}
{Print.*Option*}
{PrinterSetup}

{PTTESTM}
{PTTESTV}
{PUT}
{PUTBLOCK}
{PUTCELL}
{QGOTO}
{QUERY}
{Query.*Option*}
{QUIT}
{R}
{RANDOM}
{RANKPERC}
{READ}
{READLN}
{RECALC}
{RECALCCOL}
{REGRESS}
{Regression.*Option*}
{REQUEST}
{RESIZE}
{ResizeToSame}
{RESTART}
{RestrictInput.*Option*}
{RETURN}
{RIGHT}
{ROWCOLSHOW}
{ROWHEIGHT}
{SAMPLE}
{Scenario.*Option*}
{SCROLLOFF}
{SCROLLON}
{Search.*Option*}
{SELECTBLOCK}
{SELECTFLOAT}
{SELECTOBJECT}
{Series.*Option*}
{SETGRAPHATTR}
{SETMENUBAR}

{SETOBJECTPROPERTY}
{SETPOS}
{SETPROPERTY}
{SHIFT}
{Slide.*Option*}
{SolveFor.*Option*}
{Sort.*Option*}
{SpeedBar.*Option*}
{SPEEDFILL}
{SPEEDFORMAT}
{SPEEDSUM}
{SpellCheck.*Option*}
{STEP}
{STEPOFF}
{STEPON}
{*Subroutine*}
{TAB}
{TABLE}
{TableQuery.*Option*}
{TableView}
{TERMINATE}
{TTESTM}
{U}
{UNDO}
{UngroupObjects}
{UP}
{VLINE}
{VPAGE}
{WAIT}
{WhatIf.*Option*}
{WINDOW}
{WindowArrIcon}
{WindowCascade}
{WindowClose}
{WindowHide}
{WindowMaximize}
{WindowMinimize}
{WindowMove}
{WindowNewView}

{WindowNext}
{WINDOWSOFF}
{WINDOWSON}
{WindowPanes}
{WindowRestore}
{WindowShow}
{WindowSize}
{WindowTile}
{WindowTitles}
{WorkSpace.*Option*}
{WRITE}
{WRITELN}
{ZOOM}
{ZTESTM}

Quattro Pro DOS Compatible Commands

This command or menu equivalent is provided for compatibility with the DOS version of Quattro Pro. To learn more about it, see the Quattro Pro for DOS *@Functions and Macros* guide.

See Also

Quattro Pro for DOS Macros

Using Macros

Macro Command Descriptions

New Macro Commands

For the complete list of Quattro Pro macro commands, see the Macro Command Index.

{ANOVA1}

{ANOVA2}

{ANOVA3}

{Consolidate.*Option*}

{DATAMODEL}

{DESCR}

{DLL}

{DLL.Load}

{EXPON}

{FORM}

{FOURIER}

{FTESTV}

{HISTOGRAM}

{IMFORMAT}

{MCORREL}

{MCOVAR}

{MOVEAVG}

{MTGAMT}

{MTGREFI}

{PTTESTM}

{PTTESTV}

{RANDOM}

{RANKPERC}

{REGRESS}

{SAMPLE}

{Scenario.*Option*}

{SpeedBar.*Option*}

{SpellCheck.*Option*}

{TTESTM}

{ZTESTM}

Block Menu

Move	Lets you move data from one area to another.
Copy	Lets you copy data from one area to another.
Insert	Lets you insert blank rows, columns, blocks, or pages.
Delete	Lets you delete rows, columns, blocks, or pages.
Fill	Automatically enters a series of values into a block.
Names	Lets you name blocks so that you can refer to them by name instead of by their coordinates.
Transpose	Reverses columns and rows.
Values	Copies values calculated by formulas.
Reformat	Formats a block of text to fill given columns.
Move Pages	Lets you reposition pages in a notebook.
Insert Break	Forces a page break during output.
Object Order	Reorders layers of overlapping floating objects.

Block | Move

The Block|Move command moves the contents and properties of a block from one location to another. It overwrites existing data in the new location. The options are:

From	The block to be moved. Legal entries are block coordinates or named blocks (but not noncontiguous blocks).
To	The destination block for the moved data. Enter the upper left cell of the target block.

As when you use the Clipboard commands, Edit|Cut and Edit|Paste, in sequence, using Block|Move moves a cell or a block, including any floating objects fully enclosed in the block. However, to copy data or objects other than cell blocks, use the Clipboard commands.

The advantages of Block|Move over the Clipboard commands are

You can specify blocks to be moved by name.

Block|Move works a bit faster than Edit|Cut followed by Edit|Paste.

If you're already familiar with Block|Move, or if you have macros that use it, it's more convenient.

See Also

Using Block Commands for Copying and Moving

Copying and Moving with the Windows Clipboard

Dragging and Dropping

Moving Formulas or Referenced Cells

Using Named Blocks

Block | Copy

The Block|Copy command copies the contents and properties of a block from one location to another. It overwrites existing data in the new location. The options are:

From	The block to be copied. Legal entries are block coordinates or named blocks, including noncontiguous blocks.
To	The destination block for the moved data. Enter the upper left cell of the target block to make a single copy. If the source block is single cell, column, or row, you can make multiple copies by selecting a block of cells as the destination.
Model Copy	If the source block contains formulas with absolute references to cells within it, the formulas are adjusted, but remain absolute. See Using the Model Copy Option for details.

Block|Copy copies a cell, a block, or noncontiguous blocks the same as using the Clipboard commands, Edit|Copy and Edit|Paste, in sequence. To copy objects other than cell blocks, use the Clipboard commands. The advantages of Block|Copy over the Clipboard commands are

You can specify blocks to be copied by name.

Block|Copy works a bit faster than Edit|Copy followed by Edit|Paste.

Block|Copy provides a unique, time-saving feature called Model Copy. Use it to copy blocks that contain absolute references to cells within the copied block.

If you're already familiar with Block|Copy, or if you have macros that use it, it's more convenient.

See Also

Using Block Commands for Copying and Moving

Copying and Moving with the Windows Clipboard

Using Named Blocks

Copying Formulas

Block | Insert

The Block|Insert command inserts blank columns, rows, or pages anywhere in the notebook. You can also insert partial rows, columns, or pages. You can even insert the contents of a different notebook file into the active notebook. Wherever you insert, existing data is pushed down, to the right, or to the back of the notebook to make room.

For information on the dialog boxes opened by each command in the Insert menu, see

Block|Insert|Rows

Block|Insert|Columns

Block|Insert|Pages

Block|Insert|File

See Also

Inserting Rows or Columns

Inserting Pages

Inserting Blocks

Inserting a File into a Notebook

Block | Insert | Rows

The Block|Insert|Rows command inserts blank entire or partial rows anywhere in the notebook. Wherever you insert, existing data is pushed down to make room. The options are:

Block	The size of the block to be inserted; the upper left cell is the first cell entry to be shifted down. To insert only one row, enter any cell address in the row to insert before, such as A:B3. To insert multiple rows, enter coordinates of a block that spans the same number of rows you want to insert. For example, to insert three rows before row 3 (on page A), enter A:B3..B5.
Entire	Inserts whole rows.
Partial	Inserts only partial rows (a block).

See Also

Inserting Rows or Columns

Inserting Blocks

Block | Insert | Columns

The Block|Insert|Columns command inserts blank entire or partial columns anywhere in the notebook. Wherever you insert, existing data is pushed to the right to make room. The options are:

Block	The size of the block to be inserted; the upper left cell is the first cell entry to be shifted to the right. To insert only one column, enter any cell address in the column to insert before, such as A:B3. To insert multiple columns, enter coordinates of a block that spans the same number of columns you want to insert. For example, to insert three columns before column B (on page A), enter A:B3..D3.
Entire	Inserts whole columns.
Partial	Inserts only partial columns (a block).

See Also

Inserting Rows or Columns

Inserting Blocks

Block | Insert | Pages

The Block|Insert|Pages command inserts blank entire or partial pages anywhere in the notebook. Wherever you insert, existing data is pushed onto following pages to make room. The options are:

Block	The size of the block to be inserted; the upper left cell is the first cell entry to be shifted to the back. To insert only one page, enter any cell address on the page to insert before, such as B:B3. To insert multiple pages, enter a 3-D block starting with the page to insert before and spanning the same number of pages you want to insert. For example, to insert three pages before page B, enter B:B3..D:B3.
Entire	Inserts whole pages.
Partial	Inserts only partial pages (a block).

See Also

Inserting Pages

Inserting Blocks

Block | Insert | File

The Block|Insert|File command inserts the contents of a file anywhere in the notebook. Wherever you insert, existing data is pushed onto following pages to make room. Only the pages from the inserted file that contain data are inserted into the active notebook.

The standard options in the Insert File dialog box are explained in File-Handling Options. In addition, there is another option:

Before Page	Any cell on the page before which you want to insert the file.

See Also

Inserting a File into a Notebook

Inserting Blocks

Block | Delete

The Block|Delete command deletes entire or partial columns, rows, or pages anywhere in the notebook. Deleting space is different from erasing data with Edit|Clear, Clear Contents, or the Del key. Deleting with Block|Delete (or with the Delete button in the SpeedBar) makes remaining rows, columns, or pages move to take up the deleted space.

For information on the dialog boxes opened by each command in the Delete menu, see

Block|Delete|Rows

Block|Delete|Columns

Block|Delete|Pages

See Also

Deleting Rows or Columns

Deleting Pages

Deleting Blocks

Using Named Blocks

Block | Delete | Rows

The Block|Delete|Rows command deletes entire or partial rows anywhere in the notebook. Deleting space is different from erasing data with Edit|Clear, Clear Contents, or the Del key; deleting with Block|Delete or with the Delete button in the SpeedBar makes remaining rows move up to take up the deleted space. The options are:

Block	The size of the block to be deleted. To delete only one row, enter any cell address in the row to delete, such as A:B3. To delete multiple rows, enter coordinates of a block that spans rows you want to delete. For example, to delete three rows starting with row 3 (on page A), enter A:B3..B5.
Entire	Deletes whole rows.
Partial	Deletes only partial rows (the Block option entry).

Caution: If you delete space that is within the boundaries of a named block or a block referenced by a formula, Quattro Pro adjusts all references to the block. However, if one of the deleted cells is a coordinate cell that defines a block, the block becomes invalid and any formulas or names referencing the block show ERR. Any formulas that reference a cell within a deleted column, row, or page also appear as ERR.

See Also

Deleting Rows or Columns

Deleting Blocks

Using Named Blocks

Block | Delete | Columns

The Block|Delete|Columns command deletes entire or partial columns anywhere in the notebook. Deleting space is different from erasing (or blanking) data with Edit|Clear, Clear Contents, or the Del key; deleting with Block|Delete or with the Delete button in the SpeedBar makes remaining columns move left to take up the deleted space. The options are:

Block	The size of the block to be deleted. To delete only one column, enter any cell address in the column to delete, such as A:B3. To delete multiple columns, enter coordinates of a block that spans the columns you want to delete. For example, to delete three columns starting with column B (on page A), enter A:B3..D3.
Entire	Deletes whole columns.
Partial	Deletes only partial columns (the Block option entry).

Caution: If you delete space that is within the boundaries of a named block or a block referenced by a formula, Quattro Pro adjusts all references to the block. However, if one of the deleted cells is a coordinate cell that defines a block, the block becomes invalid and any formulas or names referencing the block show ERR. Any formulas that reference a cell within a deleted column, row, or page also appear as ERR.

See Also

Deleting Rows or Columns

Deleting Blocks

Using Named Blocks

Block | Delete | Pages

The Block|Delete|Pages command deletes entire or partial pages anywhere in the notebook. Deleting space is different from erasing data with Edit|Clear, Clear Contents, or the Del key; deleting with Block|Delete or with the Delete button in the SpeedBar makes remaining pages move forward to take up the deleted space. The options are:

Block	The size of the block to be deleted. To delete only one page, enter any cell address on the page to delete, such as B:B3. To delete multiple pages, enter a 3-D block spanning the pages you want to delete. For example, to delete three pages starting with page B, enter B:B3..D:B3.
Entire	Inserts whole pages.
Partial	Inserts only partial pages (the Block option entry).

Caution: If you delete space that is within the boundaries of a named block or a block referenced by a formula, Quattro Pro adjusts all references to the block. However, if one of the deleted cells is a coordinate cell that defines a block, the block becomes invalid and any formulas or names referencing the block show ERR. Any formulas that reference a cell within a deleted column, row, or page also appear as ERR.

See Also

Deleting Pages

Deleting Blocks

Using Named Blocks

Block | Fill

The Block|Fill command automatically fills a block with values that increase or decrease in a specific pattern. The options are:

Blocks	The block to fill with values.
Start	The first value in the series. Enter a number, a formula, a date, or a time.
Step	The constant value to add (or multiply or use as an exponent) with the Start value or the previous value.
Stop	The limit for the fill values.
Order	The Column setting fills by moving down columns, starting with the leftmost column. The Row setting fills across rows, starting with the top row.
Series	The type of fill operation to perform. If the Start value is a number or a formula, you can choose Linear (addition), Growth (multiplication), or Power (exponentiation). If the Start value is a date or time, addition is assumed, and you can use one of the remaining Series settings to determine the time unit (Year, Month, and so on) to use.

See Also

Entering a Sequence of Numbers

Using SpeedFill

Block | Names

The Block|Names command assigns, deletes, and creates a table of block names. Block names provide a more efficient way of referencing a block than using its coordinates.

For information on the dialog boxes opened by each command in the Names menu, see

Create	Assigns a name to a block.
Delete	Removes an assigned block name.
Labels	Assigns names to single cells using adjacent labels.
Reset	Erases all existing block names in the notebook.
Make Table	Creates a table in the notebook listing all named blocks by name and location.
Auto Generate	Automatically creates names from adjacent labels.

See Also

Using Named Blocks
Naming Blocks
Using Block Names in Formulas
Naming Single-Cell Blocks from Labels
Naming Blocks and Cells from Labels
Making a Table of Named Blocks
Deleting Block Names

Block | Names | Create

The Block|Names|Create command assigns a name to a block. The options are:

Name	The name to assign to the block.
Block(s)	The coordinates of the block to be named.
Names	Other named blocks existing in the active notebook.

See Also

Naming Blocks
Using Named Blocks
Using Block Names in Formulas
Naming Single-Cell Blocks from Labels
Naming Blocks and Cells from Labels
Making a Table of Named Blocks
Deleting Block Names

Block | Names | Delete

The Block|Names|Delete command removes an assigned block name. The options are:

Name	The block name to be deleted.
Block(s)	The coordinates of the block name to be deleted.
Names	Named blocks existing in the active notebook.

When you delete a block name, the entries in the block itself are unaffected. Also, formulas referencing the name change to reference the block's coordinates; their results don't change.

See Also

Deleting Block Names
Using Named Blocks
Naming Blocks
Using Block Names in Formulas
Naming Single-Cell Blocks from Labels
Naming Blocks and Cells from Labels
Making a Table of Named Blocks

Block | Names | Labels

The Block|Names|Labels command assigns names to single cells using adjacent labels. The options are:

Directions	The direction from the Blocks entry to find the cells to be named. For example, the Right setting assigns names to cells to the right of the cells in the Blocks entry.
Blocks	The coordinates of the *labels* to use as block names. If the entries are values instead of labels, they're ignored.

See Also

Naming Single-Cell Blocks from Labels

Naming Blocks and Cells from Labels

Using Named Blocks

Naming Blocks

Using Block Names in Formulas

Making a Table of Named Blocks

Deleting Block Names

Block | Names | Reset

The Block|Names|Reset command erases all existing block names in the notebook.

Although choosing Block|Names|Reset deletes all block names in the notebook from memory, the notebook itself is not affected. Formulas that reference named blocks change so they refer to block coordinates instead of names.

See Also

Using Named Blocks

Naming Blocks

Using Block Names in Formulas

Naming Single-Cell Blocks from Labels

Naming Blocks and Cells from Labels

Making a Table of Named Blocks

Deleting Block Names

Block | Names | Make Table

The Block|Names|Make Table command creates a table of entries in the notebook listing all named blocks by name and location. The option is:

Block	The address of the upper left cell where you want the table to be placed.

Caution: Make sure there is enough room for a two-column table, with one row for each block name. Quattro Pro overwrites existing data in cells it uses for the table.

The first column of the table lists block names alphabetically. The second indicates the corresponding block coordinates.

Quattro Pro doesn't update a named block table. If you add, change, or delete block names, you must re-create the table to reflect the changes. Putting a SpeedButton next to this table is a handy way of updating the table quickly.

See Also

Creating SpeedButtons

Using Named Blocks

Naming Blocks

Using Block Names in Formulas

Naming Single-Cell Blocks from Labels

Naming Blocks and Cells from Labels

Deleting Block Names

Block | Names | Auto Generate

The Block|Names|Auto Generate command automatically uses selected labels as names for adjacent blocks and cells.

Blocks Specifies the block of cells to name, including the labels to use.

When checked:

Under Top Row	Names each column included in Blocks by the label in its first row.
Right of Leftmost Column	Names each row included in Blocks by the label in its leftmost column.
Above Bottom Row	Names each column included in Blocks by the label in its bottom row.
Left of Rightmost Column	Names each row included in Blocks by the label in its rightmost column.
Name Cells at Intersections	Names each cell included in Blocks by the label in the top row of each column combined with the label in the leftmost cell of each row.

Names can be up to 15 characters long. Quattro Pro truncates longer names.

See Also

Naming Blocks and Cells from Labels

Using Named Blocks

Naming Blocks

Using Block Names in Formulas

Naming Single-Cell Blocks from Labels

Deleting Block Names

Block | Transpose

The Block|Transpose command reverses columns and rows within a selected block. You can either overwrite the original block with the new, reversed block, or retain the original block and place the new block at another location. The options are:

From	The block to be transposed.
To	The upper left cell of the block to which you want to copy the transposed data. Don't specify a cell within the source block--you will disarrange the data if you do. Be sure also that there is enough room for the transposed data, which overwrites any existing data in the destination block.

Caution: Don't transpose cells containing formulas. Cell references aren't adjusted properly when you use Block|Transpose.

See Also
Transposing Columns and Rows

Block | Values

The Block|Values command copies the values calculated by formulas without copying the formulas. Resulting values are usually numeric, but they can also be string values, depending on your formulas. You can copy these values to another part of the notebook or copy them over the formulas that computed them. The options are:

From	The block of values to be copied.
To	The upper left cell of the block of the block to which you want to copy the values. Be sure that there is enough room for the copied values, which overwrite any existing data in the destination block.

See Also
Converting Formulas to Values

Block | Reformat

The Block|Reformat command lets you adjust word wrapping in a series of label entries as though they were in a paragraph. You can enter text as a long label or a series of long labels, then reorganize the text as a paragraph, taking up as many cells or columns as you want. The option is:

Block	The coordinates of the block to contain the reformatted text. The upper left cell should be the first cell to reformat. If the lower right cell is in the same row, the text fills up as many rows as necessary to display the text within the columns containing those cells. If the lower right cell is in a row below, the text is reformatted within the block if there is enough room.

See Also
Reformatting Text Entries

Block | Move Pages

The Block|Move Pages command changes the order of pages within a notebook. The options are:

Move Page	The page(s) to be moved.
To Before Page	The page before which the moved pages will be placed.

See Also

Moving Pages

Using Named Blocks

Block | Insert Break

The Block|Insert Break command manually inserts a page break for printing. Quattro Pro automatically inserts page breaks when printing, but those page breaks may not fall in the ideal location for the layout of your data. Page breaks inserted this way are stored with the notebook and remain there until you delete them.

See Also
Inserting Page Breaks

Block | Object Order

The Block|Object Order commands manages the placement of floating objects within front-to-back layers. An object that overlaps another is in a layer in front of the other object.

To reorder the layering of overlapping objects, select an object (it has handles around it when selected) and choose one of these commands:

Bring Forward	Moves the object one layer closer to the top.
Send Backward	Moves the object one layer closer to the bottom.
Bring to Front	Brings the object to the front or top of the layers.
Send to Back	Sends the object to the bottom, or the layer closest to the page.

See Also

Layering Floating Objects

Creating Floating Objects

Inserting OLE Objects

Data Menu

The Data commands help access information in a notebook set up as a database:

Sort	Arranges records in the database according to entries in specified fields.
Restrict Input	Restricts movement of the selector to cells that aren't protected.
Form	Creates data entry and query forms without programming.
Query	Searches the database for records that meet given requirements. Optionally, you can copy the records you find to another part of the notebook, or delete them.
Table Query	Performs a query-by-example operation in an external data table.
Parse	Breaks down long labels into separate cell entries.
What-If	Sets up tables for providing hypothetical formula values.
Frequency	Sets up tables showing the number of values in a block of cells that fall within specified intervals.
Database Desktop	Opens a window in which you can access an external database.
Data Modeling Desktop	Opens a window in which you can create crosstabs.
Workgroup Desktop	Allows you to exchange notebooks with other users on a network (Workgroup edition only).

See Also

About Databases

Database Guidelines

Data | Sort

Data|Sort arranges records (rows) in the order you specify, based on the entries in one or more fields (sort keys). This command displays the Sort dialog box, with these fields and buttons:

Block	The block to be sorted, including row labels, if any, but excluding column headings.
Column	Up to five columns of sort keys, in the order they are to be sorted. For example, if you want to sort a sales prospect file, the first key might be postal zip code with the last name of your contact as the second key.
Ascending	By default, each key is sorted in ascending order from smallest to largest (A-Z and 0-9). To sort in descending order, uncheck the box.
Data	Whether values or labels appear first in the sorted column. The values are placed in a sorted group separate from the labels (which are also sorted).
Labels	Whether text sorts in Dictionary order (ordinary alphabetizing rules) or Character Code order (according to character number--for example, uppercase letters before lowercase).
Reset	Clears all entries and restores defaults.

See Also

Performing a Sort

Sort Order

Sort Keys

Tips on Sorting

Data | Restrict Input

The Database|Restrict Input command limits movement of the selector to only those cells in a block that are not protected. It also deactivates menus, so that the only changes that can be made to the notebook are to data in the unprotected cells.

This command is used mostly in macro applications, to limit access to the notebook. To return to Ready mode and restore full notebook access, press Esc or press Enter with no data on the input line.

The Restrict Input dialog box displays a single option:

Input Range
Enter the name of the block where you want to allow data entry.

See Also
Limiting Data Entry
Data Commands

Data | Form

The Data|Form command lets you create a form for entering and finding data records without programming. This command displays the Database Form dialog box.

To create a form, enter the coordinates of the database block you want to use as a basis for the form. The block should have field labels in the top row and data in each column. Click OK to display the Edit Records dialog box.

See Also
Using Database Forms

Data | Query

The Data|Query command searches a Quattro Pro database to find records that meet your requirements, or criteria.Once you've set up a criteria table the search options tell Quattro Pro to search the database block and highlight all records meeting the criteria, copy all records or just unique records to a different part of the notebook, or delete the records. The Query dialog box contains these input boxes and buttons:

Database Block	specifies the block of data, including field names, to search.
Criteria Table	specifies the block containing search conditions, including field names.
Output Block	specifies the block where you want to copy field names and records that meet the criteria.
Locate	highlights all records that meet the search criteria.
Extract	copies all records that meet the search criteria to the output block.
Extract Unique	copies records like Extract, but eliminates duplicate records.
Delete	deletes all records that meet the search criteria.
Field Names	assigns block names to fields so you can use them in search formulas.
Reset	removes all block settings (Database Block, Criteria Table, and Output Block). It doesn't affect the criteria block or output block in your notebook, only the dialog box settings.

See Also

Setting Up a Database

Searching for Records

Highlighting Matching Records

Copying Matching Records

Deleting Matching Records

Using External Databases

Data | Table Query

The Data|Table Query command performs a query-by-example (QBE) operation in an external data table. With QBE, you create a particular kind of query in which you make a form that looks like an example of the records you want to search for. The Data|Table Query options are:

Source of Query	Specifies an external query file (with Query in File) or a block in the active notebook (with Query in Block) as the source of the query text.
QBE File/Block	Specifies the file name (if you specified Query in File) or block coordinates (if you specified Query in Block) of the query text. If running a query from a block, be sure to include all rows in the column that contain text.
Browse	Lets you choose the external query file to use. See File-Handling Options for details.
Destination	Specifies the target block for the query's Answer table (its results).

If Database Desktop isn't already running, Data|Table Query starts it as a minimized application.

Tip: To make a query flexible so the same text block searches for different field values, you can replace a selection condition (the part of a query statement that narrows the search) with a cell address. This lets you place the selection condition anywhere in a notebook (outside the query block).

To do this, you edit the cell entry that contains the selection condition to a formula instead of a label, then concatenate the cell address to the end of the query statement.

For example, if a query statement reads

```
'  |  Check  |
```

you can edit it to

```
+"  |  Check" &$B$1&"  |"
```

The last statement checks for a label in cell B1 and adds it to the selection condition.

See Types of Operators for details on using the & (concatenation) operator.

See Also
Database Desktop Help

Data | Parse

The Data|Parse command breaks down long labels into individual cells. This is often necessary after you've imported an ASCII text file, since Quattro Pro interprets each line of text as a single label.

You can also use Parse to break down data in a regular notebook. For example, you could turn a single column containing a first and last name into two columns: one containing the first name and the second containing the last name.

Tip: Before importing the text file to be parsed, format it in a monospace font, such as Courier, and make sure the text in each column of information starts at the same position.

Options

Input	Specifies the block containing the data to be parsed and the format line(s) to parse it.
Output	Specifies the destination for the parsed data.
Create	Inserts a format line in the active cell, moving existing rows down.
Edit	Opens a dialog box for changing format lines.
Reset	Blanks the input and output entries.

See Also

Importing Text Files

Tools|Import Command

Using Format Lines

Using Input and Output Blocks

Data | What-If

The Data|What-If command shows how a change in one or two variables affects the rest of your data. It builds "what-if" tables that are separate from your data to show both the substitutions and the effects they have on other cells.

Data|What-If displays a dialog box with these fields and buttons:

Table Type	specifies the number of variables in the what-if formulas. Choose One Free Variable to substitute for one variable when building the what-if table; choose Two Free Variables to build a two-variable table.
Data Table	specifies the block where you want to write the data table, including values for the first variable as the leftmost column and the formula row as the first row. (If you're using two variables, the first row includes one two-variable formula plus values for the second variable.)
Input Cell	(Column Input Cell, for two-variable tables) specifies the first (or only) cell referenced by what-if formula(s).
Row Input Cell	specifies the second cell referenced by a two-variable what-if formula.
Generate	builds the table.
Reset	clears all settings.

See Also

Creating One-Variable "What-if" Tables

Creating Two-Variable "What-if" Tables

Data | Frequency

The Data|Frequency command calculates the number of values that fall within given intervals and displays the results in a frequency distribution table. The command displays a dialog box with these fields and buttons:

Value Block(s)	specifies the block or list of blocks containing values to be counted.
Bin Block	specifies the block that defines value intervals or "bins" of values to be counted.
Reset	clears all settings.

See Also
Creating Frequency Distributions

Data | Database Desktop

The Data|Database Desktop command starts the Database Desktop application so that you can work with external databases in Quattro Pro. Using Database Desktop, you can

view, edit and query databases

copy data from a database into Quattro Pro

link database data to corresponding data in Quattro Pro and have your notebook updated whenever the database changes

query databases from within Quattro Pro and have the results returned directly to the notebook

See Also
Database Desktop Help

Data | Data Modeling Desktop

The Data|Data Modeling Desktop command starts the Data Modeling Desktop application so that you can create crosstab notebook formats with data from Quattro Pro.

The following dialog box options control how data is exchanged between your notebook and the Data Modeling Desktop:

Cell range to send to Data Modeling Desktop
Specifies the range of data to use when creating cross tabs.

Cell for data returned from Data Modeling Desktop
Specifies the target block for the results from Data Modeling Desktop.

Notebook name
Specifies the notebook the data originates from.

Data Exchange method
Specifies whether data will dynamically change when the database is updated.

See Also
Data Modeling Desktop Help

Data | Workgroup Desktop

The Data|Workgroup Desktop command displays the Workgroup Desktop SpeedBar and lets you share data with other members of work groups through a variety of messaging services. This command is only available in the Workgroup edition of Quattro Pro.

See Also
Workgroup Desktop Topics

Dialog Menu

The Dialog menu appears when you open a dialog window.

Connect	Connects the values of two different objects.
Links	Sets constants or passes properties to or from objects, and runs commands.
Align	Lines up, resizes, and controls spacing between dialog elements.
Order	Determines sequence of tabbing, returned values, and on-screen layering of dialog elements.
New SpeedBar	Creates a new SpeedBar.
Open SpeedBar	Displays an existing SpeedBar for editing.
Save SpeedBar	Saves the active SpeedBar under its current name.
Save SpeedBar As	Saves the active SpeedBar to a user-specified name and location.
Close SpeedBar	Closes the active SpeedBar.

Dialog | Connect

The Dialog|Connect command provides a quick way to link a dialog control to a notebook cell or another control. Connected controls get their initial value from the cell or element, and copy their final value into the cell or element when the dialog box closes.

Dialog|Connect creates link commands for the selected object. You can use Dialog|Links to remove or modify these links..

Source	Specifies the name of the control to add link commands to.
Target	Specifies the control name or cell address of the object to get the initial value from. When the dialog box is closed, the value of Source is sent back to this object.
Dynamic Connection	Sends changes from the source to the target whenever the source's value changes, rather than waiting until the user closes the dialog box.

See Also

Connecting Controls to Cells or Other Controls

Working with Link Commands

Creating a Dialog Box

Dialog | Links

The Dialog|Links command displays the Object Link dialog box where you can attach link commands to a control, to indicate what should happen when the user changes or selects the control. A link command can do many things; it can send a value to a cell in the notebook; it can run a macro; it can close the dialog box; it can change the zoom factor in the notebook.

Link commands will usually specify at least three things: an event, an action, and an object. The event indicates what occurrence should trigger the link command; the action indicates what should happen; the object is what should be acted upon. One control can have many link commands.

The following buttons in the Object Link dialog box create or remove link commands:

Add	creates a new link command below the selected link command.
Insert	creates a new link command above the selected link command.
Delete	removes the selected link command from the control.
Delete All	removes all link commands from the control.

Click Object Link Dialog Box Components for a graphic example of a control with a link command assigned to it. Each row of pick list buttons in the Object Link dialog box is a link command.

See Also

Creating a Dialog Box

Working with Link Commands

Making Menu Commands Act

Dialog | Align

The Dialog|Align commands line up dialog elements.

Left	Aligns elements along the left edge of the leftmost object selected.
Right	Aligns elements along the right edge of the rightmost object selected.
Horizontal Center	Centers elements between the left and right sides of the dialog box.
Top	Aligns elements along the top edge of the topmost object selected.
Bottom	Aligns elements along the bottom edge of the bottommost object selected.
Vertical Center	Centers elements between the top and bottom of the dialog box.
Horizontal Space	Horizontally spaces elements by the amount of pixels you specify.
Vertical Space	Vertically spaces elements by the amount of pixels you specify.
Resize to Same	Resizes all selected controls to the size of the first control selected.

See Also
Aligning Controls

Dialog | Order

Dialog|Order commands determine sequence of tabbing, returned values, and onscreen layering of dialog elements:

Order Controls	Sets the order in which initial settings are passed to a dialog box by {DODIALOG}.
Order From	Edits the order in which initial settings are passed to a dialog box by {DODIALOG}.
Order Tab Controls	Sets the order in which controls are activated by pressing the Tab key.
Order Tab From	Edits the order in which controls are activated by pressing the Tab key.
Bring Forward	Moves the selected object in front of an overlapping object.
Send Backward	Moves the selected object behind an overlapping object.
Bring to Front	Moves the selected object in front of all overlapping objects.
Send to Back	Moves the selected object behind all overlapping objects.

See Also

Moving and Sizing Controls

Setting a Control's Value

Working with Controls

Dialog | Order | Order Controls

The Dialog|Order|Order Controls command changes the order in which control settings appear in the settings block used by {DODIALOG}.

See Also

Setting a Control's Value

Working with Controls

Dialog | Order | Order From

The Dialog|Order|Order From command changes the order in which control settings appear in the settings block used by {DODIALOG}. Unlike Order Controls, you can use it to group specific controls within an existing order together.

See Also

Setting a Control's Value

Working with Controls

Dialog | Order | Order Tab Controls

The Dialog|Order|Order Tab Controls command determines the order in which the user can tab through controls in the dialog box. You can disable this for a particular control by right-clicking it, choosing the Properties command, and setting the Tab Stop property to No. By default all controls have a Tab Stop property of Yes. To specify the order in which Tab goes from control to control, hold down the Shift key and select each control in the order you want them to cycle. Then choose Dialog|Order|Order Tab Controls.

See Also

Setting Tab Order

Setting a Control's Value

Working with Controls

Dialog | Order | Order Tab From

The Dialog|Order|Order Tab From command provides a shortcut for setting the order in which several controls are activated by the Tab key. This command works in the same way as Dialog|Order|Order From. You can pull specific controls out of an ordering and group them together using Dialog|Order|Order Tabs From.

See Also

Setting Tab Order

Setting a Control's Value

Working with Controls

Dialog | Order | Bring Forward

The Dialog|Order|Bring Forward command moves the selected object in front of an overlapping object.

See Also
Moving and Sizing Controls
Working with Controls

Dialog | Order | Send Backward

The Dialog|Order|Send Backward command moves the selected object behind an overlapping object.

See Also

Moving and Sizing Controls

Working with Controls

Dialog | Order | Bring to Front

The Dialog|Order|Bring to Front command moves the selected object in front of all overlapping objects.

See Also
Moving and Sizing Controls
Working with Controls

Dialog | Order | Send to Back

The Dialog|Order|Send to Back command moves the selected object behind all overlapping objects.

See Also

Moving and Sizing Controls

Working with Controls

Dialog | New SpeedBar

The Dialog|New SpeedBar command displays an empty dialog window that you can use to build a SpeedBar. The dialog window is the SpeedBar in an editable form. Use the dialog window SpeedBar to add controls to the SpeedBar and use Dialog|Links to make SpeedBar controls perform various functions.

See Also

Dialog Window SpeedBar

Creating a SpeedBar

Dialog | Open SpeedBar

The Dialog|Open SpeedBar command displays an existing SpeedBar for editing. Use the dialog window SpeedBar to add controls to the SpeedBar and use Dialog|Links to make SpeedBar controls perform various functions.

See Also

Dialog Window SpeedBar

Creating a SpeedBar

Dialog | Save SpeedBar

The Dialog|Save SpeedBar command saves the active SpeedBar under its current name.

See Also
Dialog Window SpeedBar
Creating a SpeedBar

Dialog | Save SpeedBar As

The Dialog|Save SpeedBar As command saves the active SpeedBar under a specified name, in a specified location.

See Also
Dialog Window SpeedBar

Creating a SpeedBar

Dialog | Close SpeedBar

The Dialog|Close SpeedBar command closes the active SpeedBar. If you have not yet saved recent changes, Quattro Pro prompts you to save the SpeedBar file.

See Also

Dialog Window SpeedBar

Creating a SpeedBar

Draw Menu

The Draw menu is available only when a graph window is active. This menu lists the commands needed to arrange graphics and to exchange graphics files with other applications.

Group	Places two or more selected objects into a group, where they are treated as one object in subsequent operations.
Ungroup	Breaks a selected group of objects into its original components.
Bring Forward	Moves the selected object in front of an overlapping object.
Send Backward	Moves the selected object behind an overlapping object.
Bring to Front	Moves the selected object in front of all overlapping objects.
Send to Back	Moves the selected object behind all overlapping objects.
Align	Aligns selected objects in any of these ways: left, right, horizontal center, top, bottom, vertical center.
Import	Loads pictures in any of these formats: .BMP, .CGM, .CLP, .EPS, CIS .GIF, .PCX, .PIC, .TIF.
Export	Saves graphs and pictures in any of these formats: .BMP, .EPS, CIS .GIF, .PCX, .TIF.

Draw | Group

Draw|Group places selected objects into a single logical unit, or *group*. You group two or more objects when you want them to be treated as a single object in subsequent operations. For example, you might group all the objects in a complex drawing to ensure that no objects are left behind or shift position when you move them.

To group objects, select them and choose Draw|Group. To nest groups within other groups, select any combination of groups and individual objects, then choose Draw|Group. Once you group objects, you can move and resize them with the mouse. To return a group to its individual components, Ungroup them.

See Also
Grouping and Ungrouping Objects

Draw | Ungroup

Draw|Ungroup breaks a group into its original components. To use this command, select a group, then choose Draw|Ungroup. If the group had other groups nested within it, these groups remain intact. You must use Draw|Ungroup again to ungroup them.

See Also
Grouping and Ungrouping Objects

Draw | Bring Forward

The objects you create or import can overlap on invisible layers. Draw|Bring Forward, which is available only in a graph window, moves a selected object one layer closer to the front (or top layer) of the graph.

Draw|Send Backward moves the selected object behind an overlapping object.

See Also

Send Backward

Bring to Front

Send to Back

Arranging Object Layers

Draw | Send Backward

The objects you create or import can overlap on invisible layers. Draw|Send Backward, which is available only in a graph window, moves a selected object one layer closer to the back (or bottom layer) of the graph.

See Also

Bring Forward

Bring to Front

Send to Back

Arranging Object Layers

Draw | Bring to Front

The objects you create or import can overlap on invisible layers. Draw|Bring to Front, which is available only in a graph window, moves the selected object in front of all overlapping objects.

See Also

Bring Forward

Send Backward

Send to Back

Arranging Object Layers

Draw | Send to Back

The objects you create or import can overlap on invisible layers. Draw|Send to Back, which is available only in a graph window, moves the selected object behind all overlapping objects.

See Also

Bring Forward

Send Backward

Bring to Front

Arranging Object Layers

Draw | Align

You can line up the center or edges of two or more objects using Draw|Align. This command opens a menu containing these commands:

Left	Lines up the left sides of the selected figures.
Right	Lines up the right sides of the selected figures.
Horizontal Center	Lines up the centers of the objects as though they are beads on an invisible horizontal string.
Top	Lines up the top edges of the selected figures.
Bottom	Lines up the bottom edges of the selected figures.
Vertical Center	Lines up the centers of the objects as though they are beads on an invisible vertical string.

You can choose more than one of these commands in sequence. For example, to align the right sides and tops of a group of selected objects, choose Draw|Align|Right, and then choose Draw|Align|Top.

When you select an object group as one of the objects to be aligned, Quattro Pro aligns the group as if it were a single object.

You can also use the Snap to Grid option of the Grid property to align objects (see Grid Options for details).

See Also
Aligning Objects

Draw | Import

Draw|Import (or the Import button on the graph window SpeedBar) brings clip art or drawings from other programs (including non-Windows programs) into the active graph window. The options are the same ones available in standard file-handling dialog boxes (see File Handling Options for details).

You can import files in any of these formats:

.BMP	(Bitmap) The format used in Windows paint programs, like Paintbrush, and the Windows Control Panel for wallpaper.
.CGM	(Computer Graphics Metafile) The most common vector-based clip art format.
.CLP	A graphic format used by Quattro Pro for DOS.
.EPS	(Encapsulated PostScript) The file format universally understood by PostScript printers and image setters. The file contains PostScript code for high-quality output plus a bitmap of the image for lower-resolution screen display. (If the .EPS file does not contain a bitmap image, you can't import it into Quattro Pro.)
CIS .GIF	(Graphics Interchange Format) A bitmap format originating on CompuServe Information Service (CIS) and widely used on other electronic bulletin boards.
.PCX	A bitmap format supported by many programs, including Paintbrush and Quattro Pro for DOS.
.PIC	The graph format used in the DOS versions of 1-2-3.
.TIF	(Tag Image File Format) A bitmap file format often used for scanned or gray scale images. Because TIF files can be large, they are often compressed.

See Also

Importing Graphics

Cropping and Resizing Bitmaps

Modifying CGM and .CLP Graphics

Placing Text and Graphics on the Spreadsheet

Draw | Export

Draw|Export saves the entire contents of the active graph window in a different file format. The exported file can then be used in other programs or sent to a 35mm slide service bureau for processing. Many of the options are the same ones available in standard file-handling dialog boxes (see File Handling Options for details). Additional options are:

TIFF Compression	Choose the compression method that the target application for this file can accept: PackBits or None (no compression).
Bitmap Gray Scale	Converts the colors in the graph to levels of gray. This is useful you plan import this file into a program that can accept only gray scale TIF files.

Quattro Pro exports to these file formats:

.BMP	(Bitmap) The format used in Windows paint programs, like Paintbrush, and the Windows Control Panel for wallpaper.
.CGM	(Computer Graphics Metafile) The most common vector-based clip art format.
.EPS	(Encapsulated PostScript) The file format universally understood by PostScript printers and image setters. The file contains PostScript code for high-quality output plus a bitmap of the image for lower-resolution screen display. (If the .EPS file does not contain a bitmap image, you can't import it into Quattro Pro.)
CIS .GIF	(Graphics Interchange Format) A bitmap format originating on CompuServe Information Service (CIS) and widely used on other electronic bulletin boards.
.PCX	A bitmap format supported by many programs, including Paintbrush and Quattro Pro for DOS.
.TIF	(Tag Image File Format) A bitmap file format often used for scanned or gray scale images. Because TIF files can be large, they are often compressed.

If you're exporting a graph to send to a 35mm slide bureau, be sure to set the graph's Aspect Ratio to 35mm Slide.

See Also
Exporting Graphics

Edit Menu

Undo	Reverses the last "undoable" action you took.
Cut	Moves data from the notebook to the Windows Clipboard.
Copy	Places a copy of selected data in the Windows Clipboard.
Paste	Inserts data from the Windows Clipboard.
Clear	Erases the contents and resets the properties of a block of cells.
Clear Contents	Erases the contents but retains the properties of a block of cells.
Paste Link	Creates links between notebook cells, or creates DDE links to other Windows applications.
Paste Special	Pastes either the properties or values of a Quattro Pro block.
Paste Format	Pastes the Clipboard selection in the format you choose.
Select All	Selects all objects in a graph window or controls in a dialog window.
Goto	Selects the specified block or cell and displays it onscreen.
Search and Replace	Searches for values and optionally replaces them.
Define Style	Defines styles for spreadsheet data.
Insert Object	Runs a server application so you can quickly create an OLE object.

Edit | Undo

The Edit|Undo command lets you reverse most kinds of operations after you've carried them out. For example, if you make an entry in a cell, then change your mind and want to remove it, choose Edit|Undo immediately after making the entry. The entry will be removed and whatever was previously entered in the cell will be returned.

You can identify your last action by the wording of the Undo command. For example, after making a cell entry, the Undo command reads "Undo Entry." After changing the Zoom Factor property in the notebook Object Inspector, the Undo command reads "Undo Notebook Property."

If you change your mind again after using Undo and want to reinstate the first change, choose Edit|Redo. As with Undo, Redo specifies the type of operation available to be redone.

Some actions are undoable only if you enable Undo in the application Object Inspector. You can identify these actions because the Undo command is dimmed immediately after the action is taken. In all situations except where program speed and available memory is absolutely crucial, it's recommended that you keep Undo enabled.

See Also
Enabling the Undo Command

Edit | Cut

The Edit|Cut command removes the selected data or object and its properties from the notebook and places it on the Windows Clipboard. The Clipboard entry can then be pasted with Edit|Paste. The Clipboard holds only one entry at a time, so if you cut or copy another selection to the Clipboard, your previous entry is replaced.

See Also

Copying and Moving with the Windows Clipboard

Using Named Blocks

Copying Formulas

Edit | Copy

The Edit|Copy command makes a copy of the selected data or object and places the copy on the Windows Clipboard. The Clipboard entry can then be pasted with Edit|Paste. The Clipboard holds only one entry at a time, so if you cut or copy another selection to the Clipboard, your previous entry is replaced.

See Also

Copying and Moving with the Windows Clipboard

Using Named Blocks

Copying Formulas

Edit | Paste

The Edit|Paste command inserts data or an object from the Windows Clipboard into the selected location in the active notebook. You can continue to paste the same data to different locations until you cut or copy another selection to the Windows Clipboard; it replaces the previous Clipboard entry.

See Also

Copying and Moving with the Windows Clipboard

Using Named Blocks

Copying Formulas

Variations on Pasting

Edit | Clear

The Edit|Clear command erases the contents and resets the properties of a block of cells to current defaults. To erase contents but retain a block's properties, use the Edit|Clear Contents command.

See Also
Erasing Entries
Using Named Blocks

Edit | Clear Contents

The Edit|Clear Contents command erases the contents but retains the properties of a block of cells. To erase contents and reset a block's properties, use the Edit|Clear command.

See Also
Erasing Entries
Using Named Blocks

Edit | Paste Link

The Edit|Paste Link command creates a live data link to a participating DDE application that has data in the Clipboard, or creates a link between notebook cells. DDE links cause data changes from the other application to be constantly reflected in Quattro Pro. Notebook links can link cells in the same notebook or cells in separate notebooks.

See Also
Creating DDE Links
Creating Links

Edit | Paste Special

Edit|Paste Special pastes either the properties or data of a Quattro Pro block that has been placed on the Clipboard. The options are:

Properties	Pastes the properties from the copied selection.
Contents	Pastes the data from the copied selection. If you choose Formulas, the complete entry is pasted. If you choose Values Only, only the values resulting from the formulas are pasted. Choosing Values Only accomplishes the same thing as using Block|Values (see Converting Formulas to Values for details).
Transpose Rows and Columns	Switches the position of entries so that data listed in columns is placed in rows and vice versa. Be sure to allow enough room in the destination area for the changed size of the data. This option is equivalent to choosing Block|Transpose.
Avoid Pasting Blanks	Avoids erasing data in the destination selection that otherwise would be replaced by blanks pasted from the source.

See Also
Using Paste|Special

Edit | Paste Format

Edit|Paste Format pastes data or objects that are in another application's format from the Clipboard into Quattro Pro. Depending on the data type, the data is entered into cells, or as a floating object above the surface of the notebook page. Format options are:

BIFF3	Excel format data.
WK1	Lotus 1-2-3 versions 2.x format data.
Embedded OLE	A floating object containing data linked from another application.
Metafile	A graphic format that creates a floating picture object in a notebook.
Bitmap	A graphic format that creates a floating bitmap object in a notebook.
Reset	Clears all entries and restores defaults.
Text	Text data is entered as labels into cells, and numeric data is entered as values.
Linked OLE	A floating object displaying data linked from another application.
Link	A DDE link, which is the same as the link created with Edit|Paste Link.
WK3	Lotus 1-2-3 version 3.x format data.
Paradox Table	Paradox for Windows or Database Desktop data.
DIB	A Device Independent Bitmap format that creates a floating bitmap object.

Note: Not all of these formats are available in all situations. Depending on the data copied to the Clipboard and the location within Quattro Pro that's selected, only a subset of the list of formats may be available.

See Also
Choosing the Data Type to Paste

Edit | Select All

The Edit|Select All command selects all objects in a graph window or controls in a dialog window.

To select all cells on a notebook page, click the Select All button just to the left of the Column A label (above Row 1).

Edit | Goto

The Edit|Goto command lets you move to named blocks (to identify or modify them) or move quickly to distant parts of the notebook. The selector moves to whatever location you choose with the command. The options are:

Reference	The cell address or block coordinates to go to. You can include a page prefix and a notebook prefix, for example, `[BUDGET]D:Z36`.
Block Names	The named block to go to.
Pages	The page to go to.

See Also

Using the GoTo Command

Using Named Blocks

Edit | Search and Replace

Edit|Search and Replace finds or alters multiple cell entries. It searches for the specified value or string, then finds each match, which you can choose to replace with a new value. The options are:

Block(s)	The block or blocks to search. To search the entire page, this edit field should be blank.
Find	The group of characters to be found in labels, values, or formulas. These characters are called the find string.
Replace	The group of characters to substitute for characters found. This is called the replacement string.
Formula	Looks at the formulas as they appear in the input line.
Value	Looks at the results of the formulas, as they appear in the cells.
Condition	Treats the find string as a conditional expression. For example, a find string of B13>500 with Condition chosen looks for the first value that is greater than 500, starting with cell B13. To search starting from the active cell (which is usually the upper left cell), substitute a question mark for the cell address, for example, ? > 500. See Types of Operators for the list of logical operators that can be used in conditional find strings.
Columns First	Sets the search path. By default, this option is unchecked for searching across rows starting with row 1. When checked, searching occurs down columns starting with column 1.
Match Whole	Determines whether the find string must be all or part of a cell entry. By default, this option is unchecked for searching for partial as well as whole entries. For example, if the find string is `cat`, `catamaran` and `scatter` are found as well as `cat`.
Case Sensitive	Controls whether entries must exactly match the find string. By default, this option is unchecked for searching for strings regardless of capitalization. For example, if the find string is `HARPER`, `Harper`, `harper`, and `HaRpEr` are found as well as `HARPER`.
Next	Begins or resumes a forward search without replacing found entries.
Previous	Begins or resumes a backward search without replacing found entries.
Replace	Lets you decide on an individual basis whether to replace each string found.
Replace All	Replaces all found strings without stopping.
Reset	Removes any entries in the dialog box and reinstates the defaults.
Close	Cancels the find operation without making a change.

If you're using a conditional expression, you can only find values; they aren't replaced when found. If you're searching for a label with Match Whole checked, you must either include the label prefix or set Look In to Value.

See Also

Search & Replace Procedure

Creating Formulas

Searching for Records

Edit | Define Style

Edit|Define Style lets you modify Quattro Pro's predefined styles, or create new ones. The options are:

Define Style For	The style to be revised or created.
Included Properties	Adds settings for the checked properties to the new style definition. Click the property name button for each checked property to choose the particular setting you want.
Delete	Deletes the style selected in the Define Style For option. Cells to which you had applied the deleted style revert to Normal style.
Merge	Opens a dialog box in which you specify to copy property settings from a cell or another style.

The Merge Style dialog box contains these options:

Merge From	Specifies whether to copy from an existing style or a cell.
Select Cell/Style	Enter the cell address, or style name, as appropriate.

Note: Only checked properties are included in the style you define. The unchecked properties will have no effect in the cells to which you apply this new style. This lets you create partial styles that affect only a few properties at a time and apply them in combination with properties already existing in cells.

To use the new style, select the cells to which you want to apply it and select the name from the Style list in the SpeedBar.

If you redefine a style that is already in use in the notebook (such as Normal, which is applied to all cells by default), all cells using that style change to reflect the new property settings. Cells in which you set individual properties with the block Object Inspector don't change. This is because styles act as default settings that are overridden by block property settings.

To copy a style to another notebook, copy a cell formatted in that style, and paste it into the new notebook. The style name then appears in the Style list for that notebook.

The predefined styles are described in Using Styles.

See Also

Defining Styles

Active Block Properties

Edit | Insert Object

The Edit|Insert Object command inserts an embedded OLE object into the active page.

When you choose Edit|Insert Object, Quattro Pro displays the list of available OLE server applications from the Windows 3.1 registration database. When you choose a server, it opens a window on top of Quattro Pro where you can create an object. When you finish work on the object and choose File|Exit or Update, the object is inserted at the position of the active cell in Quattro Pro. The size of the object is determined by the server application.

Like other floating objects, OLE objects can be resized by selecting one of the object's handles. While the handle is selected, the current size is displayed in the input line as a percentage of the size specified by the OLE server.

See Also

Edit|Paste Format

Inserting OLE Objects

Changing OLE Object Properties

Layering Floating Objects

File Menu

New	Opens a new, blank notebook window.
Open	Opens a new window and loads a file into it.
Close	Closes the active window along with its notebook.
Save	Stores the current notebook under the current name.
Save As	Stores the current notebook under the name you give it.
Retrieve	Loads a stored notebook in the current window.
Save All	Saves all open files.
Close All	Closes all open files.
Print Preview	Displays the current output selection onscreen for review.
Page Setup	Specifies the page format of output.
Print	Sends output to the specified destination.
Printer Setup	Specifies the print destination (printer, file, or screen).
Named Settings	Saves print settings to a file for later use.
Workspace	Saves or loads a group of notebooks.
Exit	Closes all open windows and exits Quattro Pro.

If you're working with Quattro Pro through an embedded OLE object in a client application, some of the File menu options change. For details, see OLE and DDE Overview.

See Also
File-Handling Options

File | New

File|New creates a new notebook window without putting away the active notebook. Quattro Pro opens a blank window, covering up existing windows. It names the window NOTEBK#.WB1, where # is the number of windows you've opened since starting Quattro Pro. You can change the name when you save the file.

You can have multiple windows open at a time, with different notebooks in each. Each new window overlays other open windows. You can reselect an open window by clicking any visible part of it, or by selecting it from the list at the bottom of the Window menu.

See Also

Using Windows

File | Open

The File|Open command loads an existing notebook file into Quattro Pro. The new window overlays other open notebook windows.

If the file has a password, a dialog box appears with space for entering it. If the password is incorrect, Quattro Pro displays an error message and cancels the Open command.

See File-Handling Options for further information on File|Open dialog box options.

See Also

Opening a Window and a File

Using Windows

Translating Files

File | Close

The File|Close command removes the active Quattro Pro notebook window and its associated graph and dialog windows and views from the desktop. This command is especially useful when system memory resources are strained by keeping large files open.

If a notebook has unsaved changes, a dialog box appears asking if you want to save your changes. If you choose Yes, the File|Save As dialog box appears; choosing No closes the file without saving it.

To close all open notebook windows, choose File|Close All. To close all open notebook windows and exit Quattro Pro at the same time, choose File|Exit.

See Also

Using Windows

Saving Files

File|Close All

File|Exit

File | Save

The File|Save command saves the notebook under its current name. If you use File|Save on a notebook that has not yet been saved, Quattro Pro displays the File|Save As dialog box.

When Quattro Pro saves the notebook, it stores the data along with all its properties, named blocks, graphs, slide shows, and dialog boxes you've created. Application properties aren't stored with individual notebooks; a notebook assumes the current settings for these properties the next time it is loaded.

See Also

Saving Files

File-Handling Options

Using Windows

File|Workspace Command

File | Save As

The File|Save As command saves the notebook under a new name you specify, and in the drive and directory of your choice. Use File|Save As when you want to change a file name or save the file to a different location.

If the file name you enter already exists, Quattro Pro warns you and gives you these options:

Replace	Overwrites the existing file.
Backup	Makes a copy of the file that exists on disk, giving it a .BAK file-name extension, and overwrites the original. To load the backup file later, include the .BAK extension with that name. This .BAK file contains your data as you previously saved it.
Cancel	Stops the command and returns you to the notebook.

When Quattro Pro saves the notebook, it stores the data along with all its properties, named blocks, graphs, slide shows, and dialog boxes you've created.

After you name a notebook, you can store changes to it with File|Save. Quattro Pro assumes you want to save the file under the same name. If you use File|Save on a notebook that has never been saved, Quattro Pro displays the File|Save As dialog box.

To save a notebook for use with another program, such as Paradox, Reflex, 1-2-3, or dBASE, include the program's data file extension when you save the notebook (see Translating Files for more information).

File names can be up to eight characters long and can consist of both letters and numbers. You can enter file names using either upper- or lowercase letters. You can't use spaces in a file name. You can use the underscore character to simulate spaces; for example, 92_SALES.

See File-Handling Options for further information on File|Save As dialog box options.

Caution: If you assign a name longer than eight characters, Quattro Pro shortens it to the first eight characters.

Unless you're saving a file for use with another program, it's best if you don't include an extension with the file name. Quattro Pro then adds the default extension (initially .WB1), so you won't have to include it when you load the file later.

See Also

Saving Files

Password Protecting a File

File | Retrieve

The File|Retrieve command loads a notebook into the active window. This removes any data from the active window, then fills it with the specified notebook.

To retrieve a file created with a different program, include the file-name extension when you retrieve it.

See File-Handling Options for further information on File|Retrieve dialog box options.

See Also

Retrieving a File into a Window

Using Windows

Translating Files

File | Save All

File|Save All saves all open notebooks. If a notebook has not yet been saved, the Save File dialog box appears so you can enter a file name.

If the file name you enter already exists, Quattro Pro warns you and gives you these options:

Replace	Overwrites the existing file.
Backup	Makes a copy of the file that exists on disk, giving it a .BAK file-name extension, and overwrites the original. To load the backup file later, include the .BAK extension with that name. This .BAK file contains your data as you previously saved it.
Cancel	Stops the command and returns you to the notebook.

See Also

Saving Files

File-Handling Options

Using Windows

File | Close All

The File|Close All command removes every Quattro Pro window and its contents from the desktop.

If a notebook has unsaved changes, a dialog box appears asking if you want to save your changes. If you choose Yes, the File|Save As dialog box appears; choosing No closes the file without saving it.

To close all open notebook windows and exit Quattro Pro at the same time, choose File|Exit.

See Also
Using Windows
Saving Files

File | Print Preview

The File|Print Preview command gives you an onscreen preview of how a document will appear in print.

The current zoom level displays to the right of Zoom. When previewing, left-clicking the page zooms in a level, increasing detail; right-clicking zooms out a level, decreasing detail. You can use scroll bars to adjust the section of the page viewed.

The SpeedBar provides a variety of tools for controlling a document while previewing. For more information on a tool, point to it, press and hold down Control, then right-click the mouse.

See Print Preview Window Keys for a list of keys you can use while previewing.

See Also
Printing Notebooks
Printing Graphs

File | Page Setup

The File|Page Setup command controls printed margins, paper type and orientation, and lets you enter and position headers and footers. The options are:

Header/Footer	Enter text for headers or footers in the edit fields. See Entering Headers and Footers for details.
Margins	Controls the amount of space between the edge of the page and the document text. See Setting Margins and Paper Type for details.
Paper Type	The type of paper loaded into your printer.
Header Font	The typeface, point size and type style for header or footer text. See Header Font for details.
Break Pages	Specifies whether to start a new printed page at each soft page break. See Inserting Page Breaks for details.
Print To Fit	Shrinks the print block to fit on as few pages as possible. See Changing Print Size for details.
Center Blocks	Centers each contiguous block of the print block between the left and right margins of the printed page. (By default, the first column of each contiguous block prints flush against the left margin.)
Scaling	The percentage to increase or decrease the size of notebook data on the printed page. The margins (except for header and footer margins) don't change.
Print Orientation	Choose Portrait to print data vertically, on individual pages; choose Landscape to print horizontally, on individual pages. When switching orientation, margins switch as well; for example, the top margin stays with the top of the notebook data printed, along with the header and top heading (if any).
Reset Defaults	Resets the dialog box to its default settings, without clearing named settings.

See Also

Named Settings

Printing Notebooks

Printing Graphs

File | Page Setup | Header Font

Specifies the typeface, size, and type style for both the header and footer.

Typeface	Determines the typeface for the header/footer. Some typefaces, such as Symbol and Zapf Dingbats, are not appropriate for headers and footers as they consist of symbols rather than alphanumeric characters.
Point Size	Determines the size of the header/footer in points. Typical font sizes for text in a letter are 10 or 12 point. Headers and footers are usually not much larger than this, and almost always smaller than titles.
Options	Determines the style of the header/footer font. If you don't check any of the options, the header/footer is in the normal font.

See Also
Entering Headers and Footers

File | Print

The File|Print command prints out parts of a notebook. The options are:

Print Blocks(s)	Indicates the block to be printed. If a cell's contents spill over into adjacent cells onscreen, include the spillover cells in the selection; otherwise, only part of the entry will print.
All pages	Specifies that all pages in the document be printed.
From...To	Specifies the beginning and ending pages in the document to print.
Copies	Specifies the number of copies to print.
Preview	Opens the same Print Preview window opened by File|Print Preview, which lets you examine the document's appearance before you actually print it. See Print Preview for details.
Options	Opens a dialog box for controlling placement of headings, gridlines, row and column borders, spacing between blocks or pages, or whether cell contents are printed. See File	Print	Options for details.
Print	Sends the document to the printer.

See Also

Defining a Print Block

Printing a Spreadsheet

File | Print | Graph

The Graph Print dialog box appears when you choose File|Print if a graph window is active or if a graph or slide show icon in the Graphs page is selected. The options are:

Copies	Specifies the number of copies to print.
Print	Sends the document to the printer.
Preview	Opens the same Print Preview window opened by File|Print Preview, which lets you examine the graph's appearance before you actually print it. See Print Preview for details.

See Also

Printing Graphs

Creating a Floating Graph

Creating a Graph in a Window

Creating a Text Graph

Creating a Slide Show

File | Print | Options

The Options dialog box of File|Print controls inclusion of headings, borders, gridlines, and cell contents, and specifies spacing between blocks in the printed document. The options are:

Headings	Adds the cell entries you specify as headings to print at the top or left of each printed page. See Adding Headings for details.
Cell Formulas	Prints each cell's address and contents instead of its calculated results. When Cell Formulas is checked, the Center Blocks option (in the Print|Page Setup dialog box) and the Top and Left Heading, Gridlines, and Row/Column Borders options are disabled.
Gridlines	Includes the spreadsheet grid in the printed document.
Row/Column Borders	Includes row and column borders in the printed document.
Print Between Blocks	Choose Lines and type an entry in the edit field to separate subblocks of noncontiguous blocks with blank lines. Choose Page Advance to separate them with page breaks.
Print Between 3D Pages	Choose Lines and type an entry in the edit field to separate pages of 3-D blocks with blank lines. Choose Page Advance to separate them with page breaks.
Reset Defaults	Resets the dialog box to its default settings, without clearing named settings.

See Also
Entering Headers and Footers

File | Printer Setup

The File|Printer Setup command lets you choose the print destination. The options are:

Printer and Port	Specifies which of the available printers to use. Before printing from Quattro Pro, use the Windows Control Panel to specify the printers available. These printers will then be available to all Windows applications.
Redirect to File	Creates a binary file to store the printed document instead of sending the document to a printer. You can type the file name or choose it using the Browse option.
Browse	Opens a dialog box from which you can choose the destination for the binary file.
Set Up	Opens a dialog box for specifying default print settings. See below for details.

By choosing the Setup option, you can specify default print settings such as:

 paper type (wide, letter, envelope)

 paper source on the printer (manual, bin 1, bin 2)

 print orientation (landscape, portrait)

 output size using a percentage (50%, 200%, and so on)

 pen colors in a plotter

 number of copies

 fonts available in the printer (cartridge fonts, soft fonts, internal fonts)

 output resolution (300 dpi, 150 dpi)

 color or black-and-white printing

 the amount of memory in the printer

Since many of these settings are printer-specific, they won't all be available for your printer. Consult your Windows documentation and printer manual for more information on these and additional settings.

Note: Changes made to printer defaults affect printing in all Windows applications. You can use File| Page Setup and the Options dialog box of File|Print to set many of these options (output size, paper type, number of copies) for only Quattro Pro.

See Also
Named Settings

Printing Notebooks
Printing Graphs

File | Named Settings

The File|Named Settings command lets you store the current print settings under a name so that you can easily reuse them. All settings in the File|Print and File|Page Setup dialog boxes are saved under the name specified. The options are:

Named Settings	After you create a named setting, it appears in this list box. Choosing a named setting from this list replaces the current print settings with those stored under the name.
Create	Lets you create a named setting from the current print settings.
Update	Replaces the settings stored under the selected name with the current print settings.
Delete	Deletes the selected named setting.

Since named settings are saved with the notebook, use File|Save to save your file after changing named settings.

See Also

Print Settings

Printing Notebooks

Printing Graphs

File | Workspace

The File|Workspace command saves or restores workspaces. A workspace is the arrangement of windows onscreen. It includes the position and size of all notebook windows and the files contained in each window. The positions of graph and dialog windows aren't saved as part of a workspace.

See Also

File|Workspace|Save Command

File|Workspace|Restore Command

Using Windows

Saving Files

File | Workspace | Save

The File|Workspace|Save command saves workspaces. A workspace is the arrangement of windows onscreen. It includes the position and size of all notebook windows and the files contained in each window. The positions of graph and dialog windows aren't saved as part of a workspace.

When saving a workspace, don't include a file-name extension; Quattro Pro includes the .WBS extension for workspace files.

Caution: Saving a workspace doesn't save the contents of the files within it; use Save, Save As, or Save All for this. Also, if you use File|Save As, you must use File|Workspace Save afterward so that Quattro Pro can load the correct file the next time you retrieve the workspace.

See File-Handling Options for further information on File| Workspace|Save dialog box options.

See Also

File|Workspace|Restore Command

Using Windows

Saving Files

File | Workspace | Restore

The File|Workspace|Restore command restores workspaces. A workspace is the arrangement of windows onscreen. It includes the position and size of all notebook windows and the files contained in each window. The positions of graph and dialog windows aren't saved as part of a workspace.

When restoring a workspace Quattro Pro overlays any existing windows with the windows stored in the workspace file, then retrieves the appropriate file for each.

Quattro Pro always retrieves the latest saved version of files when you retrieve a workspace. If you leave the workspace and save a file included in the workspace, Quattro Pro retrieves the updated version of the file when you choose File|Workspace|Restore.

See File-Handling Options for further information on File|Workspace|Restore dialog box options.

See Also

File|Workspace|Save Command

Using Windows

Saving Files

File | Exit

The File|Exit command closes all windows and exits Quattro Pro. If a notebook has unsaved changes, a dialog box appears asking if you want to save your changes. If you choose Yes, the Save File dialog box appears; choosing No closes the file without saving it.

See Also
Saving Files

Graph Menu

The Graph menu, which is available in the notebook window, in the graph window, and on the Graphs page, lists most of the commands used with graphs.

Type	Changes the graph type (line, pie, 3-D ribbon, and so on).
Series	Redefines the spreadsheet data plotted in the graph.
Titles	Adds a title, a subtitle, and axis titles to the graph.
New	Creates a graph from the selected data. If no data is selected, Quattro Pro creates a text graph.
Edit	Brings a graph into a graph window, ready for editing.
Insert	Places an existing graph as a floating graph on a spreadsheet page.
Delete	Erases a graph from the active notebook.
Copy	Copies a graph, or selected attributes of a graph, to a new or existing graph within the same notebook. You can copy the graph style (which includes all the graph properties), the graph data, the annotation objects, or any combination of the three.
View	Displays a full-screen view of a graph. Graph buttons and graph button macros become active, so you can test them. You can also select a group of graphs, either on the spreadsheet page or the Graphs page, then choose Graph|View to preview a slide show.
Slide Show	Starts a selected slide show from any window.

You must select a graph before you can use the Graph|Type, Graph|Series, or Graph|Titles command (see Selecting Graphs for Editing for details).

See Also
Draw Menu

Graph | Type

The Graph|Type command changes the active graph to a different graph type. The options are:

2-D	Choose from the Bar, Variance, Stacked Bar, High Low, Line, XY, Area, Column, Pie, 100% Stacked Bar, Stacked Bar Comparison, 100% Stacked Bar Comparison, Doughnut, and Radar graph types.
3-D	Choose from the 3-D Bar, 3-D Stacked Bar, 2.5-D Bar, 3-D Step, 3-D Unstacked Area, 3-D Ribbon, 3-D Area, 3-D Column, 3-D Pie, 3-D Surface, 3-D Contour, 3-D Shaded Surface, 3-D 100% Stacked Bar, and 3-D Doughnut graph types.
Rotate	Choose from the Rotated 2-D Bar, Rotated 3-D Bar, Rotated 2.5-D Bar, Rotated Area, Rotated Line, Rotated 2-D Stacked Bar, Rotated 2-D 100% Stacked Bar, Rotated 2-D Comparison, Rotated 2-D 100% Comparison, Rotated 3-D Stacked Bar, and Rotated 3-D 100% Stacked Bar graph types.
Combo	Choose from the Line-Bar, Multiple Columns, Area Bar, Multiple 3-D Columns, High Low-Bar, Multiple Pies, Multiple Bar, and Multiple 3-D Pies graph types.
Text	Choosing a text graph creates a blank graph that serves as the foundation for text and pictures you add using tools in the graph window SpeedBar.

See Also

2-D Graphs
3-D Graphs
Rotated Graphs
Combination Graphs
Text Graphs
Multiple Graphs
Changing the Graph Type
Choosing a Graph Type

Graph | Series

The Graph|Series command lets you change how data is plotted in the graph. To use this command, select the graph, then choose Graph|Series. The dialog box that appears has these features:

Series buttons	activate Point mode, which lets you define a series by pointing to a block on a spreadsheet page. Series buttons are labeled X-axis, Legend, 1st, 2nd, and so on, and appear on the left side of the dialog box.
Edit fields	are located to the right of the Series buttons, and display the block coordinates of the series. Double-clicking an edit field, also activates Point mode. You can define a series block by pointing to the block, or by typing block coordinates directly into the edit field.
Add button	adds a series after the selected series.
Delete button	deletes the selected series.
Reverse Series	plots the last series first, then moves backward through the series order. This is useful in 3-D bar and unstacked area graphs, for example, when a series with high values is plotted at the front of the graph, obscuring other series.
Row/Column Swap	plots columns as series when Quattro Pro assigns series by rows, and plots rows as series when Quattro Pro would plot columns. Row/column swap also puts x-axis series labels in the legend, and places legend series labels along the x-axis.

See Also

Assigning Series by Pointing

Adding a Series

Graphing Grouped or Linked Data

Deleting a Series

Graph | Titles

The Graph|Titles command lets you create titles in standard locations in graphs. The options are:

Main Title	Available in all graph types.
Subtitle	Available in all graph types except text graphs.
X-Axis Title	Available in all graphs with axes.
Y1-Axis Title	(The primary y-axis) Available in all graphs with axes.
Y2-Axis Title	(The secondary y-axis) Available in all graphs with axes except 3-D graphs.

TIp If your y-axis scale labels are greater than 1000, you can use the Show Units option to simplify the scale and add an appropriate title--"(thousands)" for example--between the y-axis title and the axis itself. Show Units is one of the Scale property options in the y-axis Object Inspector.

You can change the font, color, or other properties of the Graph Title, Graph Subtitle, and the Graph Title Box, through their Object Inspectors. You can edit the main title and subtitle text directly on the graph, or change the title or subtitle in the edit field of either the Graph|Title dialog box or the Object Inspector.

You can edit axis title text in the Graph|Titles dialog box, or in the Axis Title Object Inspector. This Object Inspector also lets you change the font, color, and other properties of axis titles.

Graph | New

The Graph|New command creates a graph in its own window. Graph|New is available from a spreadsheet page, from the Graphs page, and when a graph window is active. The options are:

Graph Name	The name assigned to the graph. You can use the default name, or change it to a more descriptive one. The graph name appears on the Graphs page, and you can choose it from the dialog boxes of other Graph menu commands (such as Graph|Edit).
Series buttons	Activate Point mode, which lets you define a series by pointing to a block on a spreadsheet page. Series buttons are labeled X-axis, Legend, 1st, 2nd, and so on, and appear on the left side of the dialog box.
Edit fields	Display the block coordinates of the series. Double-clicking an edit field also activates Point mode. You can define a series block by pointing to the block, or by typing block coordinates directly into the edit field.
Add button	Adds a series after the selected series.
Delete button	Deletes the selected series.
Reverse Series	Plots the last series first, then moves backward through the series order. This is useful in 3-D bar and unstacked area graphs, for example, when a series with high values is plotted at the front of the graph, obscuring other series.
Row/Column Swap	Plots columns as series when Quattro Pro assigns series by rows, and plots rows as series when Quattro Pro would plot columns. Row/column swap also puts x-axis series labels in the legend, and places legend series labels along the x-axis.

See Also
Creating a Graph in a Window

Graph | Edit

The Graph|Edit command places a graph into a graph window for editing. The option is:

Select Graph Displays the list of graphs in the active notebook.

The graph window SpeedBar displays tools and a palette for customizing the graph with drawings and text. Object Inspectors for graph elements are also available only in the graph window. To display an Object Inspector, right-click the part of the graph you want to change and choose the Properties command in the SpeedMenu that appears.

Graph|Edit is available from a spreadsheet page, from the Graphs page, and when a graph window is active.

See Also

Customizing Graph Properties

Enhancing Graphs

Graph | Insert

Graph|Insert lets you take any graph in the active notebook and place it as a floating graph on a spreadsheet page. The command is available from any spreadsheet page. The option is:

Select Graph	Displays the list of graphs in the active notebook. After you choose a graph and choose OK, the mouse pointer changes to a miniature graph. Click the upper left corner of the area where you want the graph to appear, or drag the mouse to specify the size and placement of the graph.

You can insert the same graph in more than one place in the notebook. Any time you make changes to a graph, all floating graphs associated with that graph reflect the changes.

To replace a floating graph with a different graph, see Source Graph.

If the graph appears to have too much background area, change the Aspect Ratio to Floating Graph.

Graph | Delete

Graph|Delete lets you select and erase any graph in the active notebook. This command is available from a spreadsheet page, a graph window, and the Graphs page. The option is:

Select Graph	Displays the list of graphs in the active notebook. After you select a graph and choose OK, the corresponding graph icon is removed from the Graphs page, and all floating graphs based on that graph are deleted.

As an alternative, you can select a graph icon on the Graphs page, then choose Edit|Cut or press Del to delete the graph.

If you select a floating graph, then choose Edit|Cut, the graph is erased from the spreadsheet page, but it is *not* deleted from the notebook. You must use Graph|Delete, or cut the graph icon on the Graphs page, to delete a graph from the notebook.

Graph | Copy

The Graph|Copy command lets you copy the entire graph, or just selected characteristics of the graph. The options are:

From	The source graph to be copied.
To	The destination graph to receive the copy (either an existing graph or a new one).
Style	Copies all properties that affect the appearance of the graph such as text fonts, line styles, and fill colors. Style also copies the graph type, and series overrides. For example, if you change the background color of the source graph, choose a dotted line style for the y-axis grid, remove the box around the legend, and plot the fourth series on a secondary y-axis, all these changes are transferred to the destination graph when you copy the style.
Data	Builds the target graph from the same data as the source graph. When you choose this option, Quattro Pro copies a reference to the data used in the source graph. (Data is not physically copied to a different location in the notebook.) Any changes you subsequently make to the data affect the source graph and all copies. Data also copies x-axis and legend series, legend label overrides, label series, and graph title, subtitle, and axis title text (but not the style of this text).
Annotate Objects	Copies all text and drawings you created for the source graph using the drawing tools on the graph window SpeedBar. Annotate Objects also includes any graphics you imported for the source graph.

Caution: To preserve annotation objects already existing in the destination graph, uncheck Annotate Objects. Annotate Objects erases all annotation items in the destination graph, even when there are none to copy from the source graph.

The options you choose can produce quite different results:

If you check Style, Data, and Annotate Objects, Quattro Pro builds an exact copy of the original graph. (You can copy the source graph through the Clipboard to get the same results.)

If you check Style and Annotate, but not Data, Quattro Pro builds a graph that has the same graph type and properties as the source graph, but uses the data of the destination graph. (The source graph serves as a style template for the destination graph).

If you copy to a new graph, Quattro Pro automatically creates this graph using data from the source graph (whether or not you checked the Data option in the dialog box). If you don't copy Style to a new graph, you create a 2-D bar graph.

See Also

Customizing Graph Properties

Enhancing Graphs

Graph | View

Graph|View gives you a full-screen view of selected graphs as they would appear in a slide show. (On the spreadsheet page, select a floating graph by clicking it. On the Graphs page, select a graph by clicking its icon. A graph displayed in the active graph window is automatically selected.) Graph buttons and graph button macros are active, so you can test them.

Press any key or click the mouse to go to the next graph. When the last graph is displayed, a mouse or key click returns you to the active window.

See Also
Creating a Graph Button

Graph | Slide Show

Graph|Slide Show runs any slide show in the active notebook. The option is:

Select Graph Displays the list of slide shows in the active notebook.

Graph|Slide Show is available from any window.

See Also

Creating a Slide Show

Help Menu

Contents	Lists available Help topics; also accessible with the Contents button in the Help window.
Search	Displays the Search dialog box; also accessible with the Search button in the Help window.
Experts	Displays available Experts. Each Expert offers an alternative way to perform a task.
Interactive Tutors	Displays available Interactive Tutors. Each tutor teaches you how to perform a task using your own "live" data.
About Quattro Pro	Gives the copyright date and system memory use.

Property Menu Commands

The Property menu commands change depending on which Quattro Pro window is active. When a notebook window is active, the Property menu includes the commands listed below. The graph and dialog windows have different commands (see Graph Window Property Menu and Dialog Window Property Menu.)

Notebook Window Property Menu Commands

Current Object	Displays the Object Inspector for the currently selected object in the active window.
Application	Displays the application Object Inspector.
Active Notebook	Displays the Object Inspector for the currently selected notebook.
Active Page	Displays the Object Inspector for the currently selected page.

See Also

Object Inspector Menus

Graph Window Property Menu

The Property menu commands change depending on which Quattro Pro window is active. When the graph window is active, the Property menu includes the following commands.

Current Object	Displays the Object Inspector for the currently selected object in the active window.
Application	Displays the application Object Inspector.
Graph Window	Displays the Object Inspector for the currently selected graph window
Graph Setup	Displays the graph setup Object Inspector
Graph Pane	Displays the graph pane Object Inspector
Legend	Displays the legend box Object Inspector
X Axis	Displays the non-scaling axis Object Inspector
Y Axis	Displays the scaling axis Object Inspector
Series	Lets you select a series from the active graph

See Also

Object Inspector

Notebook Window Property Menu

Dialog Window Property Menu

Property|Series

The Property|Series menu command lets you select a series in the active graph. After you select a series, the series Object Inspector for the current graph type appears.

See Also

Object Inspector

Graph Window Property Menu

Dialog Window Property Menu

The Property menu commands change depending on which Quattro Pro window is active. When the dialog window is active, the Property menu includes the following commands.

Current Object	Displays the Object Inspector for the currently selected object in the active window.
Application	Displays the application Object Inspector.
Dialog	Displays the Object Inspector for the currently selected Dialog window.

See Also

Object Inspector

Notebook Window Property Menu

Graph Window Property Menu

SpeedBar Menu

When you choose Tools|SpeedBar Designer, the SpeedBar menu appears in the menu bar. It offers the following options; some duplicate Designer SpeedBar buttons while others are unique:

Dock	Closes the active SpeedBar in the SpeedBar Designer and appends it below the last SpeedBar at the top of the Quattro Pro window, ready to use.
Assign Macro	Attaches the specified macro to the active SpeedBar button.
New	Opens a blank SpeedBar window.
Open	Offers a list of custom SpeedBars for editing.
Close	Closes the active SpeedBar in Edit mode.
Save	Saves the active SpeedBar under the same name.
Save As	Saves the active SpeedBar under another name.
Save All	Saves all SpeedBars currently open for editing.
Close All	Closes all SpeedBars currently open for editing.
Tile	Arranges all open SpeedBars so none overlap.

See Also

SpeedBar Designer Guidelines

Using Custom SpeedBars

SpeedBar | Dock

Closes the active SpeedBar in the SpeedBar Designer and appends it below the last SpeedBar at the top of the Quattro Pro window, ready to use.

See Also

SpeedBar Designer Guidelines

Using Custom SpeedBars

SpeedBar Menu

SpeedBar | Assign Macro

Attaches the specified macro to the active SpeedBar button. The Assign Macro dialog box has these edit fields and lists:

Location	Enter the address of the cell containing the macro you want to run, or use the Macros/Namedblocks option to choose the macro. Use full linking syntax ([LIBRARY]A:C2) if the cell address isn't in the active notebook (linking isn't needed when entering a block name).
Macro Library	If the macro is stored in a notebook designated as a macro library, choose the notebook name here. Macros/Namedblocks changes to display the macros available from the notebook.
Macros/Namedblocks	You can choose the name of a macro to run from this list. Use Macro Library to specify the list displayed.

See Also

SpeedBar Designer Guidelines

Assigning Macros

SpeedBar Menu

SpeedBar | New

Opens a blank SpeedBar window.

See Also
SpeedBar Designer Guidelines
SpeedBar Menu

SpeedBar | Open

Offers a list of custom SpeedBars for editing.

See Also

File-Handling Options

SpeedBar Designer Guidelines

Using Custom SpeedBars

SpeedBar Menu

SpeedBar | Close

Closes the active SpeedBar in Edit mode.

See Also
SpeedBar Designer Guidelines
Exiting SpeedBar Designer

SpeedBar | Save

Saves the active SpeedBar under the same name.

See Also

Saving Custom SpeedBars

SpeedBar Designer Guidelines

File-Handling Options

SpeedBar Menu

SpeedBar | Save As

Saves the active SpeedBar under another name.

See Also

Saving Custom SpeedBars

SpeedBar Designer Guidelines

File-Handling Options

SpeedBar Menu

SpeedBar | Save All

Saves all SpeedBars currently open for editing.

See Also

Saving Custom SpeedBars

SpeedBar Designer Guidelines

File-Handling Options

SpeedBar Menu

SpeedBar | Close All

Closes all SpeedBars currently open for editing.

See Also

Closing SpeedBars

SpeedBar Designer Guidelines

File-Handling Options

SpeedBar Menu

SpeedBar | Tile

Arranges all open SpeedBars so none overlap.

See Also

SpeedBar Designer Guidelines

Using Button Palettes

SpeedBar Menu

Tools Menu

Macro	Provides commands for creating, running, and debugging macros.
Spell Check	Checks spelling in notebooks and graphs (Workgroup edition only).
Scenario Manager	Allows naming and saving of notebook variations.
Consolidator	Merges blocks, altering the data with functions you select.
Define Group	Defines page groups for simultaneous operations on several pages.
Combine	Merges two notebook files into one notebook.
Extract	Copies part of a notebook into a separate file.
Import	Imports plain text or comma and quote delimited files.
Update Links	Opens, updates, changes, or deletes notebook links.
Analysis Tools	Displays a list of advanced statistical, financial, and engineering analysis tools.
Optimizer	Performs linear and nonlinear programming; solves sets of equations or inequalities to find optimum solutions.
Solve For	Calculates a formula backwards, starting with the desired result and solving for a variable that produces that result.
Advanced Math	Accesses regression analysis and matrix math tools.
SpeedBar Designer	Allows you to create your own custom SpeedBars.
UI Builder	Provides tools for customizing Quattro Pro's user interface (UI).

Tools | Macro

A macro is a stored series of keystrokes, special keys, and commands that can be played back. The Tools|Macro command displays a submenu of commands you can use with macros:

Execute	Plays back the macro you specify.
Record	Puts Quattro Pro in Macro Record mode.
Options	Sets the recording type and how addresses are recorded.
Debugger	Enters Debug mode, so you can work out problems in your macros.

See Also

Macro Topics

Special Macro Commands

Tools | Macro | Execute

Tools|Macro|Execute lets you run a macro. The macro can be stored in the active notebook, or in an open notebook designated as a macro library.

Options

Location	Enter the address of the cell containing the macro you want to run, or use the Macros/Namedblocks option to choose the macro. Use full linking syntax ([LIBRARY]A:C2) if the cell address isn't in the active notebook (linking isn't needed when entering a block name).
Macro Library	If the macro is stored in a notebook designated as a macro library, choose the notebook name here. Macros/Namedblocks changes to display the macros available from the notebook.
Macros/Namedblocks	You can choose the name of a macro to run from this list. Use Macro Library to specify the list displayed.
OK	Runs the macro. The MACRO indicator appears in the status line.

To stop a macro while it's running, press Ctrl+Break and choose OK to clear the break message and return to Ready mode.

You can also run macros by first attaching them to key combinations, SpeedButtons, graph buttons, or objects in a dialog window.

See Also

Macro Topics

Making a Notebook a Macro Library

Tools | Macro | Record

Tools|Macro|Record lets you record actions as you perform them. Recording converts actions into macro commands and stores them as labels in a block.

Options

Location	Enter the coordinates where you want to store the macro. If there's no chance of overwriting data in cells below, select a single cell; Quattro Pro fills the cells below as necessary. Selecting a block larger than a cell confines recorded commands to that block only and stops recording when the block is full.
Macro Library	To store the macro in a notebook designated as a macro library, choose the library name here, and then choose a location for the macro from Macros/Namedblocks.
Macros/Namedblocks	You can give the macro a name, or choose a named block in which to store it.
OK	Starts recording the macro. The REC indicator appears in the status line. As you perform tasks, Quattro Pro records each action, writing it as one or more macro commands in the first column of the block previously specified.

To stop recording, choose Tools|Macro|Stop Recording.

See Also

Macro Topics

Making a Notebook a Macro Library

Tools | Macro | Options

Tools|Macro|Options lets you control the notation method used when you record macros.

Options

Logical	Records actions as command equivalents, the most efficient method.
Keystroke	Records actions keystroke-by-keystroke, which is slower and harder to read in the final macro. When recording in Keystroke mode, mouse actions are ignored.
Absolute	Uses standard cell addressing notation, like A1..C6.
Relative	Uses relative referencing, which specifies cells as offset from the cell selector. For example, the cell selector's relative reference is P(0):C(0)R(0).

See Also
Macro Topics

Tools | Macro | Debugger

Tools|Macro|Debugger begins Debug mode so that you can isolate and fix problems in a macro. After you choose Tools|Macro|Debugger, run the macro you want to examine.

A debug window appears in the bottom half of the screen. The first cell of the macro appears in the middle of the debug window.

See Also

Macro Topics

Debugging a Macro

Tools | Spell Check

Tools|Spell Check checks the spelling of text in selected notebook blocks or graphs. This feature is only available in the Workgroup edition of Quattro Pro.

See Also
Using the Spell Checker

Tools | Scenario Manager

Tools|Scenario Manager displays the Scenario Manager SpeedBar, which lets you save "snapshots" of data models when values are varied.

See Also
Using the Scenario Manager

Tools | Consolidator

Tools|Consolidator displays the Consolidator SpeedBar, which lets you combine two or more source blocks of data using your choice of statistical @function operators. As you combine blocks, you can sort fields and records using column and row labels.

See Also
Using the Consolidator

Tools | Define Group

Tools|Define Group lets you set up a series of pages to be acted on as a group. After you define a group, you can turn on Group mode by clicking the Group button (labeled with a G) at the bottom of the notebook window. A blue line appears below the page tabs included in the group.

You can define more than one group in a notebook, but any page can belong to only one group.

Options

Group Name	Enter a name for the group. You can use letters and numbers, but no spaces.
First Page	Enter the name of the first page in the group.
Last Page	Enter the name of the last page in the group.
Defined Groups	Lists the groups defined in the active notebook.
Delete	Lets you delete the group currently selected in the Defined Groups list box.

When Group mode is on, changes to block and page properties affect all pages in the group, pointing from within formulas and dialog boxes point to 3-D blocks, and you can enter or delete data by "drilling" through the pages in the group with Ctrl+Enter and Ctrl+Delete, respectively.

See Also

Creating a Group

Group Mode Tips

Tools | Combine

Tools|Combine lets you copy all or part of a notebook into any area of the active notebook. Unlike File|Retrieve, it doesn't erase the active notebook; it affects only the portion of the pages covered by the inserted block.

You can also perform arithmetic operations with Tools|Combine to add, subtract, multiply, or divide cells from one notebook to another. The standard options in the File Combine dialog box are explained in File-Handling Options. In addition, there are these options:

Operation	Determines how the data will be combined. Copy inserts data from a file into the active notebook; the other options determine the operations to be performed on combined data.
Source	Entire File copies the entire file into the existing one; Block(s) copies a specified block(s) of the file. If you choose Block(s), enter a block name or coordinates in the edit field. You can also specify multiple blocks by separating them with commas. (Because the block names are from another file, you can't display a list of the names.)

The contents of the specified file or block appear in the active notebook. If you copied the data, Quattro Pro also copies any properties. It copies the contents of named blocks, but to avoid confusion with names in the active notebook, it converts references to named blocks into block coordinates.

Caution: Tools|Combine overwrites any cells in the destination block, even if they are protected.

See Also

Copying Combined File Data

Operating on Combined File Data

Translating Files

Extracting Part of a Notebook

Tools | Extract

Tools|Extract saves part of a notebook to a separate file leaving the original file intact. Tools|Extract is similar to Block|Copy, with two major differences:

With Extract you can choose to copy values only. When you choose the Values option, the actual formulas aren't copied--just the resulting values.

Extract saves the notebook's block names and graphs along with the specified block. Some block names or graphs may not be meaningful if they refer to cells outside the extracted block. You can delete them, reassign them, or ignore them.

The standard options in the File Extract dialog box are explained in File-Handling Options. In addition, there are these options:

Block(s)	The coordinates or name of the block to be saved.
Password Protection	The password to be assigned to the saved file.
Formulas	When checked, saves an exact copy of the block, including formulas.
Values	When checked, converts formulas to their resulting values.

With the Formulas option, formulas adjust to reflect their new positions, even if they are absolute. (Absolute formulas remain absolute for the new notebook after their initial adjustment; they retain the $ signs, but the references adjust.) If the formulas reference cells outside the block being saved, problems can arise. In this case, it's best to use the Values option.

See Also
Extracting Part of a Notebook

Tools | Import

Tools|Import copies a text file into the active page of a notebook. A text file contains plain text (no control characters) with minimal formatting.

The standard options in the Import File dialog box are explained in File-Handling Options. In addition, there are these options:

ASCII Text File	Imports a plain, unformatted text file. Quattro Pro converts the data into a single column of labels. Each line in the file becomes a label.
Comma & "	Imports a file that uses commas and quotes (or slashes) to separate text in rows. The delimiters are used to place text in columns in the notebook page.
Only Commas	Imports a file delimited only with commas. When you choose Only Commas, Quattro Pro uses quotes optionally to enclose text that includes a comma.
Parse Expert	Imports a text file and breaks it into columns, database fields, at the same time. You may need to do some editing when the parse is complete.

After you have your text file in the notebook, you can break up the entries into separate columns with the Data|Parse command.

Tip: After you use Database Desktop to create an external query file (see Database Desktop Help for details), you can use the ASCII Text File option of Tools|Import to copy the query text into a notebook.

Note: You don't need to use Import to load Paradox, Reflex, 1-2-3, dBASE, or Symphony files. These files translate when you access them (see Translating Files for more information).

See Also
Importing Text
Combining Files

Tools | Update Links

Tools|Update Links displays these commands related to formula links between notebooks:

Open Links	Opens any or all supporting notebooks linked to the current notebook.
Refresh Links	Updates data in the current notebook based on data in any or all unopened supporting notebooks.
Delete Links	Removes all link references to any or all supporting notebooks.
Change Link	Switches all links that reference one supporting notebook to another.

See Also

Linking Notebooks

Opening Supporting Notebooks

Updating Link Values

Removing Links

Changing Notebook Links

Tools | Update Links | Open Links

The Tools|Update Links|Open Links command opens any or all supporting notebooks linked to the current notebook. The option is:

Hotlinks	Choose the supporting notebook to open. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).

Open Links opens only notebooks that are directly linked to the active notebook. If additional notebooks are linked to those directly linked, you need to activate the directly linked notebooks and choose Open Links again. For example, suppose Notebook A refers to Notebook B, which refers to Notebook C. Activating Notebook A and choosing Open Links lists only Notebook B. To open Notebook C, activate Notebook B and choose Open Links.

See Also
Linking Notebooks

Tools | Update Links | Refresh Links

The Tools|Update Links|Refresh Links command updates data in the current notebook based on data in any or all unopened supporting notebooks. The option is:

Hotlinks	Choose the closed supporting notebook from which you want to update values. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).

Refresh Links is similar to the Update References option offered when you access a primary notebook, except you can use it at any time, and you can specify one notebook at a time from which to access values.

Refresh Links is also useful when you're working on a network. While you're using one notebook, others can be working on supporting notebooks (which remain closed to you). If people announce over the network when they save or close a file, you can use Refresh Links to access updated values in the primary notebook.

See Also
Linking Notebooks

Tools | Update Links | Delete Links

The Tools|Update Links|Delete Links command removes all link references to any or all supporting notebooks. The option is:

Hotlinks	Choose the supporting notebook from which you want to delete links. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).

All link formulas from the active notebook to the supporting notebook you choose will result in ERR values.

See Also
Linking Notebooks

Tools | Update Links | Change Link

The Tools|Update Links|Change Link command switches all links that reference one supporting notebook to another. This is especially useful after you rename a notebook. The options are:

Change Link From	Choose the supporting notebook from which you want to change links.
To	Enter the name of the notebook you want to change links to.

In the active notebook, all link references to the selected notebook change to the new notebook name.

Note: The new supporting notebook must have the same layout as the previous one because the same relative cells are referenced. If it doesn't, formulas that reference linked cells may show errors or produce meaningless values. To avoid this, enter all links as block names so the correct values are included.

See Also
Linking Notebooks

Tools | Analysis Tools

Displays a SpeedBar with Analysis Tool buttons.

See Also

Using the Advanced Analysis Tools

Advanced Analysis Tools List

Tools | Optimizer

The Tools|Optimizer command finds optimum solutions for linear and nonlinear problems.

The Optimizer dialog box offers these options:

Solution Cell	specifies the cell that contains a formula you want Optimizer to maximize, minimize, or equal. The default is Max. This entry is optional; if not used, select None.
Target Value	specifies the value to be reached by the formula in Solution Cell (if there's a Solution Cell entry and it's not set to Max or Min).
Variable Cell(s)	specifies the cells Optimizer can adjust to reach an optimal solution; problem solutions display in these cells.
Constraints	defines formulas that limit the Solution Cell formula and/or Variable Cell(s) contents.
Add	adds a new constraint.
Change	edits the selected constraint.
Delete	removes the selected constraint.
Options	offers calculation settings and methods, loads and saves model settings, and generates reports.
Reset	clears Optimizer settings.

See Also
Finding Solutions with Optimizer

Optimizer Solution Cells

When using Optimizer, you can enter an optional goal-seeking formula into a notebook cell and enter its location in the Solution Cell input box. Then, you can request Optimizer to maximize, minimize, or reach a target value for that formula. Be sure to include the parts of the formula you want to vary in the list next to Variable Cell(s).

Options

Solution Cell	specifies the cell whose value you want Optimizer to maximize, minimize, or equal. The solution cell usually contains a formula whose result depends on the variable cells.
Max	indicates you want to maximize the formula in the solution cell. This is the default.
Min	indicates you want to minimize the formula in the solution cell.
None	the default; indicates you want to find a solution without considering a solution cell.

To use a target value, enter it in the Target Value input box (or reference a cell containing the target) and check the box in front of Target Value.

See Also

Tools|Optimizer

Finding Solutions with Optimizer

Optimizer Variable Cell(s)

Variable cells are cells Optimizer can adjust to reach an optimal solution. Normally, these would be part of the formula in the solution cell. For example, if solution cell B2 contained a`+E3+F3+G3`, then the variable cell block would most likely be E3..G3 (or E3, F3, G3).

To have an effect on the solution, variable cells must be related to the solution cell and to the constraints, either directly or through references. You can assign up to 200 variable cells entered as individual cells or blocks in the Variable Cell(s) input box. You may enter a list of variable cells and blocks separated by commas.

Variable cells should contain realistic values, and must not contain dates, formulas, or text. They must not be protected.

See Also
Tools|Optimizer

Finding Solutions with Optimizer

Tools | Optimizer | Options

Tools|Optimizer|Options offers additional Optimizer settings. They can help you save time or improve accuracy:

Max. Time	Indicates how long Optimizer can spend looking for the best solution (from 1 to 1000 seconds; 100 seconds is the default).
Max. Iterations	Sets the maximum number of iterations, or trials, Optimizer can perform to find the best solution (from 1 to 1000; 100 is the default).
Precision	Controls the accuracy of the solution, from 0 to 1. The default is .000001 or 1E-06. The greater the precision (the more decimal places), the more time it takes to find a solution.
Tolerance	Indicates the maximum percentage a solution can differ from a theoretical optimum integer solution. As you increase the Tolerance setting, Optimizer can produce solutions more quickly, but they may be less accurate.
Estimates, Derivatives, and Search	Change calculation methods.
Show Iteration Results	Indicates whether Optimizer should pause between iterations so you can check the progress of the search. Check it if you want to be prompted whether to continue the search or stop.
Assume Linear	Indicates whether Optimizer uses linear or non-linear methods to solve the problem.
Automatic Scaling	When checked, can help when variable cell and target values differ greatly in size. For example, variable cells can contain investment values of several million dollars, while Optimizer tries to maximize a return-on-investment percentage in the solution cell.

Load Model, Save Model	Load and save blocks of Optimizer settings for future use.
Reporting	Generates Answer and Detail Reports.

See Also

Tools|Optimizer

Finding Solutions with Optimizer

Optimizer Methods Options

The default settings for these Optimizer options are suitable for most problems. If you're experienced in mathematical programming, you'll be comfortable changing these settings. If you're inexperienced but have trouble reaching an optimal solution, experiment with them:

Estimates	specifies the approach used to obtain initial estimates of the basic variables in each iteration. Tangent uses linear extrapolation from a tangent vector, while Quadratic uses quadratic extrapolation. Quadratic may be useful in highly nonlinear problems.
Derivatives	selects Forward (the default) or Central differencing for estimates of partial derivatives. Central requires more iterations but may help if a message indicates a better solution is not available.
Search	selects a method for computing the search direction. Newton, the default, is a quasi-Newton method and may be faster, but Conjugate can help if you see a message indicating that the objective function is changing too slowly.

See Also

Optimizer|Options

Finding Solutions with Optimizer

Tools | Optimizer | Options | Reporting

You can produce two kinds of Optimizer reports:

Answer Report

This report shows how well the constraints were met. It indicates

- Starting and final values for the solution cell
- Starting and final values for the variable cells
- Value, binding, slack, and dual value for the constraints

If Assume Linear is checked in the Optimizer Options dialog box, the Answer Report also indicates

- Gradient, Increment, and Decrement for the variable cells
- Right Value, Increment, and Decrement for the constraint cells

Binding status, Yes or No, indicates whether the constraint affects the solution; if No, that constraint can be eliminated.

Slack is the difference between the constraint constant value (Right Value) and the value actually used in the solution.

Variable dual values are sometimes called reduced gradients. They show how a one-unit increase in the variable affects the solution cell.

Constraint dual values, or Lagrange Multipliers, measure the amount by which the solution could be improved if the constraint were relaxed by one unit.

Gradient is the coefficient of the objective function for each variable cell.

Right Value is the constant in a constraint expression, the value of the right-most term.

Increment for variable cells is the amount of increase in the Gradient value before a different value is calculated for that variable cell in the solution. For constraint cells, Increment is the amount of increase in the Right Value (constraint value) before the binding status changes.

Decrement for variable cells is the amount of decrease in the Gradient value before a different value is calculated for that variable cell in the solution. For constraint cells, Decrement is the amount of decrease in the Right Value before the binding status changes.

Detail Report

This report lists the variable and solution cell values at each iteration. You can see how variable cells change throughout the solution process.

See Also

Producing Optimizer Reports

Optimizer|Options

Finding Solutions with Optimizer

Tools | Solve For

The Tools|Solve For command lets you calculate a formula backwards, starting with the result and solving for a variable that produces that result. You just specify the result you want, then indicate which variable Quattro Pro can adjust to reach that result.

Options

Formula Cell	specifies the cell containing the formula you want to solve. The formula should not contain string functions.
Target Value	specifies the result you want from the Formula Cell.
Variable Cell	indicates which cell Quattro Pro can change to solve for a desired value.
Max Iterations	determines how many passes Solve For makes to solve the formula (1 to 1000).
Accuracy	specifies how close Solve For must get to the Target Value (the default is 0.0005).

See Also

Goal-seeking with Solve For

Creating Formulas

Tools | Advanced Math

Tools|Advanced Math contains these commands for performing regression analysis and matrix operations:

Regression	Performs regression analysis on blocks of data to show how one or more independent variables influence a dependent variable.
Invert	Creates an inversion matrix for an existing square matrix (a matrix containing the same number of columns as rows); used with Tools|Advanced Math|Multiply to solve matrix problems.
Multiply	Multiplies the values in two matrixes and displays the results in a third.

See Also

Performing Regression Analysis

Introducing Matrix Operations

Tools | Advanced Math | Regression

Tools|Advanced Math|Regression creates a regression analysis table showing how one to 150 independent variables can affect a dependent variable. It displays a dialog box with these fields and buttons:

Independent	specifies the block containing up to 150 columns of independent variable (x-axis) data. Each variable must be in a separate column; the independent variables can be in noncontiguous columns.
Dependent	specifies the block (partial column) containing dependent variable (y-axis) data.
Output	specifies the block where results will be written.
Y Intercept	specifies whether to force the y-intercept value to zero (choose Zero) or whether to compute it (Compute).

See Also
Performing Regression Analysis

The Regression Output Block

The regression output block is nine rows deep and three columns wider than the number of columns in the independent block. Make sure to leave enough blank space; any underlying data will be overwritten. The output block contains this information:

Constant	The y-intercept value, zero if Y Intercept is set to Zero instead of Compute.
Std Err of Y Est	The estimated standard error of *y* values; the degree of deviation of observed *y* values from predicted values.
R Squared	The variance; the degree of relationship between independent and dependent variables. With one independent variable, R Squared is the square of the correlation between the two variables.
No. of Observations	The number of values for each variable; the number of rows in the regression table.
Degrees of Freedom	The number of observations minus the number of independent values being computed by the regression. With Y Intercept set to Zero, Degrees of Freedom = (No. of Observations) - (number of independent variables); with Y Intercept computed, Degrees of Freedom = (No. of Observations) - (number of independent variables + 1).
X Coefficient(s)	The regression coefficients of the independent (x) variables; the slope of the regression line representing the relationship between each independent variable and the dependent variable.
Std Err of Coef.	An error estimate of the X Coefficient above it. Interpret each coefficient as the X Coefficient value plus or minus the Standard Error of Coefficient.

See Also

Tools|Advanced Math|Regression

Performing Regression Analysis

Y Intercept Settings

You can imagine the relationship between the dependent variable and each independent variable plotted as a graph with an x- and y-axis. The y-intercept is the point where the graph cuts the y-axis.

Options

Compute	Calculates the y-intercept.
Zero	Forces the y-intercept value to zero.

See Also

Tools|Advanced Math|Regression

Performing Regression Analysis

Tools | Advanced Math | Invert

The Tools|Advanced Math|Invert command finds the inverse of a matrix. (When you multiply a matrix by its inverse, the resultant matrix is an identity matrix: all 1s and 0s, with only a single diagonal of 1s.)

Options

Source	specifies the matrix block you want to invert.
Destination	specifies the upper left cell of the block where you want to write the inverted matrix. If you specify the same block as the invert block, the inverted matrix overwrites the existing matrix.

See Also

Tools|Advanced Math|Multiply

Introducing Matrix Operations

Inverting a Matrix

Solving Matrix Problems

Tools | Advanced Math | Multiply

The Tools|Advanced Math|Multiply command multiplies the values in one matrix by the values in a second matrix to obtain a third. The number of columns in the first matrix must equal the number of rows in the second matrix.

Options

Matrix 1	specifies the first matrix to multiply.
Matrix 2	specifies the second matrix to multiply.
Destination	specifies the top left cell of the area where you want to write the resulting matrix.

See Also

Tools|Advanced Math|Invert

Introducing Matrix Operations

Multiplying Two Matrixes

Solving Matrix Problems

Tools | SpeedBar Designer

Lets you create your own custom SpeedBars to display with the SpeedBar Control menu.

See Also
Building Custom SpeedBars

Tools | UI Builder

Opens a dialog window where you can build dialog boxes and SpeedBars for a custom application. The dialog window SpeedBar includes tools to help you create each dialog box control.

Tools|SpeedBar Designer offers another way to create custom SpeedBars. For details, see Building Custom SpeedBars.

See Also
Application Building

Window Menu

New View	Opens a duplicate window containing the active notebook.
Tile	Displays all open windows on the screen at once.
Cascade	Arranges open windows in layers.
Arrange Icons	Aligns all minimized Quattro Pro windows.
Hide	Removes the specified window from view.
Show	Displays the specified hidden window.
Panes	Splits a window into panes.
Locked Titles	Keeps one or more columns, rows, or both onscreen as you scroll a notebook.

Window | New View

The Window|New View command displays a duplicate copy of the active notebook in a new window. This is useful when you want to look at different pages in the notebook at the same time, or if you want to see distant parts of one page at the same time.

See Also

Duplicating a Window

Using Windows

Window | Tile

The Window|Tile command displays all open windows without overlapping them. When possible, the windows are all given equal room on the screen.

See Also
Resizing Windows
Using Windows

Window | Cascade

The Window|Cascade command rearranges all open windows in overlapping layers. The top line of each window is revealed so you can see the name of the notebook, graph, or dialog box it contains.

See Also

Resizing Windows

Using Windows

Window | Arrange Icons

The Window|Arrange Icons command lines up the icons from minimized Quattro Pro windows at the bottom of the Quattro Pro desktop. If the Graphs page. is the active page, Window|Arrange Icons lines up any graph, dialog box, or slide show icons on the page.

See Also
Resizing Windows
Using Windows

Window | Hide

The Window|Hide command hides an open window from view. This is useful when you have multiple notebooks open, but don't want the screen cluttered with too many windows.

See Also

Hiding and Showing Windows

Using Windows

Window | Show

The Window|Show command displays windows that have been hidden with the Window|Hide command.

See Also

Hiding and Showing Windows

Using Windows

Window | Panes

The Window|Panes command allows you to split a window either vertically or horizontally. The options are:

Horizontal	Splits the window from left to right
Vertical	Splits the window from top to bottom
Clear	Removes the active pane
Synchronize	Makes both panes scroll when you move the selector in one of the panes.

Note: You can point across panes while entering formulas or using dialog boxes.

See Also
Splitting a Window into Panes
Using Windows

Window | Locked Titles

The Window|Locked Titles command locks specific rows and/or columns of a page onscreen as titles. When you scroll the page, the titles remain fixed onscreen while the rows below (or columns to the right) scroll as usual.

See Also

Locking Rows and Columns

Using Windows

Page Column Dialog Box

The data you have selected extends across multiple pages, and requires an additional dimension in Data Modeling Desktop.

The Data Modeling Desktop will create an additional column to reference this additional "page" dimension, but needs you to provide a meaningful name for this column. Enter the name that best categorizes the dimension.

See Also

Data Modeling Desktop Help

Spell Check Options Dialog Box

The Spell Check Options dialog box controls Spell Checker behavior in the Workgroup edition of Quattro Pro:

Language	Indicates the language to check in.
Personal Dictionary	Displays the selected Spell Checker dictionary; dictionaries have a .DCT extension.
Choose New Dictionary	Lets you browse through your hard disk to find a new .DCT file.
Remove Current Dict.	Removes the current custom dictionary; the Spell Checker will then only work with the standard Quattro Pro dictionary.
Spelling Options	Check any or all spelling options. When checked,
	Ignore Repeated Words skips double words like and and.
	Ignore Capitalization Errors skips words with capital letters in the wrong places: dOg.
	Ignore Acronym Errors skips acronyms and abbreviations that appear in the dictionary with different punctuation. For example, when this option is checked, ATM and A.T.M. are considered the same.
	Ignore Words with Numbers skips labels like M9020.
	Ignore UPPERCASE Words skips any word entered in all uppercase letters.

See Also

Using the Spell Checker

Enter Password Dialog Box

The Enter Password dialog box prompts you to enter a password when you change the Password Protection Level of your notebook. Passwords can be up to 15 characters in length. The characters you type show onscreen as pound signs (######) so they are hidden from view.

Passwords are case-sensitive. For example, if your password entry has a lowercase letter but you type a capital when prompted for a password, Quattro Pro considers these different passwords.

After you enter a password and click OK, you will be asked to verify the password. If the password differs at all from the first one you typed, Quattro Pro displays an error message and cancels the password-protected operation.

Caution: If you forget a file's password, you won't be able to load the file. For this reason, record your passwords when you create them, and store the record in a secure place.

See Also

Assigning a Password to a File

Password Dialog Box

The Password dialog box prompts you for a password when you:

- Open a password-protected file
- Change the Password Protection Level of a notebook.

Passwords can be up to 15 characters in length. The characters you type show onscreen as pound signs (######) so they are hidden from view.

See Also

Assigning a Password to a File

Save Consolidation As Dialog Box

The Save Consolidation As dialog box prompts you to name the current consolidation. Consolidation names may be up to 9 characters in length.

See Also

Naming and Saving Consolidations

Add Source Block Dialog Box

The Add Source Block dialog box prompts you to enter the name of a source block to combine with another source block.

See Also

Source Blocks

Consolidation Options Dialog Box

The Consolidation Options dialog box offers these options:

Output with Formulas	Determines whether consolidation results appear as formulas or values.
Use Labels In	Determines whether data field labels are used in the consolidation.
	Top Row matches data from different blocks based on labels in the top row of each block
	Top Column matches data from different blocks based on labels in the left column of each block

See Also

Consolidator Options

Scenario Group Control Dialog Box

The Scenario Group Control dialog box offers these options:

Group Name	Displays the name of the active group.
Capture Area	Specifies the area where the Scenario Manager tracks data and format changes:
	Notebook tracks changes within the entire notebook except the Graphs page.
	Page tracks changes on the specified page.
	Block tracks changes in the specified block.
New	Creates and names a new Scenario Manager group.
Rename	Applies another name to the active group.
Options	Offers Scenario Manager options.
Delete	Deletes the active group and all scenarios in it.

See Also

The Capture Area

Scenario Groups

New Group Dialog Box

The New Group dialog box prompts you to enter a name for the current Scenario Group. Scenario Group names may be up to 9 characters in length.

See Also

Scenario Groups

Group Options Dialog Box

This dialog box indicates whether to update all scenarios in the active group when blocks are moved, inserted, or deleted. When Update on Block Operations is checked, the Scenario Manager tracks all block operations internally; no scenario changes are recorded. If unchecked, the scenarios register any moves, insertions, and deletions. If you're using the Scenario Manager for version control, you'll probably want to uncheck this option.

See Also

Tracking Versions

Rename Group Dialog Box

The dialog box lets you rename the active scenario group. Enter the new name in the edit field. Scenario Group names may be up to 9 characters in length.

See Also

Scenario Groups

Capture Scenario As Dialog Box

The dialog box lets you name the current scenario. Enter the name in the edit field and click OK. Scenario Group names may be up to 9 characters in length.

See Also

Naming and Saving Scenarios

Scenario Report Dialog Box

The Scenario Report dialog box offers several options for displaying the Scenario Summary Report. When checked, they have these effects:

Report All Groups	Displays a report for each scenario group, not just the active group.
Use Left Labels	Identifies scenario cells with text in cells immediately to their left.
Use Top Labels	Identifies scenario cells with text above them.
Find Empty Page	Puts the Scenario Summary Report on the first empty page in the notebook. To place it somewhere else, uncheck Find Empty Page and enter a page and block in the edit field.

Since these are check boxes, you can check all, any, or none. Click OK to display the Scenario Summary Report.

If you've checked a labels option or named each changing and result cell, a label or name displays next to each value in the Scenario Summary Report.

See Also

Using the Scenario Manager

Q Plus E Dialog Box

The dialog box lets you add, edit, and execute database queries through Q+E:

Add Query	Creates a new query.
Edit Query	Changes an existing query.
Execute Query	Runs the current query.

For more information, press F1 within Q+E to display Q+E Help.

Database Form Dialog Box

The Database Form dialog box lets you search and edit fields in the database you have associated with the current form.

Dialog Box Options:

New	Creates a new, blank record added after the last record.
Delete	Erases the active record.
Revert	Cancels any edits and restores original data.
Go Next	Displays the next record.
Go Previous	Displays the previous record.
Search	Displays a search form. The Search button then changes to Edit, and the Delete button changes to Clear.
Close	enters any changes from the current session and closes the form.

See Also

Using Database Forms

Properties in the Dialog Window

Add Down Button	Adds a down arrow to the right of the combo box.
Alarm On	Instructs the Timer to generate an alarm event at the specified time of day.
Alarm Time	Specifies the time to generate an alarm event (used with Alarm On).
Allow Point Mode	Lets the pointer be used to select a block while a dialog box remains on the screen.
Attach Child	Used to group controls for moving or resizing.
Bitmap	Specifies a bitmap file (.BMP) or an internal Quattro Pro bitmap to appear on the button.
Button Type	Specifies the button's behavior.
Convert Text	Allows insertion of a new line and tabs in an edit field.
Current Time	Fetches the current time. Settings cannot be altered.
Default	Determines the default value displayed in the edit field or spin control.
Default Button	Specifies the default button in the dialog box (which button is executed by pressing Enter). Only one button in a dialog box can have this property set to Yes.
Depend On	Specifies the areas of Quattro Pro in which the control appears when the dialog box displays.
Dimension	Specifies the exact size and position of the control.
Disabled	Behaves like Grayed, but doesn't dim the control. Set it to Yes to disable the control.
Draw to Right	Controls position of check box and radio button text.
Edit Disabled	Disables typing in the edit field.
Edit Length	Specifies the maximum number of characters allowed in an edit field or spin control.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Field Type	Specifies the type of text the edit field accepts.
Fill Color	Specifies the color inside the rectangle.
Frame Color	Specifies the color of the rectangle's frame.
Grayed	Specifies whether the control can be used. Set it to Yes to disable the control. The control appears dimmer to indicate it's unavailable.
Grid Options	Shows/hides a grid and sets the number of pixels between grid lines.
Group Text	Specifies the title for the group box.
Help line	A brief description that appears at the bottom of the Quattro Pro window when the control is active.
Hidden	Specifies whether the control is visible in the dialog box. Set it to Yes to hide the control. The control is hidden to the user; it's still visible when editing.
History List	Lets you set up a combo box that keeps a history of the values entered into its edit field.
Interval in Units	Determines how frequently the timer refresh event will be generated.
Label Font	Determines label format.

Label Text	Specifies the text appearing on or next to the control.
List	Specifies a pre-existing list to appear in a combo or list box, or a block containing a list.
List Length	Specifies the maximum number of History List entries.
Maximum	Determines the maximum number allowed in the edit field.
Minimum	Determines the minimum number allowed in the edit field.
Name	Used to identify the control in macros, link commands, and @functions.
Number of Columns	Allows multiple columns in a list box.
Object Help	Determines the contents of Object Help for the selected object.
Object ID	The control's identification number.
Ordered	Sorts list box entries alphanumerically.
Parameters	Controls behavior of scroll bars.
Position Adjust	Specifies how an attached control moves when its parent control is resized.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the control's value.
Rectangle Style	Determines the type of rectangle.
Resize	Adjusts the width of a pick list to match the length of the text appearing on it.
Selected	Specifies the number I.D. of a radio button in a group box, or a selection in a list box.
Selection Text	Returns the text of the item selected in a list box.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Show Frame	Displays a box around the edit field.
Show Time	Shows time on the face of the time control.
Tab Stop	Determines whether the control can be activated by Tab.
Terminate Dialog	If set to Yes, pressing Enter triggers default button (usually Enter or Exit).
Text Draw Flags	Defines the location of label text on a control.
Timer On	Specifies whether the time control generates time events at regular intervals.
Title	Specifies the title of the dialog box, which the user sees.
Units in Milliseconds	Controls Timer intervals together with the Interval in Units property.
Value	The current setting of a control. This varies from control to control.

See Also

Dialog Window Objects

Application Building

Add Down Button Property

Adds a down arrow to the right of a combo box.

See Also
Combo Box
Working with Controls

Alarm On Property

Turns the Timer alarm on or off. When set to Yes, the time control generates an alarm event at the time specified by the Alarm Time property.

See Also
Time Control
Working with Controls

Alarm Time Property

Specifies the time of day to generate an alarm event when Alarm On is set to Yes.

See Also

Time Control

Working with Controls

Allow Point Mode Property

Lets the pointer be used to select a block while a dialog box remains on the screen. The Field Type property must be set to a Field Type other than Integer before Allow Point Mode can be enabled.

See Also
Edit Field
Working with Controls

Attach Child Property

Used to group controls for moving or resizing. See Grouping Controls for directions. The Attach Child property is available for many dialog controls.

See Also

Dialog Window Objects

Working with Controls

Bitmap Property

Specifies a bitmap file (.BMP) or an internal Quattro Pro bitmap to appear on a button.

See Also

Bitmap Button

Working with Controls

Button Type Property

Specifies the type of button: Push Button, Radio Button, Checkbox Button, OK Exit Button, or Cancel Exit button. After changing button type, the Object Inspector for the new button type will become available.

Button type	Description
Push button	Acts like a push button. You can use link commands to make it perform operations when a user clicks it.
Radio button	Acts like a radio button, including a radio button's behavior in the group box. (A bitmap button appears recessed when a user chooses it.)
Check box	Acts like a check box button. (A check box button appears recessed when a user checks it.)
OK Exit button	Closes the dialog box like a standard Windows OK button. If the dialog box was displayed by {DODIALOG}, the control values are written back to the setting block.
Cancel Exit button	Closes the dialog box like a standard Windows Cancel button. If the dialog box was displayed by {DODIALOG}, the control values aren't written back to the setting block, since the dialog box was canceled.

See Also

Bitmap Button

Push Button

Check Box

Radio Button

Working with Link Commands

Working with Controls

Convert Text Property

Controls the format of text in edit fields. If Convert Text is set to Yes, and \n is entered into the string, the text following \n displays on a new line; if \t is entered, a tab is inserted (handy for lining up columns of text within the edit field). \\ Inserts a backslash (\) into the text. Use \\ when you have to enter \n or \r as normal text into the edit field.

See Also
Edit Field
Working with Controls

Current Time Property

Fetches the current time. Settings cannot be altered.

See Also

Time Control

Working with Controls

Default Property

Determines the default value displayed in a spin control or an edit field control (with Field Type set to Integer).

See Also

Spin Control

Edit Field

Edit Integer

Working with Controls

Default Button Property

Specifies whether a push button or bitmap button is automatically clicked when the user presses Enter in a dialog box or SpeedBar. A button configured this way is called the default button. A button defined as the default button displays in the dialog box with a thick black border. Only one button in the dialog box can have this property set to Yes; by default, it's the OK button. Setting a control's Default Button property to Yes automatically sets the other control's Default Button properties to No.

See Also

Bitmap Button

Push Button

Working with Controls

Depend On Property

Specifies the areas of Quattro Pro in which the dialog control is enabled when the dialog box displays. For example, some controls are only appropriate in a graph window. Depend On can be set to Desktop, Notebook, Graph, Dialog/SpeedBar, Input Line, or Graphs Page. If all of the options are checked, the dialog control is always available. The Depend On property is available for many dialog controls.

See Also

Dialog Window Objects

Working with Controls

Dimension Property

Specifies the exact size and position of the control. See Moving and Sizing Controls for directions. The Dimension property is available for many dialog controls, and is also available for Graph window objects when running Quattro Pro in /D mode.

Note: If you start Quattro Pro in Developer Mode (with the /D parameter), the Dimension property will also be available for drawn graph objects.

See Also

Dialog Window Objects

Graph Window Objects

Working with Controls

Disabled Property

Disables the dialog control in the same manner as the Grayed property, but doesn't dim the control. The Disabled property is available for all dialog controls.

See Also

Dialog Window Objects

Working with Controls

Draw to Right Property

Controls position of check box and radio button text. If set to Yes, text is placed to the right of the check box or radio button; if set to No, text is placed to the left. The default is Yes.

See Also

Check Box

Radio Button

Working with Controls

Edit Disabled Property

Disables typing in an edit field or combo box.

See Also
Edit Field
Combo Box
Working with Controls

Edit Length Property

Specifies maximum number of characters allowed in an edit field, combo box, or spin control.

See Also

Edit Field

Combo Box

Spin Control

Working with Controls

Enabled Property

Enables dialog controls that have been turned off with the Disabled property or by setting Enabled to No. It is used by link commands and doesn't appear in the control's Object Inspector.

The Enabled property is available for all dialog controls.

See Also

Dialog Window Objects

Working with Link Commands

Working with Controls

Field Type Property

Determines the allowable type of input to the edit field: Integer, String, Real, Block, or Hidden. After changing Field Type to Integer, the Object Inspector changes to Edit Integer.

Field type	Result
Integer	Restricts user entry to integers. This adds three properties to the edit field Object Inspector (which is renamed Edit Integer). Minimum defines the lowest acceptable value users can enter. Maximum defines the highest acceptable value a user can enter. Default defines the initial value of the edit field.
String	Accepts any text a user enters (including numbers); this is the default edit field type.
Real	Accepts any number or formula.
Block	Accepts only cell addresses or block coordinates.
Hidden	Accepts any text, but displays a pound sign (#) for each character instead of the actual character. This is handy when users are entering passwords.

See Also

Edit Field

Working with Controls

Fill Color Property

Specifies the color inside a rectangle.

See Also

Rectangle

Dialog Window Objects

Working with Controls

Frame Color Property

Specifies the color of a rectangle frame.

See Also

Rectangle

Dialog Window Objects

Working with Controls

Grayed Property

Specifies whether the control can be used. When set to Yes, the control appears dimmer to indicate it's unavailable. The Grayed property is available for many dialog controls.

See Also

Dialog Window Objects

Working with Controls

Grid Options Property

Shows or hides a grid in the dialog window, sets the number of pixels between grid lines, and turns Snap To Grid on or off. The grid is an aid in laying out dialog controls.

See Also

Aligning Controls

Dialog Window Objects

Working with Controls

Group Text Property

Specifies a title for a group box.

See Also

Group Box

Working with Controls

Help Line Property

A brief description that appears at the bottom of the Quattro Pro window when the control is chosen by the user. The Help Line property is available for many dialog controls.

See Also

Dialog Window Objects

Working with Controls

Hidden Property

Specifies whether the control is visible in the dialog box. Set it to Yes or No. The control is hidden to the user; it still displays when editing. The Hidden property is available for every dialog control.

See Also

Dialog Window Objects

Working with Controls

History List Property

Set this to Yes to have a combo box store a history of value entered in its edit field; this supercedes the settings of the List property.

See Also
Combo Box
Working with Controls

Interval in Units Property

Used with the Units in Milliseconds property to specify the interval at which the time control generates the timer event. The Interval in Units property determines how frequently the timer refresh event will be generated; the Units in Milliseconds property determines the number of milliseconds represented by each unit.

See Also

Time Control

Working with Controls

Label Font Property

Determines dialog label typeface and point size, and whether it is bold, italic, underlined, or strikeout.

See Also

Label

Working with Controls

Label Text Property

Specifies the text appearing on a SpeedButton or control. To follow Windows conventions, when a button leads to another dialog box, you should end its text with an ellipsis (...).

See Also
Label
Working with Controls

List Property

Specifies a block containing a list to appear in a combo or list box. If the block contents are changed subsequent to setting the List property, the List settings will not be updated until the dialog box is opened again or an EXECUTE link command is used to refresh the list box.

See Also

Combo Box

List Box

Working with Link Commands

Working with Controls

List Length Property

Specifies the maximum number of history list entries in a combo box.

See Also
Combo Box
Working with Controls

Maximum Property

Determines the highest number that can be entered in a spin control or an edit field (when its Field Type property is set to Integer).

See Also

Spin Control

Edit Field

Edit Integer

Working with Controls

Minimum Property

Determines the lowest number that can be entered in a spin control or an edit field (when its Field Type property is set to Integer).

See Also

Spin Control

Edit Field

Edit Integer

Working with Controls

Name Property

Used to identify a dialog control or drawn graph object. It can also be used by macros, link commands, and @functions to select the object, change its property settings, or read its property settings. The Name property is available for many objects.

Note: If you start Quattro Pro in Developer Mode (with the /D parameter), the Name property will also be available for many graph window objects.

See Also

Dialog Window Objects

Graph Window Objects

Working with Controls

Number of Columns Property

Specifies how many columns of items appear in a list box. The default is one column.

See Also
List Box
Working with Controls

Object Help Property

Determines the contents of Object Help for the selected object.

Object Help Title	Title of the object. This may be identical to the Help Line property text.
Object Help Text	Briefly describes the function of the object.
Winhelp Context	(optional) Specifies the Winhelp context string of the relevant help topic. The context string must be followed immediately by the @ symbol and help file name. For example, the Winhelp Context for the Quattro Pro Help Contents screen is HELP_TOPICS@QPW.HLP.

Note The combined total of all characters in the three Object Help fields may not exceed 194 characters.

See Also
Help line

Object ID Property

Assigns an identification number for dialog controls and drawn graph objects. It shows the order in which the dialog controls were created. The Object ID property is available for many objects. You can use the Object ID to identify objects in a macro command, link command, or @function. You can't change the Object ID of an object; it's a read only property.

Note: If you start Quattro Pro in Developer Mode (with the /D parameter), the Object ID property will also be available for drawn graph objects.

See Also

Dialog Window Objects

Graph Window Objects

Working with Controls

Ordered Property

Sorts list box and combo box entries alphanumerically in ascending order. Setting it to No does not reset the order of a list that has been sorted.

See Also

List Box

Combo Box

Working with Controls

Parameters Property

Controls the range of values a scroll bar accepts, and specifies how quickly the user can change the scroll bar's value.

Option	Description
Min	A horizontal bar's value when its scroll box is as far left as possible; vertical bars are set to this value when their scroll box is at the top of the scroll bar.
Max	A horizontal bar's value when its scroll box is as far right as possible; vertical bars are set to this value when their scroll box is at the bottom of the scroll bar.
Line	The amount the scroll bar's value changes when a user clicks a scroll arrow.
Page	The amount the scroll bar's value changes when a user clicks above or below the scroll box on a vertical scroll bar, or to the left or right of the scroll box on a horizontal scroll bar.
Time	The amount of time in milliseconds that Quattro Pro waits before moving the scroll box in response to a user action (clicking a scroll bar, holding down the mouse button while pointing to a scroll arrow, and so on).

See Also

Horizontal ScrollBar

ScrollBar

Working with Controls

Position Adjust Property

Specifies how an attached control moves when its parent control is resized (See Grouping Controls for directions). The Position Adjust property is available for every dialog control.

See Also

Dialog Window Objects

Working with Controls

Process Value Property

Setting this property to Yes lets the macro command {DODIALOG} access the control's value. See the description of {DODIALOG} for more information. The Process Value property is available for many objects.

See Also

Dialog Window Objects

Working with Controls

Rectangle Style Property

Determines the type of rectangle:

Option	Description
Plain	Draws the rectangle as a solid box without a frame.
Framed	Draws a frame around the rectangle (the default). To set the color of this frame, use the Frame Color property.
Beveled Out	Displays a slightly raised rectangle.
Beveled In	Displays a slightly recessed rectangle.
Transparent	Displays a clear, unfilled rectangle. This is handy for grouping controls together without displaying the rectangle they're attached to. (See Grouping Controls for details on grouping controls).

To change a rectangle's color, right-click the rectangle and choose the Properties command, then choose Fill Color.

See Also
Rectangle
Working with Controls

Resize Property

If set to Yes, makes the pick list automatically resize whenever the text appearing on it is too wide to fit.

See Also
Pick List
Working with Controls

Selected Property

Indicates numerically which radio button in a group box is chosen, or which item in a list box (or pick list) is highlighted. Numbers are sequential (such as 0,1,2,3,...9). If the first item is chosen or highlighted, Selected is set to 0. Choosing or highlighting the second item sets Selected to 1, and so on. To get a unique ID for a specific control in a group box, right-click the control and choose the Properties command, then choose Object ID.

See Also

Group Box

List Box

Pick List

Working with Controls

Selection Text Property

Returns the text of an item selected in a list box.

See Also
List Box
Working with Controls

Show Property

The opposite of Hidden. Set it to No to hide the control from the user. It's used by link and macro commands and doesn't appear in the control's Object Inspector.

See Also

Dialog Window Objects

Working with Controls

Show Frame Property

Displays a box around an edit field or spin control. Default is on.

See Also

Edit Field

Spin Control

Working with Controls

Show Time Property

If set to Yes, shows the system time on the face of the time control.

See Also
Time Control
Working with Controls

Tab Stop Property

Determines whether the control can be activated by Tab. See Setting Tab Order for directions. The Tab Stop property is available for many dialog controls.

See Also

Dialog Window Objects

Working with Controls

Terminate Dialog Property

If set to Yes, pressing Enter triggers the default button (usually Enter or Exit). This property is available for combo boxes and edit fields.

When set to No, keeps the dialog box from activating its default button when the user presses Enter in an edit field. (See Default Button for further information.) This is important in SpeedBars, since activating the default button would close the SpeedBar.

See Also

Combo Box

Edit Field

Working with Controls

Text Draw Flags Property

Determines where the text appears in relation to its background (for check boxes and radio buttons) or in relation to the button itself (for push buttons and bitmap buttons).

Options	Description
Apply	Applies the Text Draw Flags properties you've selected.
Wrapping Text	Single Line specifies whether the label converts formatting codes. Uncheck this to allow formatting codes to be entered into the label (discussed next). This also enables the Word Break option. Word Break specifies whether to wrap the text onto a new line in the label background. Check this to enable line wrapping (available only when Single Line is unchecked).
Vertical Position	Top places text flush against the top edge of the label background. Center Vertically centers the text between the top and bottom of the background. It doesn't affect the horizontal position of the text. Bottom places text flush against the bottom of the background.
Horizontal Position	Left places text flush against the left side of the label background. Center Horizontally centers the text between the left and right sides of the label background. It doesn't affect the vertical position of the text. Right places text flush against the right side of the label background.

You can enter two special text items in a label to change its appearance:

When the Single Line option (in the Text Draw Flags property) is unchecked, you can enter the following escape sequences:

\n (or \r) makes characters to the right of it display on a second line. This only works when the Single Line option (in the Text Draw Flags property) is unchecked.

\t inserts a tab into the text (a fixed amount of space that's handy for lining up columns of text). This only works when the Single Line option (in the Text Draw Flags property) is unchecked.

\\ inserts a backslash (\) into the text. You can use this to enter \n, \r, or \t into a label as normal text.

& makes characters to the right of it appear underlined in the label text (the ampersand doesn't appear). This is handy for naming controls.

See Also

Bitmap Button

Check Box

Push Button

Radio Button

Working with Controls

Timer On Property

Turns on or off the Timer, which specifies whether the time control generates time events at regular intervals.

See Also

Time Control

Working with Controls

Title Property

Determines the dialog box title or pick list title displayed to the user.

See Also

Active Dialog Window Properties

Pick List

Working with Controls

Units in Milliseconds Property

Controls Timer intervals together with the Interval in Units property. The Interval in Units property determines how frequently the timer refresh event will be generated; the Units in Milliseconds property determines the number of milliseconds represented by each unit.

See Also

Time Control

Working with Controls

Value Property

The current setting of a control. This varies from control to control. See the control descriptions for a discussion of their values. Value doesn't appear in the Object Inspector; you can access it using macros, link commands, or @functions. All dialog controls have a Value property.

See Also

Dialog Window Objects

Working with Link Commands

Working with Controls

Properties in the Graph Window

3-D Options
3-D View
Alignment
Analyze
Aspect Ratio
Bar Options
Bkg Color
Border Color
Border Options
Border Style
Box Type
Comparison Line
Explode Slice
Fill Color
Fill Style
Format
Graph Button
Graph Type
Grid
HiLo Bar Style
Label Alignment
Label Options
Legend Position
Line Color
Line Style
Major Grid Style
Marker Style
Minor Grid Style
Name
Numeric Format
Scale
Series Options
Subtitle Bkg Color
Subtitle Color
Subtitle Font
Subtitle Style
Text Bkg Color
Text Color
Text Font

Text Style
Tick Options
Title
X-Axis Series

See Also
Objects in the Graph Window

3-D Options Property

The 3-D Options property affects the appearance of the two walls and the base of 3-D graphs. The options are:

Show Left Wall	hides or reveals the left wall.
Show Back Wall	hides or reveals the back wall of the graph.
Show Base	hides the area under the series display.
Thick Walls	gives the graph 3-D walls. When thick walls are turned off, the walls look thin, like sheets of paper. This option applies to all walls in the graph.

The property is available from the graph setup and background Object Inspector when a 3-D graph is displayed in a graph window.

When wall or base display is on, you can right-click the wall or base and change its Fill and Border Properties. When display is off, only grid lines are visible, and you cannot display an Object Inspector.

See Also

Changing 3-D Options

Major Grid Style

Minor Grid Style

3-D View Property

The 3-D View property lets you look at a 3-D graph from different vantage points. A sample graph in the dialog box lets you see changes before you apply them to your graph.

Rotation	pivots the graph on its base. You can adjust the rotation to any angle between zero and 90 degrees. When rotation is set to zero, you look directly at the front of the graph; when rotation is set to 90, you look directly at the right side of the graph. The default setting is 30.
Elevation	lets you view a graph from different heights. When elevation is set to zero, you are at the same level as the base of the graph. As you increase the angle toward 90 degrees, your view shifts to see more of the top of the graph. At 90 degrees, you look directly at the top of the graph. The default setting is 30 degrees.
Perspective	makes objects in the distance appear smaller than objects that are close. As you increase the Perspective setting, graph objects that are closest to your view get larger in proportion to objects that are farther away, and the graph looks somewhat distorted. When perspective is off, you view the graph at an ideal distance, and there is no distortion. When you check Perspective to turn this feature on, a slider bar for adjusting the amount of perspective appears. To turn perspective off, uncheck it.
Depth	adjusts the distance between the front and back of the graph. When depth is set to 166, the top and base of a 3-D bar are square. To increase the depth, move the slider to the right. To make the graph appear flatter, or more two-dimensional, move the slider to the left.
Height	is proportional to width when Quattro Pro draws a graph with the standard 3-D View settings. This height is given the value 100. To increase the proportion of height to width, move the slider to the right. To decrease the height to width ratio, move the slider to the left.
Apply	changes the view options of the graph without closing the Object Inspector; use this to preview your settings.
Reset	changes the 3-D View settings back to defaults (this doesn't affect the graph).

The 3-D View property is available from the graph setup and background Object Inspector when a 3-D graph is displayed in the graph window.

Alignment Property

The Alignment property affects the placement of text within titles, subtitles, and text boxes. The options are:

Text Alignment	makes text left-justified, centered, or right-justified.
Wordwrap	if checked, text goes to the next line when it reaches the right margin, and the box expands vertically to fit all the text you type. If unchecked, the box expands horizontally as you type; you have to press Enter to start a new line.
Tab Stops	sets tab stops within the text or title box at regular intervals you specify.

The Alignment property is available in Object Inspectors for the box that encloses the graph title and subtitle (see Graph Title Box for more information), and for boxed text you create using the Text tool on the SpeedBar (see Box for details).

See Also
Align
Label Alignment

Analyze Property

The Analyze property is available in the area, bar, and line series Object Inspectors in the graph window. It controls the type of analytical graphing in use, if any. The options are:

None	Disables analytical graphing.
Aggregation	Combines, or aggregates, multiple data points and plots them as a single point that may be the sum, average, standard deviation, variance, minimum, or maximum of the data.
Moving Average	Smooths fluctuating data points and highlights trends by plotting progressive averages for a specified number of points.
Linear Fit	Generates a straight line that best fits the data.
Exponential Fit	Generates a curve to fit data that increases or decreases geometrically.

See Also

Guidelines for Analytical Graphing

Result Tables

Area Series

Bar Series

Line Series

Aspect Ratio Property

The Aspect Ratio property in the graph window Object Inspector affects the proportion of width to height in a graph. The options are:

Floating Graph	retains the proportions of the floating graph, even when you bring this graph into a graph window.
Screen Slide	gives the correct proportions to a graph that will be part of an onscreen slide show. This is the default aspect ratio.
35mm Slide	gives the correct proportions to a graph that will be sent to a slide service for processing into 35mm slides.
Printer Preview	shows the graph as it will look when printed with the current page setup.
Full Extent	adjusts the graph display to fill the graph window and fit the area designated for the floating graph. Full extent assumes that graphs and drawings are at their proper size when displayed full-screen; these elements are scaled down in smaller views, while fonts stay the same size no matter where they are displayed. This is why graphs with the full extent aspect ratio often look different on a spreadsheet page than they do in a graph window.

Tip: Choose the aspect ratio for your final output before you begin to change properties or draw objects. A 35mm slide has different proportions than an 8 1/2 by 11 inch sheet of paper, for example. A graph and its labels may look right when the aspect ratio is set to 35mm screen slide, but everything may need to be resized when the aspect ratio is changed to Printer Preview.

Bar Options Property

The Bar Options property is in the Bar Series Object Inspector available when you right-click a bar in a graph and choose the Properties command (or choose Property|Series|series number in the menu bar). These options apply to all bars in the graph:

Bar Width	controls the thickness of the bars. When Quattro Pro draws a bar graph, it divides the axis spaces evenly between the bars (reserving some of the axis area for spaces between the bars). The initial bar width depends on how many values you plot. To make the bars thinner, move the Bar Width Percentage slider to the left; to make the bars thicker, move the slider to the right.
Bar Margin	controls the amount of extra space between the bars and the right and left sides of the graph. A bar margin can give your graph a more polished look.
Bar Overlap	determines how much the bars in a series cover the bars of the previous series. There are three settings: None (bars are placed side-by-side), Partial (bars overlap by 50% of their width), and Full (each series is drawn directly on top of the last one).

Bkg Color Property

The Bkg (Background) Color property is available for every solid object that appears in or on a graph. The background color is visible only if the Fill Style is a pattern or a wash. When the fill style is a pattern, the background color is the color *behind* the pattern. In a wash, the fill color is distributed over the background color.

See Also
Selecting Fill, Background, and Border Colors

Border Color Property

The Border Color property is available for every solid object that appears in or on a graph. The border color is the color of the frame or box around the object.

See Also

Border Style

Box Type

Selecting Fill, Background, and Border Colors

Border Options Property

The Border Options property lets you turn the display of each graph pane border off and on. This property also specifies whether graph grids appear behind or in front of bars, areas, and other graph markers.

To display the grid in front of bars and other graph markers, check Grids On Top.

The four border options--Left Border, Top Border, Right Border, and Bottom Border--appear automatically when you open the Object Inspector. To hide a side of the graph pane, uncheck the border option. To reveal it, check the option.

When a hidden side coincides with an axis, the vertical or horizontal line that represents the axis is hidden, but all other features, such as tick marks and labels, are displayed.

Tip: If you turn off left and bottom border displays to hide axis lines, you'll see a "plus sign" at the junction of the x- and y-axes, where tick marks cross. To eliminate these marks, right-click the x-axis, choose the Properties command, and set Tick Options|Tick Style to None, then do the same for the y-axis.

The Border Options property is available when you right-click the graph pane of a 2-D graph and choose the Properties command.

Border Style Property

Border Style sets the style or thickness of the frame around the object. There are nine styles, ranging from transparent to a thick line. Buttons in the dialog box illustrate each style; just click a button to choose a style. The color of the border is controlled by the Border Color property

The Border Style property is available for every solid object that appears in or on a graph, except the graph background and text boxes (these have a similar property; see Box Type for more information).

See Also
Fill and Border Properties

Box Type Property

The Box Type property is available for the graph background, text boxes, and floating graphs. Box type determines the look of the box that borders these objects. The 12 choices range from "no frame" to several 3-D styles. Buttons in the dialog box illustrate each box type; just click a button to choose one. A similar property, Border Style, sets the frame appearance for other graph objects.

See Also
Fill and Border Properties

Comparison Line Property

Selects the style and color of a comparison line for a stacked or 100% stacked bar comparison graph. The color and line options are identical to those in line graphs.

See Also

Line Color

Line Style

Explode Slice Property

The Explode Slice property is available for 2-D and 3-D pie and doughnut graphs, including multiple pie graphs. Pie and doughnut graphs plot a single series; each value is a "slice" of the graph circle. To emphasize a slice, you can explode it (pull it away from the rest of the circle). The options are:

Explode Distance	The amount of space between the pie or doughnut and the exploded slice. The space is measured as a percentage of the radius (the default value is 20%).
Explode	Enables the explode feature.

Multiple 2-D and 3-D pie graphs plot each series as a separate pie. When you explode a slice in one pie, the corresponding slices in the other pies are also exploded.

See Also
Customizing Pie and Doughnut Graphs

Fill Color Property

The Fill Color property is available for every solid object that appears in or on a graph. Fill Color is the interior color of the object. If the Fill Style is a pattern, the fill color is the color of the pattern. If the fill style is a wash, the fill color is distributed over the Bkg Color.

See Also
Selecting Fill, Background, and Border Colors

Fill Style Property

The Fill Style property is available for every solid object that appears in or on a graph, including bars, areas, line graph markers, and drawn objects such as ellipses and rectangles. The options are:

None	makes the object transparent.
Solid	makes the object one solid color (the Fill Color).
Pattern	gives you a choice of 24 two-color patterns. The color of the pattern is the Fill Color; the background behind the pattern is the Bkg Color.
Wash	gives you a choice of 6 two-color wash styles. In a wash, the Fill Color is distributed over the Bkg Color.
Bitmap	lets you fill the object with an imported bitmapped graphic (see Filling an Object with a Bitmap for details).

See Also

Bkg Color

Fill Color

Selecting a Fill Style

Changing Box or Border Style

Making an Object Solid or Transparent

Format Property

The Format property specifies how series labels are displayed (as text, or in a numeric format such as date or currency, for example). See Numeric Format Options for more information.

See Also
Adding Labels to Bars and Data Points

Graph Button Property

The Graph Button property is available for every Text Box and for the Graph Background. This property turns a text box, or the graph background, into a "live" button in a slide show. When clicked, a graph button can run a macro or show a different graph. Graph buttons are active whenever the graph appears in a slide show or when displayed with the Graph|View command. They do not work in a graph window, or in a floating graph on a spreadsheet page.

The background graph button defines what action should be taken if a viewer clicks on any area outside of the defined text-box graph buttons. This invisible button traps random or errant clicks, which makes it well suited for use in slide shows. Each graph can have as many text box graph buttons as you want, but there is only one background button per graph.

See Also

Creating a Graph Button

Creating a Background Graph Button

Turning Graphs into Slide Shows

Graph Type Property

The Graph Type property is available when you right-click the graph background and choose the Properties command when a graph is displayed in a graph window. This property controls how a series is represented in a graph--as a bar, line, area, or some variation of these basic graph types. Quattro Pro divides graph types into five categories. Click one of the topics below to learn about these categories:

2-D Graphs

3-D Graphs

Rotated Graphs

Combination Graphs

Multiple Graphs (found under Combo in the Graph Type dialog box)

Text Graphs

Click one of these topics to learn about specific graph types:

Bar Graphs

Line Graphs

Area Graphs

Pie and Column Graphs

Doughnut Graphs

XY Graphs

High-Low Graphs

Surface Graphs

Radar Graphs

Text Graphs

See Also

Choosing a Graph Type

Changing the Graph Type

Grid Property

Grid is one of two Graph Window Properties. A grid is a series of vertical and horizontal dotted lines spaced at regular intervals. When you draw objects, the grid can help you size and place objects correctly. Grid lines are a design aid only; they do not print or appear in slide shows. The options are:

Display Grid	turns the grid display on and off.
Grid Size	controls the distance between the dotted grid lines. The size is measured as a percentage of the graph window area. The default grid size, 4, means the distance between each grid line is 4% of the edge it intersects, with 25 increments in each direction (a 25 x 25 grid). The grid has the fewest guidelines when the grid size is set to 25 (a 4 x 4 grid), and the densest guidelines when the grid size is set to 1 (a 100 x 100 grid).
Snap to Grid	when checked, objects automatically align with the nearest grid point when you create or move them. You can align several objects by placing them close to the same grid line. The grid forces the objects to align with the grid and with each other.

Display Grid and Snap to Grid work independently. For example, you can place objects using Snap to Grid even when the grid display is turned off.

Note: This property is not related to the grid lines that project from the x- and y-axes on a graph. To learn about these grid lines, see Major Grid Style and Minor Grid Style.

Hi-Lo Bar Style Property

This property determines how high and low values are represented in High-Low Graphs. There are four styles:

I-Beam	is the default style. High and low values determine each end of the I-beam. Open and close are represented by left and right tick marks, respectively.
Line	connects corresponding high and low values with a line and shows open and close values as left and right tick marks.
Bar	give you a "bar and whisker" (or "candle") graph. A line connects high and low values. A bar spans the open and close values. When the close value is higher than the open, the bar is white. When the open value is higher than the close, the bar is blue. (Retain this light/dark relationship if you change fill colors.)
Marker	assigns different-colored markers to high, low, open, and close values, and connects each set of corresponding values with a line.

Label Alignment Property

A label series places labels on the bars, markers, or data points that represent each value in a series. (See Adding Labels to Bars and Data Points for details.) The Label Alignment property is available when you right-click one of these labels and choose the Properties command.

Label Alignment sets the position of series labels relative to the bars or data points. The graph type determines which alignment options are available.

On bar graphs, you have four choices, Above, Top, Middle, and Bottom. Above places the labels above the top margin of each bar. Top, Middle and Bottom place labels on the bars, in different locations. Top places labels near the top margin; middle puts labels over the center of each bar, and bottom places labels just above the x-axis.

On line and area graphs, you can center series labels over the data points, or place each label to the Left, Above, to the Right, or Below the corresponding data point.

Label Options Property

The Label Options property is available when you right-click a slice in a pie or doughnut graph or a section of a column graph and choose the Properties command. The options are:

Label Series	places text labels next to the data labels. (If you have an x-axis series defined for the graph, the x-axis series appears in this edit field.)
Data Label	controls the format of the label that displays the value for each slice. Currency shows the values from the spreadsheet page, in dollars. Percent (the default) shows the data as a percentage of the value of the whole series. Value shows the actual data that appears on the spreadsheet page. None means no value labels appear next to the slice.
Show Tick	turns tick mark display off and on. Be sure to turn this option off if you don't have a label series defined, and if you set Data Label to None.

When you've defined an x-axis series, its block coordinates appear in the Label Series edit field. Either the x-axis series or the label series can provide labels in pie and column graphs. If you specify a new label series, it overrides the x-axis series.

See Also

Customizing Pie and Doughnut Graphs

Customizing Column Graphs

Legend Position Property

The Legend Position Property specifies where a legend box appears on the graph. The left option hides the legend. The middle option creates a horizontal legend and places it below the graph. The right option creates a vertical legend and places it to the right of the graph.

This property is available in the graph setup and background Object Inspector and the legend Object Inspector for all 2D graphs (except pie and column graphs) and for 3D graph types that have legend boxes.

Line Color Property

The Line Color property is available for all line series (including those in Line graphs, XY graphs, High-Low graphs and in line series in combination graphs you create by overriding the graph type of an individual series). This property determines the color of the line that connects the points in a series, the color of outline-style markers, and the border color of solid (filled) markers.

The Line Color property also chooses the color for the lines, polylines, and freehand lines you create using tools on the graph window SpeedBar.

See Also

Line Style

Marker Style

Line Style Property

The Line Style property is available for all line series (including those in Line graphs, XY graphs, High-Low graphs and in line series in combination graphs you create by overriding the graph type of an individual series). This property determines the thickness of the line that connects the points in a series, the color of outline-style markers, and the border color of solid (filled) markers. You can also choose from several dashed line styles.

Line Style also chooses the style for the lines, polylines, and freehand lines you create using tools on the graph window SpeedBar.

See Also
Line Color
Marker Style

Major Grid Style Property

The Major Grid Style property is available for the x-axis and the primary and secondary y-axes of a graph. Major grid lines extend from each major tick on an axis to the opposite side of the graph. The options are:

Line Style	lets you choose the appearance of the grid lines. The blank selection in the upper left corner turns the grid off.
Color	lets you select a color from the palette, or use the scales to create a new color (see Creating Custom Color Palettes for more about color scales).

The appearance of the major grid is also affected by the Increment value, which is a Scale property option.

See Also
Minor Grid Style

Marker Style Property

The Marker Style property is available for all line series (including those in line graphs, XY graphs, high-low graphs and in line series in combination graphs you create by overriding the graph type of an individual series). This property specifies the symbol placed at each data point in a series.

There are sixteen marker styles. Eleven of these are *outline* styles--markers with no interior fill. The other five are *solid* markers with filled interiors. The color of outline-style markers and the border color of solid markers is determined by the Line Color. The type of line that forms the outline or border is set by the Line Style property.

Minor Grid Style Property

The Minor Grid Style property is available for the x-axis in an XY Graph and in the primary and secondary y-axes of a graph. Minor grid lines extend from each minor tick on an axis to the opposite side of the graph. The Minor Grid Style property selects the line style and line color for these lines. The options are:

Line Style	lets you choose the appearance of the grid lines. The blank selection in the upper left corner (the default) displays no lines.
Color	lets you select a color from the palette, or use the scales to create a new color (see Creating Custom Color Palettes for more about color scales).

The appearance of the minor grid is also affected by the No. of Minors value, which is a Scale property option.

See Also
Major Grid Style

Name Property

The Name property is available when you view the properties of graph icons and slide show icons on the Graphs page. This property lets you rename graphs and slide shows.

Note: If you start Quattro Pro in Developer Mode (with the /D parameter), the Name property allows you to assign a name to drawn graph objects.

See Also

Graph Window Objects

Graph Icon Properties

Slide Show Icon Properties

Numeric Format Property

The Numeric Format property is available for labels on the primary y-axis, the secondary y-axis (which has the same set of properties as the primary y-axis) of a graph, and for the scaling x-axis of XY Graphs.

By default, the Numeric Format setting is General, which means the numbers on the axis scale are displayed exactly as they are calculated from your data. Other formats are available, including formats that add commas, dollar signs, or other characters to your original number.

If the format you choose allows a variable number of decimal places, an edit field appears. If you want to display other than the default number of decimal places (2), enter a number from 0 to 15 in the edit field.

If you choose Date or Time, you must choose a specific date or time format.

If you choose User Defined, you can choose from a list of formats you create yourself (see Defining Custom Numeric Formats for details).

See Numeric Format Options for descriptions and examples of each numeric format.

Scale Property

The Scale property is available for labels on the primary Y-Axis, the secondary y-axis (which has the same set of properties as the primary y-axis) of a graph, and for the scaling x-axis of XY Graphs. When Quattro Pro draws a graph, it automatically adjusts the axis scale to fit the range of numbers assigned to the axis. You can change this scale to fine-tune the graph or to zoom in on a specific area. The options are:

Scale Type	sets the scale to Normal or Log (logarithmic). In a normal scale, the value of each major division is exactly the same. In a logarithmically scaled axis, each major division of the axis represents 10 times the value of the previous major division. A logarithmic scale is useful when you're plotting series with wide ranges in magnitude.
Show Units	simplifies the axis labels where numbers on the axis scale are 1000 or greater. Show Units displays only the first characters of the label and automatically adds the appropriate units title in parentheses along the y-axis: "(thousands)" or "(millions)" for example. (To change the font, color, or other properties of the Show Units title, right-click it and choose the Properties command to display the Axis Title Object Inspector.)
Automatic	refers to the scale that is automatically calculated from your data. If you change High, Low, Increment, or No. of Minors settings, this option becomes unchecked. If you check it, it clears your changes and resets the scale to Quattro Pro's original settings.
High and Low	set the highest and lowest values for the axis scale. (To set a descending scale, reverse the high and low values.)
Increment	sets the increase (or in descending scales, the decrease) in values between major tick marks. This option also affects the spacing between major grid lines.
No. of Minors	specifies how many minor tick marks and minor grid lines to place between major tick marks. (The default style for minor grid lines is "no lines," so you won't see minor grid lines unless you select a different Line Style option for the Minor Grid Style property.)
Zero Line	appears as an option of the Scale property only when you right-click the y-axis of a variance graph and choose Properties. The zero line provides the frame of reference for the "variance" that gives this graph type its name. Values less than the zero line setting project below the line; higher values extend above the line.

See Also
Major Grid Style

Series Options Property

Series Options appear in the Line Series Object Inspector, the Bar Series Object Inspector, and the Area Series Object Inspector for 2-D and 3-D graphs. To view the properties of a particular series, right-click the bar, line, or area that plots the series and choose the Properties command, or choose Property|Series|series number in the menu bar. The options are:

Data Series	selects or changes the data series to be plotted. See Changing the Data Series details.
Label Series	places labels over bars in bar graphs, and over data points in line and area graphs. See Adding Labels to Bars and Data Points to create a label series. To adjust the position and appearance of these labels, see Series Label.
Legend	lets you enter legend text (or the coordinates where legend text is located), to override the legend label that is placed in the graph by the series. See Overriding a Legend Label for instructions.
Override Type	changes the graph type of the series you right-clicked. Use this option to create a custom combination graph. You can override a series on 2-D bar, line, variance, and high-low graphs. (High-low graphs restrict the override to the fifth series or greater.) See Creating a Custom Combination Graph for details.
Y-Axis	lets you plot the selected series against the primary y-axis or secondary y-axis. This option is available only for 2-D bar, line, variance, and high-low graphs. See Plotting Series on the Secondary Y-Axis for instructions.

See Also
Legend

Subtitle Background Color Property

The Subtitle Background (Bkg) Color property is available for graph subtitle text only. This property provides the second color for pattern and wash subtitle styles. The subtitle background color is also the drop shadow color.

See Also

Adding Titles to the Graph

Subtitle Color

Subtitle Font

Subtitle Style

Subtitle Color Property

The Subtitle Color property chooses the color for graph subtitle text. When you select a pattern subtitle style, the subtitle color is the color of the pattern. If you choose a wash subtitle style, the subtitle color is distributed over the subtitle background color.

See Also

Adding Titles to the Graph

Subtitle Bkg Color

Subtitle Font

Subtitle Style

Subtitle Font Property

The Subtitle Font property determines the font and font size of the graph subtitle, and whether the subtitle text appears bold, italic, underlined, or with strikeout.

See Also

Adding Titles to the Graph

Subtitle Bkg Color

Subtitle Color

Subtitle Style

Subtitle Style Property

The Subtitle Style property includes options to fill graph subtitle text with a solid color, a two-color wash, or an imported bitmap. It also lets you add a drop shadow.

See Also

Adding Titles to the Graph

Subtitle Bkg Color

Subtitle Color

Subtitle Font

Text Bkg Color Property

The Text Bkg (Background) Color property is available for all text objects, including titles, axis labels, and text you create using the Text tool on the graph window SpeedBar. This property provides the second color for a pattern or wash Text Style, and is also the shadow color. The text background color isn't visible when the text style is a solid color.

See Also
Text Color
Text Font

Text Color Property

The Text Color property is available for all text objects, including titles, axis labels, and text you create using the Text tool on the graph window SpeedBar. This property determines the color of solid-colored text. When you select a pattern Text Style, the text color is the color of the pattern. If you choose a wash text style, the text color is distributed over the Text Bkg Color.

See Also
Text Font

Text Font Property

The Text Font property is available for all text objects, including titles, axis labels, and text you create using the Text tool on the graph window SpeedBar. Text font lets you choose the typeface (Courier, Helvetica, Times Rmn, for example), the type size (in points), and whether the text has a bold, italic, underline, or strikeout feature.

See Also

Text Bkg Color

Text Color

Text Style

Text Style Property

The Text Style property is available for all text objects, including titles, axis labels, and text you create using the Text tool on the graph window SpeedBar. This property includes options to fill text with a solid color, a two-color wash, or an imported bitmap. It also lets you add a drop shadow.

See Also

Text Bkg Color

Text Color

Text Font

Tick Options Property

The Tick Options property is available for the X-Axis, the Y-Axis, and the secondary y-axis (which has the same set of properties as the primary y-axis) of a graph. Major divisions on the axis are indicated by tick marks. You adjust the display and placement of tick marks and axis labels through the Tick Options property.

Tick Style	controls the appearance of tick marks. None means no tick marks are displayed. The Below and Above settings extend tick marks below or above the x-axis, respectively. The Left and Right settings extend tick marks to the left or right of the y-axis, respectively. The Across setting makes tick marks cross the axis.
Display Labels	turns axis label display off and on.
Number of Rows	places the labels in a single row, or in two or three staggered rows.
No Overlapping Labels	eliminates some of the labels when label text would otherwise overlap.
Skip __ Labels	lets you choose how many evenly-spaced labels to eliminate, to prevent overlap. If you enter 2, for example, the first label is displayed, the second and third labels are skipped, the fourth label is displayed, the fifth and sixth labels are skipped, and so on.
Length Limit	lets you specify the maximum number of characters displayed in a label. Any additional characters won't appear in the label.

See Also

Major Grid Style
Minor Grid Style
Scale
Text Bkg Color
Text Color
Text Font
Text Style
X-Axis Series

Title Property

The Title property provides an edit field where you can change the text of an x-axis or y-axis title. This property is equivalent to choosing Graph|Titles and changing the entry for the corresponding axis.

See Also

Adding Titles to the Graph

Developer Mode

XAxis Series Property

The X-Axis Series property lets you add or change x-axis labels. This property is equivalent to selecting an x-axis series when you create a graph. To do this,

1. Enter axis labels in a spreadsheet page, if a suitable block of labels doesn't already exist.
2. Right-click the x-axis and choose the Properties command, or choose Property|X-Axis. The X-Axis Series property is already selected in the Object Inspector.
3. Double-click the Select Range edit field, then point to the block of axis labels on the spreadsheet page. If your labels are not in a continuous row or column, hold down the Ctrl key as you select each cell or subblock.
4. Press Enter to return to the Object Inspector (or type the coordinates directly in the edit field), then choose OK.

In XY Graphs (scatter diagrams), the x-axis series is data, not labels. Quattro Pro gives the x-axis a scale to match the data.

Properties in the Notebook Window

Block Alignment	Changes alignment of cell entries in a block.
Border Color	Sets the color of a floating object's border.
Borders	Removes or displays the borders.
Box Type	Determines the type of box to surround the floating object.
Column Width	Sets and resets the width of selected columns.
Conditional Color	Assigns colors ERR values or to values within, greater than, and less than a specified range.
Data Entry Input	Constrains types of data that can be entered in a block.
Default Effect	Specifies a default slide show effect.
Default Width	Sets the column width for all columns, except those that have been adjusted individually.
Display Zeros	Determines whether or not to display zero values.
Display	Hides parts of the notebook display.
Font	Sets typeface, style, and point size of text for the selected block.
Grid Lines	Hides and displays the spreadsheet grid.
Label Alignment	Sets the position of labels within cells, unless set individually.
Label Text	Specifies text to display on a SpeedButton.
Line Color	Changes colors of a page's drawn lines.
Line Drawing	Draws lines around cells and blocks.
Link Settings	Controls OLE links.
Macro Library	Turns a notebook into a macro library.
Macro	Specifies a macro for the selected SpeedButton to run.
Name	Specifies a name for a graph, dialog, or slide show icon in the Graphs page.
Name (Page)	Names the active page.
Numeric Format	Changes display format of values in a block.
Object Name	Specifies the name of a floating object.
Object Settings	Controls behavior of an OLE object.
Palette	Determines the default color palette available for notebook cell colors.
Password Level	Sets the level of password protection for the active notebook.
Protection (Page)	Sets or removes overall write-protection for a spreadsheet page.
Protection (Block)	Disables or enables protection for individual blocks of cells.
Recalc Settings	Determines calculation mode, order, and number of iterations for spreadsheet recalculation.; toggles formula compilation and display of error sources in cells.
Reveal/Hide	Hides and displays columns and rows.
Row Height	Sets and resets the height of selected rows.
Shading	Shades blocks of cells with the specified color.
Source Graph	Determines the graph displayed in a floating graph.

System	Turns a notebook into a system notebook.
Tab Color	Changes the tab color of the active notebook page.
Text Color	Specifies text color for the selected block.
Zoom Factor	Proportionally reduces or enlarges cells and floating objects onscreen without actually resizing them.

See Also
Objects in the Notebook Window

Recalc Settings Property

The notebook Recalc Settings property controls how Quattro Pro updates formula results when you change values those formulas depend on. Options are discussed in the following topics:

Setting Recalculation Mode to specify when formulas are recalculated

Setting Recalculation Order to change the sequence in which formulas are calculated (rarely changed)

Limiting Recalculation Iterations to determine how many times formulas are recalculated before calculation is considered complete (rarely changed).

Compiling Formulas to save memory and boost recalculation speed for certain types of notebooks.

Auditing Errors to display the source of ERR and NA values in cells.

See Also
Active Notebook

Zoom Factor Property

The notebook Zoom Factor property lets you pull back to see a whole printed page, or focus in on the detail of a few cells.

The default setting is 100%. Percentages less than 100% show more columns and rows; percentages greater than 100% show fewer.

Note: The Zoom Factor setting doesn't affect printed output; use the Scaling option in the File|Page Setup dialog box to scale printed documents (see File|Page Setup for details).

See Also
Active Notebook

Notebook Palette Property

The notebook Palette property determines the palette of colors displayed in the block and page properties that set colors, such as the block Text Color property and the page Line Drawing property. The options are:

Notebook Palette	Each color square controls the color displayed in the same location in the corresponding palettes.
Edit Color	Opens a dialog box for modifying the active color square. Select a color from the larger palette of available colors, and/or use the HSB, RGB, and CMY color models to create the exact color you want. See Creating Custom Colors for details on using the color models.
Reset Defaults	Returns the notebook palette to the default Quattro Pro settings.

Caution: Changing any color square in the notebook palette also changes any previous uses of that color square throughout the notebook. For example, if you set the Shading property for a cell with the color square that's initially set to red (the leftmost square in the second row of the palette) and later change the corresponding color square in the notebook palette to green, the shaded cell switches to green.

See Also
Active Notebook

Notebook Display Property

To allow more space in a notebook window, or to create a special presentation notebook, you may want to temporarily hide the scroll bars or the page tabs. Uncheck the display elements you want to hide; check them when you want them displayed again.

See Also
Active Notebook

Macro Library Property

A macro library is a convenient place to store frequently used macros. When you set a notebook's Macro Library property to Yes, Quattro Pro searches it when you try to run a macro that isn't in the active notebook.

See Also
Active Notebook

Password Level Property

Sets the level of password protection for the active notebook.

See Also
Assigning a Password to a File
Active Notebook

System Property

A system notebook, when hidden, stays open even when users choose File|Close All. Application developers can use system notebooks to hold application macros and ensure their availability while Quattro Pro is running.

When you set a notebook's System property to Yes, you make it a system notebook. You can use this property with Window|Hide or the Password Level property, depending on the level of protection you want to apply (Medium level hides the notebook when you save and close it). If you're using a system notebook for macros, you'll probably want to apply the Macro Library property too.

See Also

Creating System Notebooks

Window|Hide

Assigning a Password to a File

Using Macro Libraries

Active Notebook

Numeric Format Property

The block Numeric Format property controls the displayed appearance of values in the selected cell or block. Ordinarily, the default Numeric Format is General, which displays numbers exactly as you enter them (unless the column width is too narrow). Click Numeric Format Options for descriptions and examples of each numeric format.

Note: All numeric formats leave the cell values intact; they only affect the way values are displayed.

If the format you choose creates a number too wide for its column, a string of asterisks appears. You can redisplay the number by widening the column.

The format you assign to a cell stays with the cell, even if you delete its contents. If you move the contents, however, the format moves with the data and is removed from the original cell. If you copy a formatted cell, the copy takes on the format of the original cell. If you paste a copied or cut block with Edit|Paste Special, you can control whether the cell's properties are pasted.

See Also

Active Block

Using Styles

Using Paste|Special

Column Width

Numeric Format Options

Format	Description	Examples
Fixed	Displays no more than the specified number of decimal places.	46.1 or 0.56 or -34.123
Scientific	Uses scientific notation. Allows only one digit in the integer portion of the number.	2.35E+2 or 4.76E+9
Currency	Displays numbers as currency, using currency symbols specified with the application International property	$3,467.00 or 35 or ($56.24)
Comma	Separates thousands with commas, and shows negative numbers in parentheses.	15,120.25 or (2,456)
General	Displays numbers as entered, unless they're too long to fit the active cell. Then, they're rounded (if fractional) or translated into scientific notation.	456.9452 or -365 or 0.41 or 1.955E+6
+/-	Transforms values into a horizontal bar graph. Each integer translates into a symbol: **+** for each positive integer, **-** for each negative integer, and **.** for zero.	**+ + +** **- - - -** **. .**
Percent	Displays numbers as percentages.	13.40% or -56.44%
Date	Displays numbers as dates in the format you choose:	
	DD-MMM-YY	09-Apr-92
	DD-MMM	09-Apr
	MMM-YY	Apr-92
	Long International	04/09/92
	Short International	04/09
Time	Displays numbers as times in the format you choose:	
	HH:MM:SS AM/PM	11:31:28PM
	HH:MM AM/PM	05:15AM
	Long International	06:56:10
	Short International	06:56
Text	Displays formulas instead of their results. Displays numbers entered in the block in General format. This is also known as the Show	+B6*C3 or @SUM(B1..B10)

	Formulas format.
Hidden	Suppresses display of both value and label entries. Entries still appear on the input line when the cell is selected.
User Defined	Lets you choose from a list of formats you create. See Defining Custom Numeric Formats for details.

If the format you chose allows a variable number of decimal places, an edit field appears. If you want to display other than the default number of decimal places (2), enter a number from 0 to 15 in the edit field.

If you've hidden cell contents with the Hidden numeric format, you may want to use the page Protection property to prevent the cells' entries from being accidentally overwritten (see Protection for more information).

Formatting Date Serial Numbers

Quattro Pro stores each date you enter (either with Ctrl+Shift+D or a date @function) as a five-digit serial number. Quattro Pro uses this date number for calculations. If you enter the date with Ctrl+Shift+D, the cell is switched to the date format in which you enter it. If you enter the date using @DATE, @DATEVALUE, or @NOW, you must reformat the cells with the Date numeric format. Otherwise, Quattro Pro displays the results as a date serial number.

Precision of Numbers

Numeric formatting doesn't affect the way Quattro Pro stores values, only the way it displays them. For example, some formats limit the number of that appear. This does not affect the precision of Quattro Pro calculations, which are accurate to 16 significant digits. The stored number is displayed on the input line when you select a cell.

A significant digit is any digit that is not a leading zero; decimals, commas, dollar signs, and percent signs do not count as part of the total. For example, the number 1.23 has three significant digits. The number .000123 also has three significant digits because the leading zeros do not count.

Quattro Pro rounds the appearance of numbers to the specified decimal place in the case of Fixed, Scientific, Currency, Comma, and Percent formats. The +/- format truncates decimal numbers to whole integers. The General format rounds off fractional numbers as necessary to fit in the cell.

To round the number actually stored in a cell, use @ROUND (see Function Index for details).

See Also

Using Named Blocks

International

Using Styles

Block Font Property

The block Font property controls the appearance of entries in the active block. Options are:

Typeface	The list of typefaces available vary depending on the font-scaling software running on your system. If a font has a **TT** (TrueType) or an **a** (ATM) symbol beside it, text in that font will appear in print just as it does onscreen. Fonts with a printer symbol will print on your printer, but may not appear in the correct font onscreen. Fonts without a symbol appear accurately onscreen, but may not appear in correct font when printed.
Point Size	Type a point size, or choose one from the list.
Options	Check Bold, Italic, Underline, and/or Strikeout.

If you enlarge or reduce the font size, the row height changes to display the tallest letters in the row (unless you've set the row height explicitly; see Row Height for more information).

See Also

Active Block

Using Styles

Defining Styles

Shading Property

The Shading property controls the background color of cells (the color of entries themselves is controlled by the block Text Color property). The shading color is a mixture of two colors: Color 1 and Color 2. The options are:

Color 1	The first color to be mixed.
Color 2	The second color to be mixed.
Blend	Choose the Blend square that provides the mix you want. The square at the far left is the same as the Color 1 selection; the Color 2 selection appears in the far right square.

With most color choices, you need to deselect the block to see the new shading. This is because selected cells are shown in reverse color.

To change the colors available, use the notebook Palette property (see Palette for details).

Note: If shading doesn't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

See Also

Active Block

Text Color

Using Styles

Defining Styles

Block Alignment Property

The Alignment property controls the placement of entries within cells. The options are:

General	Right-aligns values. Also, aligns labels according to the setting of the page Label Alignment property (see Label Alignment for details), which is set to left-alignment by default.
Left	Left-aligns all types of entries.
Right	Right-aligns all types of entries.
Center	Centers all types of entries.
Center Across Block	Centers text across the selected block.

There are three other ways to change alignment of entries. You can

 use the Left, Center, Right, or Block Center alignment buttons in the SpeedBar.

 precede an individual label with a label-prefix character (see Aligning Labels for more information).

 set the alignment of all labels later entered in the page with the page Label Alignment property (see Label Alignment for details).

See Also

Active Block

Aligning Cell Entries

Line Drawing Property

The Line Drawing property draw lines around cells in a block. The options are:

Line Types	Selects a line type. The No Line choice removes previously-drawn lines, and the No Change choice cancels a change you make to a line.
Line Segments	Applies the selected line type to the lines you indicate. Click wedges outside the example box to indicate lines.
All	Applies the selected line type around every cell in the block (not available when defining styles).
Outline	Applies the selected line type around the block but not between cells.
Inside	Applies the selected line type between cells but not around the perimeter of the block (not available when defining styles).

You can combine line drawing options in the same block. For example, you can draw a double-lined border around a block, and single vertical lines between the cells in the block.

To change the screen color of lines, change the page Line Color property (see Line Color for details).

See Also

Drawing Lines Around Blocks

Active Block

Using Styles

Defining Styles

Block Protection Property

The block Protection property turns off protection in the active block. This property has an effect only if you've already turned on overall page protection with the page Protection property (see Enabling Page Protection for details). The options are:

Protect	The default setting. Choose this to reprotect a block that you've previously unprotected with this property.
Unprotect	Removes protection from the active block.

When a cell is protected, you can't edit, replace, or delete its contents. Nor can you delete a column or row that contains a protected cell.

With page protection enabled, you can move the selector around the entire page, but you can make changes only to unprotected cells. To restrict the selector to unprotected cells only, use Data|Restrict Input (see Restrict Input for details).

See Also

Active Block

Using Styles

Using Named Blocks

Defining Styles

Text Color Property

The Text Color property controls the color of cell entries This property controls the color of the characters that make up the data, not the color of the cell shading.

Color Palette — Choose the color you want. If you already changed the cell's shading, make sure the Text Color setting will contrast enough to be visible.

Note: If text doesn't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

To change the colors available, use the notebook Palette property (see Palette for details). If you create a new color for the palette that is dithered and later choose it for the Text Color property, your text will appear in a substitute, nondithered color. You can use dithered colors for other elements besides text and drawn lines.

Tip: With most color choices, you need to deselect the block to see the new shading. This is because selected cells are shown in reverse color.

See Also

Shading

Active Block

Using Styles

Data Entry Input Property

The Data Entry Input property limits the types of entries allowed in the active block. The options are:

General	The default setting. All entries will be allowed in the block.
Labels Only	Only label entries will be allowed. If someone tries to enter a date, time, or number, it's converted to a label.
Dates Only	Only date or time entries will be allowed. If someone tries to type a label or number, an error message appears.

This feature is especially helpful if you're setting up a notebook for others to use. For example, presetting cells for label-only entry makes it easier to enter phone numbers and social security numbers because they contain hyphens that are otherwise interpreted as minus signs.

For the list of acceptable date and time formats, see Entering Dates and Times.

See Also

Active Block

Search Commands

Database Commands

Search Procedure

Using Styles

Row Height Property

The Row Height property adjusts the height of single or multiple rows to an exact size. Select any cell in one or more rows, or select the row borders. The options are:

Row Height	Shows the current height of the active row. Type a new height here.
Reset Height	Resets the active row's height to the default row height (as determined by the largest font used in the row)
Set Height	Changes the active row's height to the Row Height entry.
Unit	Specifies that the Row Height entry be evaluated in Points, Inches, or Centimeters

See Also

Active Block

Resizing Rows and Columns

Column Width Property

Using Named Blocks

Using Styles

Column Width Property

The Column Width property adjusts the width of single or multiple columns to an exact size. Select any cell in one or more columns, or select the column borders. The options are:

Column Width	Shows the current width of the active column. Type a new width here.
Extra Characters	If Auto Width is chosen, type the number of extra characters beyond the widest entry to allow for column width.
Set Width	Changes the active column's width to the Column Width entry.
Reset Width	Resets the active column's width to the default column width (see Default Width for details)
Auto Width	Changes the active column's width to the longest entry in the column plus the Extra Characters entry. See below for details.
Unit	Specifies that the Column Width entry be evaluated in Characters, Inches, or Centimeters

The Auto Width option resizes contiguous columns based on the longest entry. The longest entry is chosen based on your initial column or block selection:


If entire columns are selected, the width is based on the longest entry in each column.


If a multi-row block (or just part of a column) is selected, the width in each column is based on the longest entry in each column of the block.


If a single-row block (or just one cell) is selected, the width is based on the longest entry in that row and all cells below it.

Tip: If a column you're adjusting contains a long entry that spills over into blank cells to the right, and you don't want the column adjusted to that cell entry's length, specify a multi-row block that stops short of the cell.

See Also

Active Block

Resizing Rows and Columns

Row Height Property

Reveal/Hide Property

The Reveal/Hide property hides rows or columns from view without losing the data they contain. You can later redisplay the rows or columns with the same property. The options are:

Dimension	Choose Rows to operate on active rows; choose Columns to operate on active columns.
Hide	Hides the active rows or columns.
Reveal	Redisplays hidden rows or columns surrounded by the active rows or columns.
No Change	The default setting for the Operation option.

After you use Hide, columns to the right of the hidden columns move left to fill in the empty space, or rows below the hidden rows move up. However, the identifying row numbers and column letters in the borders don't change. In other words, if you hide column B, the columns onscreen are labeled A, C, D, and so on.

If you use Tools|Extract to save part of a notebook that includes hidden rows or columns (see Extracting Part of a Notebook for details), Quattro Pro saves the hidden rows or columns in the new file, although they will still be hidden from view when you load the file.

See Also
Active Block

Page Name Property

The page Name property controls the name of the active page. The name appears on the page's tab and in formulas referring to cells on the page. Pages are initially named with letters of the alphabet in sequence, from A to Z, continuing from AA to AZ, up to IV. The options are:

Page Name	Enter a name up to 15 characters long. See below for details.
Reset	Restores the page name back to its original letter name.

You can use letters and numbers in the name, as well as the following special characters

~ ` ! % _ | \ ' ?

You can't use spaces or any other special characters. Also, you can't use a name you've previously assigned to a group in the same notebook.

Note: The Graphs page (the last page in the notebook) cannot be renamed.

See Also

Using Blocks in Formulas

Active Page

Page Protection Property

The page Protection property turns on protection in the active page. It works in tandem with the block Protection property (see Removing Block Protection for details). Use page protection to set up protection for a page, then use block protection to unprotect the specific blocks of cells you want to allow to be changed.The options are:

Disable	The default setting. Choose this option to turn off protection you've previously enabled with this property. All blocks in the page will be unprotected without regard to their individual block Protection settings.
Enable	Protects all entries in the active page, except those specific unprotected with the block Protection property.

To prevent unauthorized access to an entire notebook, protect the it with a password (see Assigning a Password to a File for details).

See Also
Active Page

Line Color Property

The Line Color property controls the color of lines drawn on the active page with the block Line Drawing property (see Line Drawing for details) By default, the lines are black.

Color Palette	Choose the color you want. If lines don't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

To change the colors available, use the notebook Palette property (see Palette for details). If you create a new color for the palette that is dithered and later choose it for the Line Color property, your lines will appear in a substitute, nondithered color. You can use dithered colors for other elements besides text and drawn lines.

See Also
Active Page

Conditional Color Property

The page Conditional Color property changes the color of specific types of data in the active page: values above or below a specified range, and ERR values. The options are:

Smallest Normal Value	(Along with Greatest Normal Value) lets you enter a range of values you consider normal. Cells with values within this range will appear in the Normal Color. The default settings are 0 (smallest) and 1E+300 (largest). You can specify different colors for values above and below this range by clicking the option and clicking a color square in the color palette.
Below Normal Color	Sets the color of cells whose values are below the Smallest Normal Value.
Normal Color	Sets the color of cells whose values fall within the range set by Smallest Normal Value and Greatest Normal Value.
Above Normal Color	Sets the color of cells whose values are above the Greatest Normal Value.
ERR Color	Specifies the color to use for ERR and NA values generated by formula errors.
Enable	Indicates whether to use the colors set with this menu.

To change the colors available, use the notebook Palette property (see Palette for details).

Note: If colored entries don't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

See Also
Active Page

Label Alignment Property

The Label Alignment property setting sets the default alignment of labels later entered in the active page.The options are:

Left	The default setting: left-aligns labels in cells.
Right	Right-aligns labels in cells.
Center	Centers labels in cells.

Label entries already existing in the page don't adjust. To change the alignment of existing data, use the Alignment buttons in the SpeedBar or the block Alignment property. All new labels will be aligned according to this new default unless you precede them with a different label-prefix character (see Aligning Labels for details). Values (including string values that result from some @functions) are not affected by label alignment settings.

The setting you make here determines the effect on labels of the General setting of the block Alignment property.

See Also

Block Alignment

Aligning Labels

Aligning Cell Entries

Active Page

Display Zeros Property

The Display Zeros property suppresses the display of any value in the active page that equals exactly zero, whether it was entered directly or calculated with a formula. The options are:

Yes	The default setting: displays zeros.
No	Hides zeros.

Zero suppression doesn't remove the zero values from the page. They remain in memory and reappear if you set Display Zeros to Yes.

A value must equal exactly zero to be suppressed. For example, a value such as .004 that appears as 0 if decimal precision is 2 or less, will still appear if Display Zeros is set to No, because it is not actually 0.

Caution: When zero suppression is on, it's easy to accidentally write over cells containing formulas that evaluate as zero. Make sure Undo is enabled (see Enabling the Undo Command for more information), or consider protecting the page (see Enabling Page Protection for details).

See Also
Active Page

Default Width Property

The Default Width property sets the default width of all columns in the active page. By default, columns are wide enough to display approximately nine characters in the default font. The options are:

Column Width	The default column width in characters, inches, or centimeters.
Unit	The measurement system for the Column Width setting.

The Default Width setting doesn't affect columns that were explicitly adjusted using the Fit button, the block Column Width property, or the mouse. Before those columns can be affected by a change in the default width, you must select them and check the Reset Width option of the block Column Width property.

See Also

Setting Automatic Row & Column Sizes

Column Width

Resizing Rows & Columns with the Mouse

Active Page

Page Borders Property

The Borders property turns off row and column borders in the active page. The options are checked by default:

Row Borders	Uncheck to turn off the numbered row borders at the left of the page.
Column Borders	Uncheck to turn off lettered column borders at the top of the page.

This property is useful for setting up a page as a form for data input.

See Also
Active Page

Grid Lines Property

The Grid Lines property turns off the spreadsheet grid in the active page.The options are checked by default:

Horizontal	Uncheck to turn off grid lines that separate rows.
Vertical	Uncheck to turn off grid lines that separate columns.

This property is useful for viewing a page as it would look when printed (grid lines aren't printed by default).

See Also
Active Page

Tab Color Property

The Tab Color property changes the tab color of the active page. The default tab color of the active page is white.

See Also
Creating Custom Colors
Active Page

Macro Property

Specifies a macro for the active SpeedButton to run. See Creating Macro SpeedButtons for information on the Execute Dialog button.

See Also
SpeedButton

Border Color Property

Border Color specifies a color for the border of the active floating object. The palette of colors used is determined by the Notebook Palette property.

See Also
Box Type

Box Type Property

Box Type specifies the appearance of the border surrounding the active floating object. The options are:

None	no border at all
Thin/Medium/Thick	increasingly thick borders
Drop Shadow	a black offset border that gives the object a 3-D appearance

See Also
Creating Floating Objects

Object Name Property

Object Name specifies a name for the active floating object. This name can be used in macros that work with this object.

See Also
What Is a Macro?

Default Effect Property

Specifies a default slide show effect for the active slide show icon.

See Also
Creating a Slide Show

Link Settings Property

Controls the behavior of the active linked OLE object. The options are:

Update Method	Choose Automatic to make data always reflect edits in the server application. Choose Manual to update data only when Update Now is chosen.
Edit	See the previous explanation for embedded OLE objects.
Update Now	Choose this when Manual update method is chosen to copy the latest data from the server application.
Unlink	Choose this to remove the link from the server application. This converts the OLE object to a picture object, which you can no longer edit.
Change Link	If the file to which you are linked is moved to different directory, choose this to relink to the file in its new location.

See Also
Choosing the Data Type to Paste

Name Property

Specifies a name for the active graph, dialog box, or slide show icon in the Graphs page, or the active dialog control in the dialog window. This name can be used in macros that work with these icons or the objects they represent.

See Also
What Is a Macro?

Object Settings Property

Controls the behavior of the active embedded OLE object. The options are:

Change to Picture	Use this to convert an embedded OLE object to a picture object. Once this has been done, you will no longer be able to edit the object.
Edit, Play	(Or whatever action is appropriate to the server application you start up.) You can run a video, play a sound, edit the object, and so on. When you are finished working in the server application, choose File|Update or File|Exit.

See Also
Choosing the Data Type to Paste

Source Graph Property

The Source Graph property determines which graph is displayed in the active floating graph. object.

Select Graph displays a list of graphs in the active notebook. The new floating graph will have the same dimensions as the original floating graph.

See Also
Creating a Floating Graph

Application Properties

You can display the application properties by right-clicking the Quattro Pro title bar or by choosing the Application command in the Property menu. Properties in the application Object Inspector affect the overall functioning of Quattro Pro. Any changes you make to these properties remain in effect until you change them again, even after you exit and restart Quattro Pro.

Display	Sets clock display, range syntax, menu style, and screen repaint during macros.
International	Sets currency, punctuation, date, and time formats. Also determines sort table used, and provides LICS conversion.
Startup	Sets startup directory, autoload file, startup macro, file extension, beep alert, and key compatibility, and enables the Undo command.
Macro	Controls the behavior of macros.
SpeedBars	Lets you save a SpeedBar arrangement when you exit.
Delay Time	Specifies drag-and-drop delay time.

Note: If you start Quattro Pro in Developer mode (with the /D parameter), two additional properties are available:

Title	Displays a custom title in the Quattro Pro title bar.
Enable Inspection	Globally enables/disables mouse right-clicking.

See Also
Active Notebook Properties

Application Display Property

The Display property sets display characteristics that affect the entire Quattro Pro application. The options are:

Clock Display	Displays the date and time on the status line in your choice of formats. Standard displays the date and time as DD-MMM-YY and HH:MM AM/PM; International displays the date and time in the Long International formats specified with the International property. If the time or date displayed is incorrect, you must update your computer's internal clock. You can do this with the Windows Control Panel.
Display Options	Displays or hides the SpeedBar, the input line, or the status line.
3-D Syntax	Switches between two page notation methods for 3-D blocks. The default method expresses page references first, followed by a colon and the block coordinates. For example, A..C:B4..D9 refers to the block B4..D9 on pages A, B, and C. The alternate syntax refers to the same block by referring to each corner of the block with the page reference included, as in A:B4..C:D9.

See Also

International Property

Switching the Page Range Syntax

Application International Property

The International property controls the appearance of currency, punctuation, dates, times, and international characters. It also lets you choose the language to govern sort orders. The initial defaults are standard for the United States. If you are using Quattro Pro in another country, or are doing business with another country, you can change these settings to suit your requirements. The options are:

Currency	Sets the currency symbol ($ is the default) used in the Numeric Format property's Currency setting and in the Currency styles. Also controls whether the currency symbol appears before the formatted values (Prefix) or after them (Suffix), and whether negative values are preceded with a minus sign (Signed) or are surrounded with parentheses (Parentheses).
Punctuation	Controls the characters used as the thousands separator, the decimal separator, and as argument separators. The options first show the punctuation marks used to mark thousands and the decimal place, followed by the punctuation mark used to separate arguments in @functions and macros (a1,a2). The last four options specify that a blank space separates thousands in numbers.
Date Format	Determines the international formats given as options for date display from the block Numeric Format property. Each of these options is shown in its long and short versions.
Time Format	Determines the international formats given as options for time display from the block Numeric Format property. Each of these options is shown in its long and short versions.
Language	Sets the language to determine Quattro Pro's sort order.
LICS	Converts Lotus International Character Set characters into standard ANSI characters. LICS characters are identical to the standard ANSI set except in the range 128 through 255, which is usually used for international and graphics characters. When you save the file, Quattro Pro writes these characters back to the LICS equivalents. For more on LICS characters, see the 1-2-3 documentation.

Each Windows Default setting available with these options uses the corresponding setting in the Windows Control Panel. See your Windows documentation for details.

See Also

Choosing a Currency Symbol

Controlling Punctuation

Setting Date Formats

Setting Time Formats

Block Numeric Format

Changing the Sort Order

Application Startup Property

The Startup property includes information that is used every time you start Quattro Pro. The options are:

Directory	The directory initially displayed by file-handling commands.
Autoload File	The file to be loaded every time Quattro Pro is started.
Startup Macro	The macro to be run every time any notebook containing a macro with this name is opened. See Running a Macro Automatically for details.
File Extension	The default file extension to be used when you don't specify a file extension with file-handling commands.
Use Beep	Makes Quattro Pro beep when you make errors.
Undo Enabled	Makes the Edit|Undo command available after the full range of undoable operations. See Enabling the Undo Command for details.
Compatible Keys	Makes a variety of keys work in the same way as in Quattro Pro for DOS. When unchecked, these keys work as in most Windows applications. See Changing Key Compatibility for details.
Move Cell Selector on Enter Key	With Compatible Keys unchecked, checking this setting makes the selector move down a cell each time you enter data. For details, see Setting Cell Selector Behavior.

See Also

Setting the Startup Directory

File Handling Options

Application Macro Property

The Macro property lets you control macros and alternate menu systems. The options are:

Macro Suppress-Redraw	Controls which parts of the screen redraw while macros are running. Panel prevents redrawing of menus, dialog boxes, the status line, and the input line. Window prevents redrawing of windows only. Both suppresses redrawing of both windows and panels. Suppressing redrawing speeds up macros, since Quattro Pro doesn't have to stop and draw each of these items used in the macro.
Slash Key	Controls which menu system displays when you press the slash key (/) from Ready mode.

See Also

Setting Macro and Menu Options

DOS Spreadsheet Users

Macros

Application Building.

Application SpeedBars Property

The SpeedBars property lets you automatically save the current SpeedBar arrangement when you exit Quattro Pro. When you restart Quattro Pro, the arrangement will be restored.

See Also

Selecting Secondary SpeedBars

Application Delay Time Property

The Application Delay Time property specifies how long to wait, in milliseconds, before changing from block selection to Drag-and-Drop mode.

See Also

Setting Drag and Drop Delay Time

Dragging and Dropping

Application Title Property

The Application Title property changes the title displayed on Quattro Pro's title bar. This property is only available after starting Quattro Pro in developer mode (with /D parameter).

See Also
Application Building

Enable Inspection Property

The Enable Inspection property globally enables/disables mouse right-clicking This property is only available after starting Quattro Pro in developer mode (with /D parameter).

See Also
Application Building

Dialog Window Objects

Dialog window objects are controls used in building applications. If you have not yet selected an object and choose the Property|Current Object command, the properties that control the Dialog window appear. See Active Dialog Window Properties for details.

Bitmap Button

Check Box

Color Control

Combo Box

Edit Field

Edit Integer

File Control

Group Box

Horizontal ScrollBar

Label

List Box

Pick List

Push Button

Radio Button

Rectangle

ScrollBar

Spin Control

Time Control

See Also

Properties in the Dialog Window

Active Dialog Window Properties

The following properties control the behavior of the dialog window:

Dimension	Specifies the exact size and position of the dialog window.
Title	Specifies the title of the dialog box, which the user will see.
Position Adjust	Specifies how the dialog box moves when the Quattro Pro window is resized.
Grid Options	Shows/hides a grid and sets the number of pixels between grid lines.
Name	Used to identify the dialog box in macros, link commands, and formulas.
Disabled	Disables all controls in the dialog box.
Value	The current setting of the dialog box (a comma separated list of the current control settings of the dialog box).

See Also

Properties in the Dialog Window

Bitmap Button Control

A bitmap button has a graphic image on its face instead of text. It can imitate the behavior of a check box, radio button, or push button. There are several different ways to specify the bitmap that appears on a bitmap button's face:

Right-click the control and choose the Properties command, then choose Bitmap. You'll see a list of predefined bitmaps. Select a bitmap from the list (it then appears under Preview) and choose OK.

Choose Browse from the Bitmap property dialog box, specify the name of an external Windows bitmap (BMP) file, and choose OK. The bitmap will be saved with the active notebook.

If the Clipboard contains a bitmap, select the bitmap button and choose Edit|Paste to paste the bitmap onto the button's face. The bitmap will be saved with the active notebook.

Bitmap buttons have the value of whatever button they're imitating. To specify how the button behaves, set the bitmap button's Button Type property.

Bitmap Button Properties

Bitmap	Specifies a bitmap file (.bmp) or an internal Quattro Pro bitmap to appear on a button.
Label Text	Specifies the text appearing on the button.
Text Draw Flags	Defines the location of label text on the bitmap button.
Default Button	Specifies the default button in the dialog box (which button is executed by pressing Enter).
Button Type	Specifies the type of button.
Dimension	Specifies the exact size and position of the bitmap button.
Hidden	Specifies whether the bitmap button is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	Specifies the bitmap button's identification number.
Position Adjust	Specifies how an attached element moves when the bitmap button is resized.
Name	Used to identify the bitmap button in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the bitmap button can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the bitmap button appears when the dialog box displays.
Tab Stop	Determines whether the bitmap button can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the bitmap button.

See Also

Push Button
Working with Controls

Push Button Control

A push button runs a specified command when the user clicks it. To have a macro run when the user clicks a push button, add a DOMACRO link command to the push button.

Push buttons don't have a value. To make a push button close the dialog box, use the Button Type property.

Push Button Properties

Label Text	Specifies the text appearing on the button.
Text Draw Flags	Defines the location of label text on a button.
Default Button	Specifies the default button in the dialog box (which button is executed by pressing Enter).
Button Type	Specifies the type of button.
Dimension	Specifies the exact size and position of the button.
Hidden	Specifies whether the button is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The button's identification number.
Position Adjust	Specifies how an attached element moves when the button is resized.
Name	Used to identify the button in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the button can be used. The control appears dimmer to indicate that it's unavailable.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the button appears when the dialog box displays.
Tab Stop	Determines whether the button can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the button.

See Also

Bitmap Button

Working with Controls

Check Box Control

Check boxes present a Yes/No option to the user. The text that appears in this button is stored in its Label Text property. The value of a check box is Yes when it's checked, No when it's unchecked.

Check Box Properties

Draw to Right	Controls the position of check box text.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the check box's value.
Label Text	Specifies the text that appears by the check box.
Text Draw Flags	Defines the location of label text adjacent the check box.
Button Type	Specifies the type of button in the check box.
Dimension	Specifies the exact size and position of the check box.
Hidden	Specifies whether the check box is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The check box's identification number.
Position Adjust	Specifies how an attached element moves when the check box is resized.
Name	Used to identify the check box in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the check box can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the check box appears when the dialog box displays.
Tab Stop	Determines whether the check box can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the check box.

See Also

Working with Controls

Color Control

A color control lets the user change a color setting. Its value is in the format Red,Green,Blue where Red is the amount of red in the color, Green the amount of green, etc. Each amount is a number from 0 to 255. Each part of a color control's value is a number between 0 and 255. For example, black is 0,0,0; white is 255,255,255; and red is 255,0,0. The current color setting appears in the large rectangle on the right side of the control.

The user can click HSB or CMY (on the color control) to specify the color using a different color model (see Selecting Fill, Background, and Border Colors for more information on color models). The value of the color control doesn't change when a different color model is used to specify the color; the control value is always in Red,Green,Blue format.

Color controls are large; make sure the dialog window is large enough to accommodate a color control before you add one.

Color Control Properties

Dimension	Specifies the exact size and position of the color control.
Hidden	Specifies whether the color control is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The color control's identification number.
Position Adjust	Specifies how an attached element moves when the color control is resized.
Name	Used to identify the color control in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the color control can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the color control's value.
Tab Stop	Determines whether the color control can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the color control.

See Also

Working with Controls

Combo Box Control

Combo boxes (also called drop-down list boxes) are a special type of list box that lets the user add items to the list. They consist of an edit field, a down arrow button, and a list box (which only appears when the user clicks the down arrow or presses Down in the edit field.)

The value of the combo box is the value of its edit field. There are several ways users can change a combo box's value:


They can choose the edit field and enter the value. (You can disable this by setting Edit Disabled to Yes.)


They can press the Down key to move through the list, then press Enter on the desired choice. (Pressing Enter also clicks the dialog box's default button, which typically closes the dialog box. To disable this, set Terminate Dialog to No.)


They can click the down arrow button to display the list, then click the desired choice. (To remove a combo box's down arrow button, set Add Down Button to No.)

There are two ways you can specify the items that appear in a list box:


Using the Clipboard. Copy a block to the Clipboard, then select the list box and choose Edit| Paste. Each row of the block becomes one item in the list.


Using the List property. If your list is already in a block in the notebook, right-click the list box and choose the Properties command, then choose List, type or point to the block coordinates, and choose OK.

You can make a combo box store a history list of the entries that have been typed into its edit field. This history displays as the combo box's list.

Combo Box Properties

List	Specifies a block containing a list to appear in the combo box.
Allow Point Mode	Lets the pointer be used to select a block while the dialog box remains in the foreground.
Add Down Button	Adds a down arrow to the right of a combo box.
History List	Adds confirmed entries to the list in the combo box.
List Length	Specifies the maximum number of history list entries.
Terminate Dialog	If set to Yes, pressing Enter triggers the default button (usually Enter or Exit).
Edit Disabled	Disables typing in the edit field.
Edit Length	Specifies the maximum number of characters in the edit field or spin control.
Ordered	Sorts entries alphanumerically.
Dimension	Specifies the exact size and position of the combo box.
Hidden	Specifies whether the combo box is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The combo box's identification number.

Position Adjust	Specifies how an attached element moves when the combo box is resized.
Name	Used to identify the combo box in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the combo box can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the combo box appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the combo box's value.
Tab Stop	Determines whether the combo box can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the combo box.

See Also
Working with Controls

Edit Field Control

An edit field is where a user types specific information. You can use the Field Type property to restrict the type of information an edit field accepts.

Field type	Result
Integer	Restricts user entry to integers. This adds three properties to the edit field Object Inspector (which is renamed Edit Integer). Minimum defines the lowest acceptable value users can enter. Maximum defines the highest acceptable value a user can enter. Default defines the initial value of the edit field.
String	Accepts any text a user enters (including numbers); this is the default edit field type.
Real	Accepts any number or formula.
Block	Accepts only cell addresses or block coordinates.
Hidden	Accepts any text, but displays a pound sign (#) for each character instead of the actual character. This is handy when users are entering passwords.

Edit Field Properties

Field Type	Determines the allowable type of input to the edit field (see above).
Edit Length	Specifies the maximum number of characters in the edit field.
Allow Point Mode	Lets the pointer be used to select a block while a dialog box is displayed.
Show Frame	Displays a box around the field.
Terminate Dialog	If set to Yes, pressing Enter triggers the default button (usually Enter or Exit).
Convert Text	Allows insertion of a new line and tabs in an edit field.
Dimension	Specifies the exact size and position of the edit field.
Hidden	Specifies whether the edit field is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The edit field's identification number.
Position Adjust	Specifies how an attached element moves when the edit field is resized.
Name	Used to identify the edit field in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the edit field can be used. The control appears dimmer to indicate it's unavailable.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the edit field appears when the dialog box is displayed.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the edit field's value.
Tab Stop	Determines whether the edit field can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the edit field.

See Also
Working with Controls

Edit Integer Control

The Edit Integer control is an edit field that only accepts integers. You create this control by placing an edit field in your dialog box, and setting the Field Type property to Integer.

Edit Integer Properties

Minimum	Determines the minimum number accepted in the edit field.
Maximum	Determines the maximum number accepted in the edit field.
Default	Determines the default text displayed in the edit field.
Field Type	Determines the type of input that the edit field accepts.
Edit Length	Specifies the maximum number of characters in the edit field.
Show Frame	Displays a box around the field.
Terminate Dialog	If set to Yes, pressing Enter triggers the default button (usually Enter or Exit).
Dimension	Specifies the exact size and position of the edit field.
Hidden	Specifies whether the edit field is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The edit field's identification number.
Position Adjust	Specifies how an attached element moves when the edit field is resized.
Name	Used to identify the edit field.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the edit field can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the edit field appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the edit field's value.
Tab Stop	Determines whether the edit field can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the edit field.

See Also

Working with Controls

File Control

A file control button displays a list of files on disk. From this list, the user can choose a file name, directory, and drive. The value of a file control determines the directory, drive letter, and file name (or wildcard) that appears. For example, the value C:\FILES*.WB1 means

Drives is set to C:

Directories is set to C:\FILES\

File Name is set to *.WB1

File Types is set to *.WB1 (if a wildcard is specified; otherwise File Types isn't affected by the value)

Whenever a user chooses a different directory to view, or a different file name or wildcard to use, the change is reflected in the file control's value.

File controls are large; make sure the dialog window is large enough to accommodate a file control before you add one.

File Control Properties

Dimension	Specifies the exact size and position of the file control.
Hidden	Specifies whether the file control is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The file control's identification number.
Attach Child	Used to group file controls for moving or resizing.
Position Adjust	Specifies how an attached element moves when the file control is resized.
Name	Used to identify the file control in macros, link commands, and formulas.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the file control can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the file control appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the file control's value.
Tab Stop	Determines whether the file control can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the file control.

See Also

Working with Controls

Group Box Control

A group box usually contains other controls such as radio buttons or check boxes. It looks like a rectangle with a title at the top. When a group box contains radio buttons, its value is the title of the radio button chosen; otherwise, it does not have a value.

Initially, the Process Value of a group box is set to Yes, and the Process Values of radio buttons within the group box are set to No. In addition to the Value property, the group box has a property called Selected that indicates numerically which of its radio buttons is chosen, starting from the top left radio button (zero) and proceeding to the bottom right button.

Group Box Properties

Group Text	Specifies a title for a group box.
Selected	Specifies the number I.D. of a radio button, or a selection in a list box.
Dimension	Specifies the exact size and position of the group box.
Hidden	Specifies whether the group box is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The group box's identification number.
Attach Child	Used to group controls for moving or resizing.
Position Adjust	Specifies how an attached element moves when the group box is resized.
Name	Used to identify the group box.
Disabled	Behaves like Grayed, but doesn't dim the control.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the group box's value.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of a group box.

See Also

Working with Controls

Horizontal ScrollBar Control

Scroll bars let the user pick a value from a fixed range of values. You can use a vertical scroll bar or a horizontal one. The value of a scroll bar is an integer from a fixed range of integers.

Use the Parameters property to restrict the range of values that horizontal and vertical scroll bars accept and specify how manipulating the scroll bar changes its value.

For a complete explanation of scroll bars, scroll boxes, and scroll arrows, see the *Windows User's Guide*.

Horizontal ScrollBar Properties

Parameters	Controls behavior of scroll bars.
Dimension	Specifies the exact size and position of the scroll bar.
Hidden	Specifies whether the scroll bar is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The scroll bar's identification number.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the scroll bar.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the scroll bar can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the scroll bar appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the scroll bar's value.
Tab Stop	Determines whether the scroll bar can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the scroll bar.

See Also

Working with Controls

Label Control

A label is simply descriptive text that you place beside another control, to describe the control's purpose to the user. Labels don't have a value. Resizing a label doesn't increase the text size; it makes a larger background to display the text on.

To see the background of a label, select it. By default, the label text appears in the upper left corner of its background; to specify a different position for the label text, right-click the label and choose the Properties command, then use the Text Draw Flags property. You can also use Text Draw Flags to create a multiple-line label and wordwrap the label text so that it fits in the label background. The options available under Text Draw Flags are:

Option	Description
Apply	Applies the Text Draw Flags properties you've selected.
Wrapping Text	Single Line specifies whether the label converts formatting codes. Uncheck this to allow formatting codes to be entered into the label (discussed next). This also enables the Word Break option. Word Break specifies whether to wrap the text onto a new line in the label background. Check this to enable line wrapping (available only when Single Line is unchecked).
Vertical Position	Top places text flush against the top edge of the label background. Center Vertically centers the text between the top and bottom of the background. It doesn't affect the horizontal position of the text. Bottom places text flush against the bottom of the background.
Horizontal Position	Left places text flush against the left side of the label background. Center Horizontally centers the text between the left and right sides of the label background. It doesn't affect the vertical position of the text. Right places text flush against the right side of the label background.

When the Single Line option (in the Text Draw Flags property) is unchecked, you can enter the following escape sequences:

\n (or \r) makes characters to the right of it display on a second line. This only works when the Single Line option (in the Text Draw Flags property) is unchecked.

\t inserts a tab into the text (a fixed amount of space that's handy for lining up columns of text). This only works when the Single Line option (in the Text Draw Flags property) is unchecked.

\\ inserts a backslash (\) into the text. You can use this to enter \n, \r, or \t into a label as normal text.

& makes characters to the right of it appear underlined in the label text (the ampersand doesn't appear). This is handy for naming controls.

Label Properties

Label Text	Inputs text to appear adjacent to a dialog control.
Label Font	Determines label format.
Text Draw Flags	Defines the location of label text adjacent to a dialog box control.
Dimension	Specifies the exact size and position of the label text.

Hidden	Specifies whether the label is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The label text's identification number.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the label text.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the label text appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the label text's value.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the label text.

See Also
Working with Controls

List Box Control

A list box presents the user with a list of items to choose from. Whichever item the user picks becomes the value of the list box. There are two ways to specify the items that appear in a list box:

Using the Clipboard. Copy a block to the Clipboard, then select the list box and choose Edit| Paste. Each row of the block becomes one item in the list.

Using the List property. If your list is already in a block in the notebook, right-click the list box and choose the Properties command, then choose List, type or point to the block coordinates, and choose OK.

List box properties change the way items appear.

List Box Properties

List	Specifies a pre-existing list to appear in a list box, or a block containing a list.
Ordered	Sorts List box entries alphanumerically.
Number of Columns	Allows multiple columns in a list box.
Selection Text	Returns text of item selected in a list box.
Selected	Specifies the number ID of a selection in a list box.
Dimension	Specifies the exact size and position of the list box.
Hidden	Specifies whether the list box is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The list box's identification number.
Attach Child	Used to group elements for moving or resizing.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the list box.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the list box can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the list box appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the list box's value.
Tab Stop	Determines whether the list box can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of a list box.

See Also

Working with Controls

Pick List Control

A pick list button displays a list of choices when the user points to it and holds down the left mouse button. If the list is empty, the user can double-click the pick list button and type a setting. The value of a pick list is whichever item the user selects (the value is initially set to the first item in the pick list). To create a pick list,

1. Enter the items of the pick list as labels in a block in the notebook, one label per row.
2. Select the block and choose Edit|Copy.
3. Activate the dialog window you want to add the pick list to.
4. Choose the Pick List tool from the dialog SpeedBar and click in the dialog window to create the pick list button.
5. Choose Edit|Paste to create the list. The first item in the list displays on the pick list button's face.
6. Use the Title property, if desired, to specify a title to display initially on the pick list button. This title doesn't have to be an item on the list. You can use a link command to set the Title property whenever the dialog box displays.

Choosing an item from the list changes the value of the list. By default, a pick list button doesn't resize when a user chooses an item too large to display on the button's face. You can set Resize to Yes to make it expand or shrink, depending on the size of the item the user picks.

Pick List Properties

Title	Determines the dialog box title displayed to the user.
Selected	Specifies the number ID of a selection in a pick list.
Resize	Adjusts the width of a pick list to match the length of the current string in the list.
Dimension	Specifies the exact size and position of the pick list.
Hidden	Specifies whether the pick list is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The pick list's identification number.
Attach Child	Used to group elements for moving or resizing.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the pick list.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the pick list can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the pick list appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the pick list's value.
Tab Stop	Determines whether the pick list can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the pick list.

See Also

Working with Controls

Radio Button Control

Radio buttons, when placed in a group box with other radio buttons, present a list of mutually exclusive choices to the user--only one radio button can be chosen at a time. When the user chooses one of the radio buttons, it darkens to indicate that it's chosen.

To automatically fill a group box with radio buttons, paste a block stored in the Clipboard into a group box; one radio button is created for each row of the block. You can also hold down Ctrl and drag the lower right handle of the group box to increase its size. When you release the handle, radio buttons appear within. Repeat this process until the desired number of radio buttons appear.

By default, a radio button's Process Value property is set to No, and the group box containing it has Process Value set to Yes. When its Process Value property is set to Yes, a radio button has a value of Yes when chosen, No when not chosen.

Radio Button Properties

Draw to Right	Controls position of radio button's text.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the radio button's value.
Label Text	Inputs text to appear adjacent to a radio button.
Text Draw Flags	Defines the location of label text adjacent to a dialog box element.
Button Type	Specifies the type of button.
Dimension	Specifies the exact size and position of the radio button.
Hidden	Specifies whether the radio button is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The radio button's identification number.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the radio button.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the radio button can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the radio button appears when the dialog box displays.
Tab Stop	Determines whether the radio button can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the radio button.

See Also

Working with Controls

Rectangle Control

A rectangle can be used to visually organize controls in a dialog box. Rectangles don't have a value. To change how a rectangle appears, right-click it and choose the Properties command, then choose Rectangle Style.

Rectangle Properties

Rectangle Style	Determines the type of rectangle.
Fill Color	Specifies the color inside a rectangle.
Frame Color	Specifies the color of a rectangle frame.
Dimension	Specifies the exact size and position of the rectangle.
Hidden	Specifies whether the rectangle is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The rectangle's identification number.
Attach Child	Used to group rectangles for moving or resizing.
Position Adjust	Specifies how an attached rectangle moves when its parent is resized.
Name	Used to identify the rectangle.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the rectangle appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the rectangle's value.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.

See Also
Working with Controls

ScrollBar Control

Scroll bars let the user pick a value from a fixed range of values. You can use a vertical scroll bar or a horizontal one. The value of a scroll bar is an integer from a fixed range of integers.

Use the Parameters property to restrict the range of values that horizontal and vertical scroll bars accept and specify how manipulating the scroll bar changes its value.

For a complete explanation of scroll bars, scroll boxes, and scroll arrows, see the *Windows User's Guide*.

Vertical ScrollBar Properties

Parameters	Controls the behavior of scroll bars.
Dimension	Specifies the exact size and position of the scroll bar.
Hidden	Specifies whether the scroll bar is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The scroll bar's identification number.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the scroll bar.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the scroll bar can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the element appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the scroll bar's value.
Tab Stop	Determines whether the scroll bar can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the scroll bar control.

See Also

Working with Controls

Spin Control

A spin control is an edit field with two arrows on its edge. The user can either click an arrow to increase or decrease the spin control's value, or type the value. The value of a spin control is always an integer. (You can use edit fields to enter decimal numbers.)

Spin Control Properties

Edit Length	Specifies maximum number of characters in the spin control.
Show Frame	Displays a box around the field.
Minimum	Determines the minimum number of in the edit field.
Maximum	Determines the maximum number in the edit field.
Default	Determines the default text displayed in the edit field.
Dimension	Specifies the exact size and position of the spin control.
Hidden	Specifies whether the spin control is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The spin control's identification number.
Attach Child	Used to group elements for moving or resizing.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the spin control.
Help line	A brief description that appears at the bottom of the Quattro Pro window.
Grayed	Specifies whether the element can be used.
Disabled	Behaves like Grayed, but doesn't dim the control.
Depend On	Specifies the areas of Quattro Pro in which the spin control appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the spin control's value.
Tab Stop	Determines whether the spin control can be activated by Tab.
Object Help	Determines the contents of Object Help for the selected object.
Enabled	The opposite of Disabled. Set it to No to disable the control.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the spin control.

See Also

Working with Controls

Time Control

A time control shows the user the current time. You can also use a time control to run link commands at regular intervals or at a certain time of day. Time controls don't have a value.

You can use time controls to display the current time in a dialog box or SpeedBar. To add a clock,

1. Create a time control in the dialog window.
2. Right-click the time control and choose the Properties command, then set Show Time to Yes. The timer changes into a small digital clock.
3. Right-click the timer and choose the Properties command, then set Timer On to Yes. This enables the clock.

You can use a time control to run link commands at regular intervals or at a certain time of day. This is handy for creating SpeedBar controls that periodically retrieve current values from the active window. There are two ways to configure a time control for generating regular events: as a Timer and as an Alarm.

As a **Timer**, it generates the Timer event at regular intervals (for example, every two seconds). Two properties of the time control are used to specify the amount of milliseconds to wait between Timer events, as shown in the following formula:

Wait = Units in Milliseconds X Interval in Units

Units in Milliseconds is set to 1000 by default, so you can use the Interval in Units property to specify how many seconds to wait between Timer events. To make a timer,

1. Right-click the time control and choose the Properties command, then choose Units in Milliseconds and enter how many milliseconds it takes for one unit to elapse.
2. Right-click the time control and choose the Properties command, then choose Interval in Units and enter how many units to wait before generating a Timer event.
3. Right-click the time control and choose the Properties command, then choose Timer On and set it to Yes.

As an **Alarm**, the time control generates the Alarm event at a certain time of day (for example, at 5:00). The SpeedBar must be open; for example, an alarm set for 3:00 won't generate an Alarm event if the SpeedBar is not onscreen at 3:00. To make a time control an Alarm,

1. Right-click the time control and choose the Properties command, then choose Alarm On and set it to Yes.
2. Right-click the time control and choose the Properties command, then choose Alarm Time and enter the time that the Alarm event should be generated.

A time control can have both of these functions enabled simultaneously. For example, you could have a timer generate Timer events every two seconds and send out an alarm event at 5:00 to signal the end of the work day.

Time Properties

Show Time	Shows time on the face of the time control.
Interval in Units	Determines how frequently the timer refresh event will be generated.
Units in Milliseconds	Controls Timer intervals together with the Interval in Units property.
Current Time	Fetches the current time. You can't set this property; it's read only.
Timer On	Turns the timer on or off.
Alarm Time	Instructs the Timer to generate an alarm event at the specified time of day.
Alarm On	Turns the Timer alarm on or off.

Dimension	Specifies the exact size and position of the time control.
Hidden	Specifies whether the time control is visible in the dialog box. The control is hidden to the user; it's still visible when editing.
Object ID	The time control's identification number.
Attach Child	Used to group time controls for moving or resizing.
Position Adjust	Specifies how an attached element moves when its parent is resized.
Name	Used to identify the time control in macros, link commands, and formulas.
Depend On	Specifies the areas of Quattro Pro in which the time control appears when the dialog box displays.
Process Value	Setting this property to Yes lets the macro command {DODIALOG} access the time control's value.
Object Help	Determines the contents of Object Help for the selected object.
Show	The opposite of Hidden. Set it to No to hide the control from the user.
Value	The current setting of the time control.

See Also
Working with Controls

Graph Window Objects

You can find or create the following objects in a graph window. The graph window is also an object with properties of its own. (See Graph Window Properties for details.)

Drawn Graph Objects (created with tools on the SpeedBar)

Arrows
Ellipses
Freehand Polygons
Freehand Polylines
Lines
Polygons
Polylines
Rectangles
Rounded Rectangles
Text Boxes

Fixed Graph Objects

3-D Walls and Base
Area Series
Axis Titles
Bar Series
Column Graphs
Graph Pane
Graph Setup and Background
Graph Subtitle
Graph Title
Graph Title Box
Legend
Line Series
Pie and Doughnut Graphs
Series Labels
X-Axis
Y-Axis

See Locating Graph Objects to learn where to right-click and display Object Inspectors for fixed graph objects and for the graph window.

Arrows

You use the Arrow tool on the graph window SpeedBar to create a straight line with an arrow at the end. Arrows have these properties:

Fill Color	assigns a fill color to the arrowhead.
Bkg Color	provides a second color for the arrowhead. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the arrowhead.
Border Color	controls the color of the arrow's "tail" and the border of the arrowhead.
Border Style	controls the line style of the arrow's "tail" and the border of the arrowhead.

Text Boxes

You create text boxes using the Text tool on the graph window SpeedBar. Text boxes have two different Object Inspector menus: one for the text itself and one for the box that surrounds the text. To display the text Object Inspector, right-click the text and choose the Properties command in the SpeedMenu. To display the text box Object Inspector, right-click any part of the background area that is not covered by text and choose the Properties command.

An easy way to determine which Object Inspector you'll see is to move (not drag) the mouse pointer over the text box. If you right-click when the pointer is an arrow, you can display text box properties. When the pointer is an I-beam, right-clicking accesses text properties.

Text properties

Text Font	determines the font and font size of the text, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of the text.
Text Bkg Color	provides the second color for two-color fills and shadows.
Text Style	sets the fill style for text, and lets you add a drop shadow.

Text box properties

Graph Button	turns a text box into a "live" button in a slide show.
Alignment	makes boxed text left-justified, centered, or right-justified.
Box Type	determines the look of the box.
Fill Color	assigns a fill color to the box.
Bkg Color	provides a second color for the box. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the box.
Border Color	controls the color of the box's border.

Ellipses

The Ellipse tool on the graph window SpeedBar creates circles and ovals. Ellipses have these properties:

Fill Color	assigns a fill color to the interior of the ellipse or circle.
Bkg Color	provides a second color for the interior. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the interior.
Border Color	controls the color of the outline around the ellipse or circle.
Border Style	controls the line style of the border.

You can also use the palette on the SpeedBar to change these properties (see Using Color Palettes for details).

Freehand Polygons

The Freehand Polygon tool on the graph window SpeedBar creates a closed shape after you drag the mouse to draw the border. Freehand Polygons have these properties:

Fill Color	assigns a fill color to the interior of the polygon.
Bkg Color	provides a second color for the interior. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the interior.
Border Color	controls the color of the outline around the polygon.
Border Style	controls the line style of the border.

You can also use the palette on the SpeedBar to change these properties (see Using Color Palettes for details).

Freehand Polylines

The Freehand Polyline tool on the graph window SpeedBar draws a line anywhere you drag the mouse. Freehand Polylines have these properties:

Line Color	lets you choose the color of the polyline.
Line Style	lets you change the thickness of the polyline, or choose one of several dashed line styles.

To quickly change the line color only, select the freehand polyline, then click a square on the Draw palette (see Using Color Palettes for details).

Lines

The Line tool on the graph window SpeedBar draws a straight line when you drag the mouse. Lines have these properties:

Line Color	lets you choose the color of the line.
Line Style	lets you change the thickness of the line, or choose one of several dashed line styles.

To quickly change the line color only, select the line, then click a square on the Draw palette (see Using Color Palettes for details).

Polygons

The Polygon tool on the graph window SpeedBar creates closed shapes with as many sides as you want. Polygons have these properties:

Fill Color	assigns a fill color to the interior of the polygon.
Bkg Color	provides a second color for the interior. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the interior.
Border Color	controls the color of the outline around the polygon.
Border Style	controls the line style of the border.

You can also use the palette on the SpeedBar to change these properties (see Using Color Palettes for details).

Polylines

The Polyline tool on the graph window SpeedBar creates lines with as many straight segments as you want. Polylines have these properties:

Line Color	lets you choose the color of the polyline.
Line Style	lets you change the thickness of the line, or choose one of several dashed line styles.

To quickly change the line color only, select the polyline, then click a square on the Draw palette (see Using Color Palettes for details).

Rectangles

The Rectangle tool on the graph window SpeedBar creates rectangles and squares. Rectangles have these properties:

Fill Color	assigns a fill color to the interior of the rectangle or square.
Bkg Color	provides a second color for the interior. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the interior.
Border Color	controls the color of the outline around the rectangle or square.
Border Style	controls the line style of the border.

You can also use the palette on the SpeedBar to change these properties (see Using Color Palettes for details).

Rounded Rectangles

The Rounded Rectangle tool on the graph window SpeedBar creates rectangles and squares that have rounded corners. Rounded Rectangles have these properties:

Fill Color	assigns a fill color to the interior of the rectangle or square.
Bkg Color	provides a second color for the interior. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the interior.
Border Color	controls the color of the outline around the rectangle or square.
Border Style	controls the line style of the border.

You can also use the palette on the SpeedBar to change these properties (see Using Color Palettes for details).

3-D Walls and Base

Area Fill properties control the appearance of the walls and base of 3-D graphs. The base and each wall are separate objects; you view and change these properties individually:

Fill Color	assigns a fill color to the wall or base.
Bkg Color	provides a second color for the wall or base. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the wall or base.
Border Color	controls the color of all edges of the wall or base.
Border Style	controls the line style of all edges of the wall or base.

Note: Grid lines on the walls are a property of the Y-Axis. Grid lines on the base are properties of the X-Axis.

Area Series

The area series Object Inspector lists all the properties available for customizing each series in an area, surface, shaded surface, or contour graph. To display the Object Inspector, right-click the area that represents the series you want to change and choose the Properties command.

Series Options	control the way the selected series is plotted. By changing series option settings, you can change the data that is plotted or override legend labels. (You cannot add a label series, change the graph type of a selected series, or plot a series on a secondary y-axis in area and surface graphs.)
Analyze	selects the type of analytical graphing.
Fill Color	selects the interior color of the area or surface.
Bkg Color	provides the second color for areas that have a pattern or wash fill style.
Fill Style	can be none (transparent), a solid color, a pattern, a wash, or a bitmap graphic.
Border Color	is the color of the lines that surround the selected area.
Border Style	chooses the thickness of the lines that surround an area.

Axis Title

You create titles for the x-axis, the y-axis, and the secondary y-axis using the Graph|Titles command. Titles have these properties:

Title	changes the words in the title.
Text Color	is the fill color of the axis title text.
Text Bkg Color	is the second color for pattern or wash text styles.
Text Font	selects the font family and size of axis title text.
Text Style	determines whether the interior of each letter will be solid, a pattern, a wash, or a bitmap. You can also can add a drop shadow.

Bar Series

The bar series Object Inspector lists all the properties available for customizing each series in a bar graph. These are the properties:

Series Options	control the way the selected series is plotted. You can change the data that is plotted, add series labels, override legend labels, change the graph type of just the selected series, and plot the selected series against the secondary y-axis.
Analyze	selects the type of analytical graphing.
Bar Options	adjust the width, spacing, and overlap of all bars in the graph.
Comparison Line	selects the style and color of a comparison line for a stacked or 100% stacked bar comparison graph.
Fill Color	selects the interior color of the bars.
Bkg Color	provides the second color to bars that have a pattern or wash fill style. (You won't see the background color if the bars don't have one of these fill styles.)
Fill Style	makes the interior of the bars transparent (select None), a solid color, a pattern, a wash, or a bitmap graphic.
Border Color	selects the color of the frame around the bar.
Border Style	chooses the line thickness of the border.

Column Graphs

Column graphs plot a single series. Each value in the series is represented by a section of the column, and section labels usually express values as a percentage of the whole. Negative values are plotted as though they are positive.

To change the appearance of a column graph, right-click a section and choose the Properties command to display its Object Inspector. These five properties apply to every label on the column graph:

Label Options	define a label series, determine how the data label is expressed, and turn tick mark display off and on.
Text Font	determines the font and font size of label text, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of label text.
Text Bkg Color	provides the second color for two-color fills and shadows.
Text Style	sets the fill style for label text, and lets you add a drop shadow.

These properties apply to the column section you right-clicked:

Fill Color	assigns a color to the interior of the section.
Bkg Color	assigns a second color for a pattern or wash fill style.
Fill Style	makes the interior transparent, or fills it with a solid color, a pattern, a wash, or an imported bitmapped graphic.
Border Color	controls the color of the perimeter of the section.
Border Style	controls the line style of the perimeter.

Graph Pane

The graph pane is the area bordered by the x- and y-axes in a 2-D graph. Grid lines, and the bars, lines, or areas that represent series are drawn on top of this area. (Horizontal grid lines are a property of the y-axis; vertical grid lines belong to the x-axis.)

To change graph pane properties, right-click the graph pane and choose the Properties command, or choose Property|Graph Pane. There are six properties:

Border Options	let you display or hide each edge of the graph pane.
Fill Color	selects the interior color of the graph pane area.
Bkg Color	provides the second color when the graph pane has a pattern or wash fill style.
Fill Style	makes the graph pane transparent, a solid color, a pattern, a wash, or a bitmapped graphic.
Border Color	is the color of the graph pane perimeter.
Border Style	chooses the thickness of the graph pane perimeter.

Graph Setup and Background

The graph background is the area behind the graph title, the legend, the graph itself, and any drawing elements you add. The graph setup and background Object Inspector appears when you right-click the graph background and choose the Properties command or choose Property|Graph Setup. These are the properties:

Graph Type	determines the kind of graph displayed.
Legend Position	specifies where the graph legend appears.
3-D View	changes the way the graph is drawn by adjusting the rotation, elevation, perspective, depth, and height of the graph.
3-D Options	turn wall and base display off and on.
Box Type	changes the style of the box that encloses the graph background.
Fill Color	selects the interior color of the graph background.
Bkg Color	provides the second color when the background has a pattern or wash fill style.
Fill Style	makes the background transparent, a solid color, a pattern, a wash, or a bitmapped graphic.
Border Color	is the color of the box that encloses the graph background.
Graph Button	turns the graph background into a "live" button in a slide show.

Graph Subtitle

The graph title and subtitle appear in a single text box, which is centered above the graph; the subtitle is placed under the title. To change the appearance of the subtitle, right-click it and choose the Properties command to display the subtitle Object Inspector.

Subtitle Font	determines the font, font size, .and adds bold, italic, underlined, or strikeout effects.
Subtitle Color	determines the color of the text.
Subtitle Bkg Color	provides the second color for two-color fills and shadows.
Subtitle Style	sets the fill style for text, and lets you add a drop shadow.

Graph Title

The graph title and subtitle appear in a single text box, which is centered above the graph; the title is placed over the subtitle. To change the appearance of the title, right-click it and choose the Properties command to display the title Object Inspector. These are the properties:

Text Font	determines the font, and font size of the title, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of the title text.
Text Bkg Color	provides the second color for two-color text fills and shadows.
Text Style	sets the fill style for the title text, and lets you add a drop shadow.

Graph Title Box

The graph title and subtitle are enclosed in a single text box. When you create a title, you don't see the text box because its fill style and box type are set to None by default. To change the appearance of the title box, click either the title or the subtitle to display the handles that mark the boundary of the graph title box, then move (don't drag) the mouse pointer over the title and subtitle area. When the pointer is an arrow, right-click and choose the Properties command to display the graph title box Object Inspector. There are five properties:

Box Type	determines the look of the box.
Fill Color	assigns a fill color to the box.
Bkg Color	provides a second color for the box. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the box.
Border Color	controls the color of the box's border.

Graph Window

Every graph has its own window; you set properties for each graph window individually. To display the Object Inspector for the active graph window, right-click the title bar (or the window area outside the graph border, if the graph window is maximized), or choose Property|Graph Window. There are two properties:

Aspect Ratio	is the proportion of width to height. Choices are Floating Graph, Screen Slide, 35mm Slide, Printer Preview, and Full Extent.
Grid Options	turn grid display off and on, control the spacing of grid dots, and turn the Snap to Grid feature off and on.

Legend

A legend box appears as soon as you define a legend series in all 2-D graphs (except pie, doughnut, and column graphs), and in some 3-D graphs. To change legend properties, right-click the legend box and choose the Properties command, or choose Property|Legend. (Avoid the series markers--a right-click on a series marker accesses series properties.) The properties are:

Legend Position	specifies where the graph legend appears.
Text Font	determines the font and font size of legend label text, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of legend label text.
Text Bkg Color	provides the second color for two-color text fills and shadows.
Text Style	sets the fill style for text, and lets you add a drop shadow.
Box Type	determines the look of the legend box perimeter.
Fill Color	assigns a fill color to the legend box.
Bkg Color	provides a second color for the legend box. (Background color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the legend box.
Border Color	controls the color of the legend box's border.

Legend labels in 3-D and multiple graphs

Many 3-D graphs display legend labels along the z-axis. In multiple graphs, the legend label for each series appears as a "title" over the sub-graph that plots the series. Right-click a label and choose the Properties command, or choose Property|Legend to display the Object Inspector for this legend, which lists Text Color, Text Bkg Color, Text Font and Text Style properties.

Line Series

A line series is any data series that is plotted as a line in a line graph, rotated line graph, 3-D ribbon graph, or combination graph. To change line series properties, right-click either the line or the legend marker that corresponds to the line and choose the Properties command, or choose Property|Series. The properties are:

Series Options	control the way the selected series is plotted. You can change the data that is plotted, add series labels, override legend labels, change the graph type of just the selected series, and plot the selected series against the secondary y-axis.
Analyze	selects the type of analytical graphing.
Marker Style	changes the way each data point appears.
Fill Color	selects the interior color of the markers.
Bkg Color	provides the second color for markers that have a pattern or wash fill style.
Fill Style	sets the interior style of solid markers to None (transparent), a solid color, a pattern, a wash, or a bitmap graphic.
Line Color	lets you choose the color of the line that connects the points in the series.
Line Style	lets you change the thickness of the line that connects the points. You can also choose from several dashed line styles.

Pie and Doughnut Graphs

Pie and doughnut graphs plot a single series; each value is a "slice." Negative values in the series are plotted as though they are positive. You can explode a slice (pull it away from the rest of the circle) to emphasize it.

When you define an x-axis series or label series for a pie or doughnut graph, its labels appear next to the percentages. By default, the label for each slice shows what percentage the segment contributes to the whole. A tick mark connects the label and the slice.

To explode a slice, change the appearance of slices or labels, define a label series, or turn tick mark display off or on, right-click a pie or doughnut slice and choose the Properties command to display the Object Inspector.

These five properties apply to every label on the pie or doughnut graph:

Label Options	define a label series, determine how the data label is expressed, and turn tick mark display off and on.
Text Font	determines the font and font size of label text, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of label text.
Text Bkg Color	provides the second color for two-color fills and shadows.
Text Style	sets the fill style for label text, and lets you add a drop shadow.

These properties apply to the slice you right-clicked:

Explode Slice	pulls the slice away from the rest of the circle. You set the distance as a percentage of the radius.
Fill Color	assigns a color to the interior of the slice.
Bkg Color	assigns a second color for a pattern or wash fill style.
Fill Style	makes the interior transparent, or fills it with a solid color, a pattern, a wash, or an imported bitmapped graphic.
Border Color	controls the color of the perimeter of the slice.
Border Style	controls the line style of the perimeter.

Series Labels

Series labels appear on the bars, markers, or data points that represent each value in a bar graph or line graph. See Adding Labels to Bars and Data Points for details on creating series labels.

To change the properties of series labels, right-click one of the labels and choose the Properties command. There are six properties, which apply to every series label in the graph:

Label Alignment	sets the position of the labels relative to the data points. The graph type determines which alignment options are available.
Format	specifies how labels are displayed (as text, or in a numeric format such as date or currency, for example). See Numeric Format for a description of numeric format options.
Text Font	determines the font and font size of the series label text, and adds bold, italic, underlined, or strikeout effects.
Text Color	determines the color of the text.
Text Bkg Color	provides the second color for two-color fills and shadows.
Text Style	sets the fill style for text, and lets you add a drop shadow.

To delete series labels, delete the label series.

X-Axis

All graphs, except pie, doughnut, and column graphs, position data relative to two scales: the x-axis and the y-axis. The x-axis is the horizontal scale on standard graphs and the vertical scale on rotated graphs. X-axis divisions typically represent categories or time periods; labels along this axis are provided by the x-axis series.

XY graphs (often called scatter diagrams) plot values against two numeric scales. The x-axis scale in these graphs is determined by the values in the x-axis series. Unlike most other graph types, where the x-axis series contains labels, the x-axis series in an XY graph contains data.

Normal Graph

X-Axis Series	selects the block of labels placed beneath each division of the x-axis.
Tick Options	turn x-axis label display off or on, and choose a method for dealing with overlapping labels.
Text Color	selects the principal color of the x-axis label text.
Text Bkg Color	provides the second color for pattern and wash Text Styles. The text background color is also the drop shadow color.
Text Font	determines the font and font size of the label text, and whether the text appears bold, italic, underlined, or with strikeout.
Text Style	sets the fill style for labels, and lets you add a drop shadow.
Major Grid Style	selects a style and color for the lines that separate x-axis divisions.

XY Graph (same as Y-Axis Properties)

Scale	controls the range of values plotted on the x-axis and determine the placement of major and minor divisions and scale labels.
Tick Options	control the appearance of tick marks and labels.
Numeric Format	specifies how the numbers on the x-axis scale are expressed (as percentages, as currency, or in scientific notation, for example).
Text Color	selects the principal color of the x-axis labels.
Text Bkg Color	provides the second color for pattern and wash Text Styles. The text background color is also the drop shadow color.
Text Font	determines the font and font size of the labels, and whether the text appears bold, italic, underlined, or with strikeout.
Text Style	sets the fill style for labels, and lets you add a drop shadow.
Major Grid Style	selects the color and style of the lines that extend from each scale label to the opposite side of the graph.
Minor Grid Style	selects the color and style of the lines that extend from positions *between* the scale labels on the y-axis to the opposite side of the graph.

YAxis

All graphs, except pie, doughnut, and column graphs, position data relative to two scales: the x-axis and the y-axis. The y-axis is the vertical scale on standard graphs and the horizontal scale on rotated graphs. Y-axis scale divisions and label positions are initially determined by the data plotted in the graph. The properties are:

Scale	lets you change the range of values plotted on the y-axis and determine the placement of major and minor divisions and scale labels.
Tick Options	control the appearance of tick marks and labels.
Numeric Format	specifies how the numbers on the y-axis scale are expressed (as percentages, as currency, or in scientific notation, for example).
Text Color	selects the principal color of the y-axis labels.
Text Bkg Color	provides the second color for pattern and wash Text Styles. The text background color is also the drop shadow color.
Text Font	determines the font and font size of the labels, and whether the text appears bold, italic, underlined, or with strikeout.
Text Style	sets the fill style for labels, and lets you add a drop shadow.
Major Grid Style	selects the color and style of the lines that extend from each scale label to the opposite side of the graph.
Minor Grid Style	selects the color and style of the lines that extend from positions *between* the scale labels on the y-axis to the opposite side of the graph.

Objects in the Notebook Window

While a notebook window is active, you can right-click and choose the Properties command or use the Property|Current Object command to display the properties of any of the following objects:

Active Block	Controls settings for the current block or cell
Graph Object	Controls settings for the selected floating graph
SpeedButton	Controls settings for the selected SpeedButton
Bitmap Object	Controls settings for the selected bitmap
Picture Object	Controls settings for the selected picture
OLE Object	Controls settings for the selected OLE object

Other objects that can be accessed only while a notebook window is active are

Active Notebook

Active Page

In addition, the Graphs page of a notebook window supports the following objects:

Graph Icon

Dialog Icon

Slide Show Icon

See Also

Locating Notebook Window Objects

Properties in the Notebook Window

Active Block Properties

The active block is the currently selected cell or group of cells. Most block properties affect the appearance of cell entries or of rows or columns. A few properties, however, such as Protection and Data Entry Input, affect the ways you can enter data into blocks.

Numeric Format	Changes display format of values in a block.
Font	Sets typeface, style, and point size of text for the selected block.
Shading	Shades blocks of cells with specified color.
Alignment	Changes alignment of cell entries in a block.
Line Drawing	Draws lines around cells and blocks.
Protection	Disables or enables spreadsheet protection for individual blocks of cells.
Text Color	Specifies text color for the selected block.
Data Entry Input	Constrains types of data that can be entered in a block
Row Height	Sets and resets the height of selected rows.
Column Width	Sets and resets the width of selected rows.
Reveal/Hide	Hides and displays columns and rows.

Note: Although it's tempting to preset block properties in large areas of the notebook, doing so consumes memory. It's more efficient to set properties only in the cells or pages you're currently using. Or, change the Normal style to change default properties.

If you move a cell, its properties move with the data and are removed from the original cell. If you copy a cell, the copy takes on the properties of the original cell. If you paste a copied or cut block with Edit|Paste Special, you can control whether the cell's properties are pasted (see Using Paste|Special for details).

See Also

Active Page Properties

Active Notebook

Graph Object Properties

A floating graph object appears in a layer above the cells in a spreadsheet page.

Source Graph	Specifies the graph displayed in the floating object.
Border Color	Sets the color of the floating graph's border.
Box Type	Determines the type of box to surround the floating graph.
Object Name	Specifies the name of the current floating graph.

See Also
Creating a Floating Graph

Active Notebook Properties

Notebook properties control recalculation methods, zoom effects, the color palette, screen display and whether the notebook is a macro library. The settings are saved with the notebook and affect all pages in the notebook.

You can display the notebook Object Inspector by right-clicking the notebook's title bar, or by choosing Property|Active Notebook.

Recalc Settings	Determines calculation mode, order, and number of iterations for spreadsheet recalculation; toggles formula compilation and ERR or NA source display in cells.
Zoom Factor	Proportionally reduces or enlarges text and graphics onscreen without actually resizing them.
Palette	Determines the default color palette available for notebook cell colors.
Display	Hides parts of the notebook display
Macro Library	Makes a notebook into a macro library
Password Level	Sets a password and determines the security level of a notebook.
System	Turns a notebook into a system notebook.

See Also

Active Page Properties

Active Block Properties

Page Properties

Most page properties affect the appearance of data or screen elements, with a few exceptions: Name (which enters a page name) and Protection (which turns on overall page protection).

Name	Names the active page.
Protection	Sets or removes write-protection for a spreadsheet page
Line Color	Changes the color of drawn lines.
Conditional Color	Assigns colors to ERR values and values within, greater than, and less than a specified range.
Label Alignment	Sets the position of all labels within cells, except those set individually.
Display Zeros	Determines whether or not to display zero values.
Default Width	Sets the column width for all columns, except those adjusted individually.
Borders	Removes or displays the notebook borders.
Grid Lines	Hides and displays the spreadsheet grid.
Tab Color	Changes the tab color of the active notebook page.

See Also

Active Block

Active Notebook

SpeedButton Properties

A SpeedButton floats above cells in a spreadsheet page and, when pressed, runs a macro.

Macro	Specifies a macro for the selected SpeedButton to run.
Label Text	Specifies text to display on the selected SpeedButton.
Border Color	Sets the color of the SpeedButton's border.
Box Type	Determines the type of box to surround the SpeedButton.
Object Name	Specifies the name of the current SpeedButton object.

See Also
Creating Macro SpeedButtons

Bitmap Object Properties

A bitmap object floats above cells in a spreadsheet page and contains a picture in bitmap form.

Border Color	Sets the color of the object's border.
Box Type	Determines the type of box to surround the object.
Object Name	Specifies the name of the current object.

See Also
Creating Floating Objects

Picture Object Properties

A picture object floats above cells in a spreadsheet page and contains a picture in the metafile graphics format.

Border Color	Sets the color of the object's border.
Box Type	Determines the type of box to surround the object.
Object Name	Specifies the name of the current object.

See Also
Creating Floating Objects

OLE Object Properties

An OLE object floats above cells in a spreadsheet page and links other applications to Quattro Pro. Depending on the type of OLE object, the Object Inspector contains either Link or Object Settings.

Link Settings	Controls behavior of a linked OLEobject.
Object Settings	Controls behavior of an embedded OLE object.
Border Color	Sets the color of the object's border.
Box Type	Determines the type of box to surround the object.
Object Name	Specifies the name of the current object.

See Also
Changing OLE Object Properties

Graph Icon Properties

Graph icons appear on the Graphs page. Each icon represents a specific graph in the active notebook.

Name	Specifies a name for a graph icon.

See Also
The Graphs Page

Slide Show Icon Properties

Slide show icons appear on the Graphs page. Each icon represents a specific slide show in the active notebook.

Name	Specifies a name for a slide show icon.
Default Effect	Specifies a default slide show effect.

See Also

Editing a Slide Show

The Graphs Page

Dialog Icon Properties

Dialog icons appear on the Graphs page. Each icon represents a specific dialog box in the active notebook.

Name	Specifies a name for a dialog icon

See Also
The Graphs Page

Property Reference

Quattro Pro has a variety of commands and @functions that affect property settings. This help section lists the objects and properties you can manipulate using


@PROPERTY


{GETOBJECTPROPERTY}


{GETPROPERTY}


{SETOBJECTPROPERTY}


{SETPROPERTY}


link commands in a dialog box or menu command

The topics in this help section list the various Quattro Pro objects and describe how to identify the objects in an @function or command.

Property Reference Overview

The property tables in this section are divided into three parts: the Object|Property column, the Argument column, and the Syntax column.

The Object|Property column lists the objects and their properties. The Argument column shows how to enter the property name in an @PROPERTY function or macro command. If a property name is followed by an asterisk (*), it's a hidden property and doesn't appear in the object's Object Inspector. If a property name is followed by (R), you cannot set it with a macro command or link command; it is a read only property.

The Syntax column shows the layout of special property settings. If a syntax isn't listed, look at how the setting is changed in the object's Object Inspector menu. For example, because the Horizontal Scroll Bar option in a notebook Object Inspector is set with a check box, the property setting is entered as Yes or No; similarly, if a setting is changed by an edit field, you can use the text normally typed into that edit field.

Italicized items in the Syntax column describe the type of data returned; items in normal type are entered (when setting a property). If vertical bars (|) separate items, then only those items are returned or allowed in the property. For example, Both|Window|Panel|None means that the property setting is either Both, Window, Panel, or None; you can enter only one of these items to set the property. Another example is Precision|Type, which indicates that either the type or precision setting is listed in that position. Items in angle brackets (<>) are optional.

See Also
Property Reference

Object Precedence

If a situation arises where a property command could affect multiple objects (because they have the same name), the object highest on this list is identified:

1. Dialog box
2. Graph
3. Floating object
4. Named block

When a Graphs page is active, you can't identify a dialog box or graph. The icon representing the dialog box or graph is identified instead.

See Also

Property Reference Overview

Property Reference

Identifying Objects

Entering a string into a command (or @function) that specifies the name of an object and property to manipulate is called identifying an object. The following table lists the ID syntax of Quattro Pro objects. In the table, Property is the name of the property setting to read or set. ObjName is the name of the object to manipulate. This name is stored one of the following properties: Name, Object Name, or Object ID.

Object	ID Syntax	Notes
Block (or Cell)	[NBName]BlockAddress.Property	[NBName] is optional
Dialog box	[NBName]DialogName:.Property	[NBName] is optional.
Dialog control	[NBName]DialogName:ObjName.Property	[NBName] is optional. DialogName: isn't needed if the dialog box containing the control is active.
Dialog icon	DialogName.Property	Dialog icons can only be identified when the Graphs page is selected.
Floating object	[NBName]Page:ObjName.Property	[NBName] and Page: are optional.
Graph	[NBName]GraphName.Property	[NBName] is optional.
Graph icon	GraphName.Property	Graph icons can only be identified when the Graphs page is selected.
Graph object	[NBName]GraphName:ObjName.Property	[NBName] is optional. GraphName: isn't needed if the graph window containing the object is active.
Menu command	Menupath.Property	
Notebook	[NBName].Property	You can change properties of the active notebook using Active_Notebook.Property.
Page	Active_Page.Property	You can only read or set properties of the active page using this syntax.
Slide show icon	GraphName.Property	Slide show icons can only be identified when the Graphs page is selected.
SpeedBar	[SpdBarName].Property	
SpeedBar control	[SpdBarName]ObjName.Property	[SpdBarName] isn't needed if the SpeedBar containing the control is active.

See Also

Block Property Example
Control Property Example
Property Reference Overview
Property Reference

Block Property Example

The following table shows examples of how you can apply the information in the Identifying Objects topic to manipulate a block property: the Numeric Format property of the cell A:A23.

Property Command	Notes
{GETOBJECTPROPERTY B:C32,"A:A23.Numeric_Format"}	The setting is stored in B:C32.
ON Init SET General TO A:A23.Numeric_Format	You can also use ID's with the SEND and RECEIVE commands. This link command formats A:A23 as General.
@PROPERTY("A:A23.Numeric_Format")	
{SETOBJECTPROPERTY "A:A23.Numeric_Format","Currency,2"}	This command formats A:A23 as Currency (2 decim places).

See Also

Control Property Example

Identifying Objects

Property Reference Overview

Control Property Example

The following table shows examples of how you can apply the information in the Identifying Objects topic to manipulate an object property: the Disabled property of a bitmap button named Bitmap1 in a dialog box named Dialog1.

Property Command	Notes
{GETOBJECTPROPERTY B:C32,"Dialog1:Bitmap1.Disabled"}	The setting is stored in B:C32.
ON Init SET Yes TO Dialog1:Bitmap1.Disabled	You can also use ID's with the SEND and RECEIVE link commands. This link command disables the button.
@PROPERTY("Dialog1:Bitmap1.Disabled")	
{SETOBJECTPROPERTY "Dialog1:Bitmap1.Disabled","No"}	This command enables the button.

{SETPROPERTY} and {GETPROPERTY} work on the selected object, and just take the name of the property to manipulate.

See Also

Block Property Example

Identifying Objects

Property Reference Overview

Common Properties

Many objects contain the same type of property. These are some of the more common object properties:

Color

Dimension

Font

See Also

Common Graph Object Properties

Property Reference Overview

Property Reference

The Color Property

The Color property appears in many Object Inspector menus, but with a variation in title (Border_Color, Line_Color, Text_Color, and so on). The syntax of Color is:

Red,Green,Blue

Red is the amount of red in the color. *Green* is the amount of green in the color. *Blue* is the amount of blue in the color. Each of these components is an integer from 0 to 255; 0 indicates that none of the hue is present; 255 indicates maximum saturation for the hue. Black is 0,0,0 and white is 255,255,255.

When setting a color using an Object Inspector menu, you can click HSB or CMY (on a color control) to specify the color using a different color model. Using a different color model does not affect the syntax of the Color property; the Color property is always in *Red,Green,Blue* format.

You can also retrieve each of these color components individually. The following table lists the argument required to read or set an individual component of a Color property. *Item* is the word appearing in the Object Inspector (and property tables) that describes what color is being manipulated.

Property	Argument	Syntax
Item_Color	*Item*_Color	Red, Green, Blue
Red	*Item*_Color.Red	
Green	*Item*_Color.Green	
Blue	*Item*_Color.Blue	

See Also

The Dimension Property

The Font Property

Common Graph Object Properties

Common Properties

The Dimension Property

The Dimension property shows the precise size and position of an object relative to the window containing it. The syntax of Dimension is:

X,Y,Width,Height

X is the distance in pixels between the left edge of the object and the left side of the window. *Y* is the distance in pixels between the top edge of the object and the bottom edge of the window's title bar (or from the top of the graph background in the case of a drawn object). *Width* is the width of the object, in pixels. *Height* is the height of the object, in pixels.

You can also retrieve each of these settings individually. The following table lists the argument required to read or set an individual option of the Dimension property.

Property	Argument	Syntax
Dimension	Dimension	*X, Y, Width, Height*
X Pos	Dimension.X	
Y Pos	Dimension.Y	
Width	Dimension.Width	
Height	Dimension.Height	

See Also

The Color Property

The Font Property

Common Graph Object Properties

Common Properties

The Font Property

The Font property appears in many Object Inspector menus, but with a variation in title (Font, Label_Font, Text_Font). The syntax of Font is:

Typeface, PointSize, Bold, Italic, Underline, Strikeout

Typeface is the name of the typeface used. The list of typefaces available varies from system to system. *PointSize* is the size of the text, in points (a point is 1/72nd of an inch). *Bold, Italic, Underline,* and *Strikeout* each set one attribute of the text's appearance. Each is set to Yes or No.

You can also retrieve each of options individually. The following table lists the argument required to read or set an individual option of a Font property. *Item* is the word appearing in the Object Inspector (and property tables) that describes what font is being manipulated.

Property	Argument	Syntax
Font	Font	*Typeface, PointSize, Bold, Italic, Underline, Strikeout*
Typeface	Font.Typeface	
Point Size	Font.Point_Size	
Bold	Font.Bold	
Italic	Font.Italic	
Underline	Font.Underline	
Strikeout	Font.Strikeout	
Item Font	*Item*_Font	*Typeface, PointSize, Bold, Italic, Underline, Strikeout*
Typeface	*Item*_Font.Typeface	
Point Size	*Item*_Font.Point_Size	
Bold	*Item*_Font.Bold	
Italic	*Item*_Font.Italic	
Underline	*Item*_Font.Underline	
Strikeout	*Item*_Font.Strikeout	

See Also

The Color Property

The Dimension Property

Common Graph Object Properties

Common Properties

Common Graph Object Properties

Two properties are found in many graph objects:

Fill Style

Text Style

See Also

Common Properties

Property Reference Overview

Property Reference

The Fill Style Property

The Fill Style property is common to many graph objects. It determines the color and pattern of objects. If Fill Style is None, leave the *Type* field empty

Property	Argument	Syntax
Fill Style	Fill_Style	None|Solid|Pattern|Wash|Bitmap, *Type*
None	Fill_Style	None
Solid	Fill_Style	Solid,Solid
Pattern	Fill_Style	Pattern,Solid
	Fill_Style	Pattern,"Tight Dots"
	Fill_Style	Pattern,"Thick Stripes Down"
	Fill_Style	Pattern,"Thick Stripes Up"
	Fill_Style	Pattern,"Vertical Lines"
	Fill_Style	Pattern,"Horizontal Lines"
	Fill_Style	Pattern,"Horizontal Grid"
	Fill_Style	Pattern,"Hatch 1"
	Fill_Style	Pattern,"Diagonal 1"
	Fill_Style	Pattern,"Diagonal 2"
	Fill_Style	Pattern,Vertical
	Fill_Style	Pattern,Horizontal
	Fill_Style	Pattern,"Loose Dots"
	Fill_Style	Pattern,"Medium Dots"
	Fill_Style	Pattern,Pepita
	Fill_Style	Pattern,Scales
	Fill_Style	Pattern,"Diagonal Grid"
	Fill_Style	Pattern,"Hatch 2"
	Fill_Style	Pattern,"Fuzzy Stripes Down"
	Fill_Style	Pattern,Weave
	Fill_Style	Pattern,"Zig Zag"
	Fill_Style	Pattern,"Staggered Dashes"
	Fill_Style	Pattern,Lattice
	Fill_Style	Pattern,Bricks
Wash	Fill_Style	Wash,"Left to right"
	Fill_Style	Wash,"Right to left"
	Fill_Style	Wash,"Center to left and right"
	Fill_Style	Wash,"Top to bottom"
	Fill_Style	Wash,"Bottom to top"
	Fill_Style	Wash,"Center to top and bottom"
Bitmap	Fill_Style	Bitmap,"Crop to fit"|"Shrink to fit",*BitmapName*

See Also

Property Reference Overview

Property Reference

The Text Style Property

The Text Style property is common to many graph objects. It determines whether graph text is solid, a washed tone, or a bitmap. If the style is Solid or Wash, leave the *BitmapName* field empty. *Shadow* indicates whether text should have a drop shadow (Yes) or no shadow (No).

Property	Argument	Syntax
Text Style	Text_Style	Solid|Wash|Bitmap, *WashType*, *BitmapName,Shadow*
Solid	Text_Style	Solid,None,,*Shadow*
Wash	Text_Style	Wash,"Left to right",,*Shadow*
	Text_Style	Wash,"Right to left",,*Shadow*
	Text_Style	Wash,"Center to left and right",,*Shadow*
	Text_Style	Wash,"Top to bottom",, *Shadow*
	Text_Style	Wash,"Bottom to top",,*Shadow*
	Text_Style	Wash,"Center to top and bottom",,*Shadow*
Bitmap	Text_Style	Bitmap,"Crop to fit"|"Shrink to fit",*BitmapName,Shadow*

See Also

Property Reference Overview

Property Reference

Active Object Properties

There are some situations where you want to view the property settings of an active object without using its exact name. You can use the names in the following table to identify these objects.

Active Object	Description
Active_Block	The selected block.
Active_Notebook	The active notebook window.
Active_Page	The active notebook page.

For example, this formula returns the setting of the Macro Library property in the active notebook window:

```
@PROPERTY("Active_Notebook.Macro_Library")
```

You can also use {SETPROPERTY} and {GETPROPERTY} to manipulate to properties of selected objects (blocks, controls, graph objects, floating objects, and so on).

This table lists the properties of active objects.

Object|Property	Argument	Syntax
Active Block	see Block in topic Notebook Object Properties for a list of block properties	
Active Notebook	see Notebook in topic Notebook Object Properties for a list of notebook properties	
Active Page		
Name	Name	
Protection	Protection	Disable|Enable
Line Color	Line_Color	0-15
Conditional Color	Conditional_Color	Enable, SmallVal, GreatVal, BelColor, Normal, AboveCol, ERRCol
Enable	Conditional_Color.Enable	
Smallest Normal Value	Conditional_Color.Smallest_Normal_Value	
Greatest Normal Value	Conditional_Color.Greatest_Normal_Value	
Below Normal Color	Conditional_Color.Below_Normal_Color	0-15
Normal Color	Conditional_Color.Normal_Color	0-15
Above Normal Color	Conditional_Color.Above_Normal_Color	0-15
ERR Color	Conditional_Color.ERR_Color	0-15
Label Alignment	Label_Alignment	Left|Right|Center
Display Zeros	Display_Zeros	
Default Width	Default_Width	
Borders	Borders	Row, Column
Row Borders	Borders.Row_Borders	
Column Borders	Borders.Column_Borders	
Grid Lines	Grid_Lines	Horizontal, Vertical
Horizontal	Grid_Lines.Horizontal	

Vertical	Grid_Lines.Vertical	
Tab Color	Tab_Color	see Color

See Also
Property Reference Overview
Property Reference

Menu Command Properties

Each command on the menu bar has properties you can manipulate (see Changing Command Properties for details). To identify a menu command, enter its path separated by forward slashes (/). Don't include ellipses (...). For example, this formula returns whether Edit|Paste Format is grayed on the menu:

```
@PROPERTY("/Edit/Paste Special.Grayed")
```

This table lists some special ways to identify menu commands.

ID Syntax	Description
/<-	The first item on the menu bar. You can include this name at the end of a menu path (for example, /File/<- identifies the first item on the File menu).
/->	The last item on the menu bar. You can include this name at the end of a menu path (for example, /Tools/Macro/-> identifies the last item on the Tools|Macro menu).
/n	The nth item on the menu bar. You can include this name at the end of a menu path (for example, /Help/2 identifies the second item on the Help menu).

This table lists the properties of menu commands.

Object|Property	Argument	Syntax
Menu Command		
Title *	Title	
Checked *	Checked	
HotKey *	HotKey	
Help Line *	Help_Line	
Depend On *	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Enabled *	Enabled	
Grayed *	Grayed	
Disabled *	Disabled	
Hidden *	Hidden	
Show *	Show	

See Also

Property Reference Overview

Property Reference

Dialog Control Properties

To identify a dialog box (not the objects in it), use its name followed by a colon. For example, `@PROPERTY("Dialog1:.Title")` returns the Title property of the dialog box Dialog1. To see the property settings of a dialog box outside of the active notebook, enter the notebook name in brackets before the dialog box name:

```
[NOTEBK1.WB1]Dialog1
```

You can use this syntax to identify objects in a dialog box:

[*Notebook*]*DialogName:ObjectName.Property*

Notebook is the name of the notebook containing the dialog box; it is optional. *DialogName* is the name of the dialog box containing the object (omit *DialogName*: if the dialog window containing the object is active). *ObjectName* is either the object ID number of the object (found in its Object ID property) or the name of the object (found in its Name property). *Property* is one of the strings listed in the Argument column of the dialog control property tables. For example, this formula returns the Dimension property of the object named Button1 in the dialog box Dialog1:

```
@PROPERTY("Dialog1:Button1.Dimension")
```

You can use a similar syntax to identify SpeedBars, but the name of the SpeedBar must be in brackets ([]). For example, this macro disables the SpeedBar QUIKSAVE.BAR:

```
{SETOBJECTPROPERTY "[QUIKSAVE.BAR].Disabled","Yes"}
```

You can use this syntax to identify objects in a SpeedBar:

[*SpdBarName*]*ObjectName.Property*

SpdBarName is the name of the SpeedBar containing the object (omit [*SpdBarName*] if the SpeedBar containing the object is active). *ObjectName* is either the object ID number of the object (found in its Object ID property) or the name of the object (found in its Name property). *Property* is one of the strings listed in the Argument column of the dialog property control tables. For example, this formula returns the Dimension property of the object named Button1 in the SpeedBar QUIKSAVE.BAR:

```
@PROPERTY("[QUIKSAVE.BAR]Button1.Dimension")
```

To view properties, arguments, and syntax for the following dialog and SpeedBar controls, choose from the list:

Bitmap Button

Button

Check Box

Color Control

Combo Box

Edit Field

Edit Integer

File Control

Group Box

Horizontal Scroller (HScrollBar)

Label

List Box

Pick List

Radio Button

Rectangle

Spin Control

Time Control (TimeCtrl)

Vertical Scroller (ScrollBar)

For a table of arguments and syntax for the entire dialog box and SpeedBar as objects, see Dialog Box and SpeedBar Properties.

See Also

Property Reference Overview

Property Reference

Dialog Box and SpeedBar Properties

This table shows arguments and syntax for the entire dialog box and SpeedBar as objects:

Object	Argument	Syntax
Dialog Box (Dialog) and SpeedBar		
Dimension	Dimension	see Dimension
Title	Title	
Position Adjust	Position_Adjust	*Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer*
Grid Options	Grid_Options	*Gridsize, GridShown, GridEnabled*
Name	Name	
Disabled	Disabled	
Enabled *	Enabled	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Bitmap Button Properties

Object|Property	Argument	Syntax
Bitmap Button		
Bitmap	Bitmap	
Label Text	Label_Text	
Text Draw Flags	Text_Draw_Flags	Apply, HorCenter, AlignRight(Yes)orLeft(No), VertCenter, AlignBottom(Yes)orTop(No), WordBreak, SingleLine
Default Button	Default_Button	
Button Type	Button_Type	Push Button|Radio Button|Check Box|OK Exit Button|Cancel Exit Button
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Button Properties

Object|Property	Argument	Syntax
Button		
Label Text	Label_Text	
Text Draw Flags	Text_Draw_Flags	Apply, HorCenter, AlignRight(Yes)orLeft(No), VertCenter, AlignBottom(Yes)orTop(No), WordBreak, SingleLine
Default Button	Default_Button	
Button Type	Button_Type	Push Button|Radio Button|Check Box|OK Exit Button|Cancel Exit Button
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Tab Stop	Tap_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Check Box Properties

Object|Property	Argument	Syntax
Check Box		
Draw to Right	Draw_to_right	
Process Value	Process_Value	
Label Text	Label_Text	
Text Draw Flags	Text_Draw_Flags	Apply, HorCenter, AlignRight(Yes)orLeft(No), VertCenter, AlignBottom(Yes)orTop(No), WordBreak, SingleLine
Button Type	Button_Type	Push Button|Radio Button|Check Box|OK Exit Button|Cancel Exit Button
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Color Control Properties

Object|Property	Argument	Syntax
Color Control		
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	Red,Green,Blue

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Combo Box Properties

Object|Property	Argument	Syntax
Combo Box		
List	List	
Allow Point Mode	Allow_Point_Mode	
Add Down Button	Add_Down_Button	
History List	History_List	
List Length	List_Length	
Terminate Dialog	Terminate_Dialog	
Edit Disabled	Edit_Disabled	
Edit Length	Edit_Length	
Ordered	Ordered	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Edit Field Properties

Object|Property	Argument	Syntax
Edit Field		
Field Type	Field_Type	Integer|String|Real|Range|Hidden
Edit Length	Edit_Length	
Allow Point Mode	Allow_Point_Mode	
Show Frame	Show_Frame	
Terminate Dialog	Terminate_Dialog	
Convert Text	Convert_Text	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Edit Integer Properties

Object|Property	Argument	Syntax
Edit Integer		
Minimum	Minimum	
Maximum	Maximum	
Default	Default	
Field Type	Field_Type	Integer|String|Real|Range|Hidden
Edit Length	Edit_Length	
Show Frame	Show_Frame	
Terminate Dialog	Terminate_Dialog	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

File Control Properties

Object|Property	Argument	Syntax
File Control		
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Group Box Properties

Object|Property	Argument	Syntax
Group Box		
Group Text	Group_Text	
Selected	Selected	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Disabled	Disabled	
Process Value	Process_Value	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Horizontal Scroller Properties

Object|Property	Argument	Syntax
Horizontal Scroller (HScrollBar)		
Parameters	Parameters	Min, Max, Line, Page, Time
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Label Properties

Object|Property	Argument	Syntax
Label		
Label Text	Label_Text	
Label Font	Label_Font	see Dimension
Text Draw Flags	Text_Draw_Flags	Apply, HorCenter, AlignRight(Yes) or Left(No), VertCenter, AlignBottom(Yes) or Top(No), WordBreak, SingleLine
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Grayed	Grayed	
Name	Name	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

List Box Properties

Object|Property	Argument	Syntax
List Box		
List	List	
Ordered	Ordered	
Number of Columns	Number_of_Columns	
Selection Text	Selection_Text	
Selected	Selected	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Pick List Properties

Object|Property	Argument	Syntax
Pick List		
Title	Title	
Selected	Selected	
Resize	Resize	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Radio Button Properties

Object|Property	Argument	Syntax
Radio Button		
Draw to Right	Draw_to_right	
Process Value	Process_Value	
Label Text	Label_Text	
Text Draw Flags	Text_Draw_Flags	Apply, HorCenter, AlignRight(Yes) or Left(No), VertCenter, AlignBottom(Yes) or Top(No), WordBreak, SingleLine
Button Type	Button_Type	Push Button|Radio Button|Check Box| OK Exit Button|Cancel Exit Button
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Rectangle Properties

Object|Property	Argument	Syntax
Rectangle		
Rectangle Style	Rectangle_Style	Plain|Framed|Beveled Out|Beveled In|Transparent
Fill Color	Fill_Color	see Color
Frame Color	Frame_Color	see Color
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPaged
Enabled *	Enabled	
Show *	Show	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Spin Control Properties

Object|Property	Argument	Syntax
Spin Control		
Edit Length	Edit_Length	
Show Frame	Show_Frame	
Minimum	Minimum	
Maximum	Maximum	
Default	Default	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Time Control Properties

Object|Property	Argument	Syntax
Time Control (TimeCtrl)		
Show Time	Show_Time	
Interval in Units	Interval_In_Units	
Units in Milliseconds	Units in Milliseconds	
Current Time (R)	Current_Time	Hour, Minute, Second
Hour	Current_Time.Hour	
Minute	Current_Time.Minute	
Second	Current_Time.Second	
Timer On	Timer_On	
Alarm Time	Alarm_Time	Hour,Minute,Second
Hour	Alarm_Time.Hour	
Minute	Alarm_Time.Minute	
Second	Alarm_Time.Second	
Alarm On	Alarm_On	
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Attach Child	Attach_Child	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Object Help	Object_Help	Title, Text, Context
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Vertical Scroller Properties

Object|Property	Argument	Syntax
Vertical Scroller (ScrollBar)		
Parameters	Parameters	Min, Max, Line, Page, Time
Dimension	Dimension	see Dimension
Hidden	Hidden	
Object ID (R)	Object_ID	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Name	Name	
Help Line	Help_Line	
Grayed	Grayed	
Disabled	Disabled	
Depend On	Depend_On	Desktop, NoteWin, GraphWin, DiaWin, EditWin, GraphsPage
Process Value	Process_Value	
Tab Stop	Tab_Stop	
Object Help	Object_Help	Title, Text, Context
Enabled *	Enabled	
Show *	Show	
Value *	Value	

See Also

Dialog Control Properties

Property Reference Overview

Property Reference

Graph Annotation Object Properties

Drawn objects in a graph are created by the graph SpeedBar; see Fixed Graph Object Properties for a list of fixed graph objects like the x-axis. To identify a drawn object in a graph, use this syntax:

[*Notebook*]*GraphName*:*ObjectName.Property*

Some properties of a drawn object only appear in an Object Inspector menu when Quattro Pro is loaded with the /D command line switch.

[*Notebook*] is the name of the notebook containing the graph; it's optional. *GraphName* is the name of the graph containing the object (omit *GraphName*: if the graph window containing the object is active). *ObjectName* is either the object ID number of the object (found in the Object ID property) or the name of the object (found in the Name property). *Property* is one of the strings listed in the Argument column of the next table. For example, this formula returns the Line Style property of the object named ProfitCallout in the graph YTD1:

```
@PROPERTY("YTD1:ProfitCallout.Line_Style")
```

To view properties, arguments, and syntax for the following graph annotation objects, choose from the list:

Arrow

Box

Ellipse

Freehand Polygon

Freehand Polyline

Line

Polygon

Polyline

Rectangle

Rounded Rectangle

Text

See Also

Property Reference Overview

Property Reference

Arrow Properties

Object|Property	Argument	Syntax
Arrow		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4| S1W1|S2W1|S3W1|S4W1| S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Box Properties

Object|Property	Argument	Syntax
Box		
Graph Button	Graph_Button	GotoSlide?, SlideName, Effect, Duration, Slow|Med| Fast, Overlay?, RunMacro?, MacroText
Alignment	Alignment	Left|Right|Center, WordWrap, TabStops
Box Type	Box_Type	No box|Single outline|Rounded corners| Double outline|Thick outline 1|Thick rounded corners|Three dimensional| Thick outline 2|Bevel out|Shadowed Thick outline 3|Bevel in
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Ellipse Properties

Object|Property	Argument	Syntax
Ellipse		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1| S2W1|S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Freehand Polygon Properties

Object|Property	Argument	Syntax
Freehand Polygon		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	
Line Color	Line_Color	see Color
Line Style	Line_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Freehand Polyline Properties

Object|Property	Argument	Syntax
Freehand Polyline		
Line Color	Line_Color	see Color
Line Style	Line_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Line Properties

Object|Property	Argument	Syntax
Line		
Line Color	Line_Color	see Color
Line Style	Line_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Polygon Properties

Object|Property	Argument	Syntax
Polygon		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1| S2W1|S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Polyline Properties

Object|Property	Argument	Syntax
Polyline		
Line Color	Line_Color	see Color
Line Style	Line_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Rectangle Properties

Object|Property	Argument	Syntax
Rectangle		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Rounded Rectangle Properties

Object|Property	Argument	Syntax
Rounded Rectangle		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Object|Property	Argument	Syntax
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1
Name	Name	
Dimension	Dimension	see Dimension
Object ID * (R)	Object_ID	

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Text Properties

Object|Property	Argument	Syntax
Text		
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Value	Value	Contents of the textbox

See Also

Graph Annotation Object Properties

Property Reference Overview

Property Reference

Fixed Graph Object Properties

To set a property of the graph (not the objects in it), use its name followed by a period. For example, this formula returns the Aspect Ratio property of the graph Graph1:

```
@PROPERTY("Graph1.Aspect_Ratio")
```

To identify a fixed object in a graph (as opposed to an object created with the graph SpeedBar; see the previous section for a list of drawn graph objects) use this syntax:

[*Notebook*]*GraphName*:*ObjectName.Property*

Notebook is the name of the notebook containing the graph; it is optional. *GraphName* is the name of the graph containing the object (omit *GraphName* if the graph window containing the object is active). For a list of *ObjectName* options, see Fixed Graph Object Names.

Property is one of the strings listed in the Argument column of the next table. For example, this formula returns the color of the third series of the graph PROFIT.

```
@PROPERTY("PROFIT:G$Series[3,1].Fill_Color")
```

To view properties, arguments, and syntax for the following fixed graph objects, choose from the list:

Area Fill

Axis Title

Bar Series

Line Series

Area Series

Column Graph

Graph Pane

Graph Setup and Background

Graph Subtitle

Graph Title

Graph Title Box

Graph Window

Legend Box

Pie Graph

Series Label

X-Axis

Y-Axis

See Graph Window Properties and Graph Setup and Background Properties for graph properties. For a list of drawn graph objects (such as arrows), see Graph Annotation Object Properties.

See Also

Property Reference Overview

Property Reference

Fixed Graph Object Names

To identify a fixed object in a graph (as opposed to an object created with the graph SpeedBar; see the previous section for a list of drawn graph objects) use this syntax:

[*Notebook*]*GraphName*:*ObjectName.Property*

The following table lists *ObjectName* settings. See Fixed Graph Object Properties for information on related arguments.

Object|Property	Description
G$Base	Base of a 3-D graph
G$Graph	Background of the graph window
G$LeftWall	Left wall of a 3-D graph grid
G$Legend	Graph legend
G$Series[x,y]	yth data point of the xth series in the graph
G$SeriesLabel	Series labels
G$Title	Title, subtitle, and title box of the graph
G$X1Axis	x-axis
G$X1Title	x-axis title
G$Y1Axis	Primary y-axis
G$Y1Title	Primary y-axis title
G$Y2Axis	Secondary y-axis
G$Y2Title	Secondary y-axis title

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Area Fill Properties

Object| Property	Argument	Syntax
Area Fill		
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1| S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Axis Title Properties

Object|Property	Argument	Syntax
Axis Title		
Title	Title	
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Bar Series Properties

Object|Property	Argument	Syntax
Bar Series		
Series Options	Series_Options	DataBlock, LabelBlock, Legend, Bar|Line|Area|Default, Primary|Secondary
Analyze	Analyze	None|Aggregration|Moving Average|Linear Fit|Exponential Fit,... **Note:** If Aggregration, other arguments are: ...<Table>,Show in Legend?(0|1), Days|Weeks|Months|Quarters|Years, Weeks|Months|Quarters|Years, SUM|AVG|STD|STDS|MIN|MAX|VAR|VARS **Note:** If Moving Average, other arguments are: ...<Table>,Show in Legend?(0|1), Period, None|Standard **Note:** If Linear Fit or Exponential Fit, other arguments are: ...<Table>, Show in Legend?(0|1)
Bar Options	Bar_Options	Width%, Margin%, No|Partial|Full
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2| S0W3| S0W4|S1W1|S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Line Series Properties

Object|Property	Argument	Syntax
Line Series		
Series Options	Series_Options	DataBlock, LabelBlock, Legend, Bar|Line|Area|Default, Primary|Secondary
Marker Style	Marker_Style	M00|M01|M02|M03|M04|M05|M06|M07|M08|M09|M10|M11|M12|M13|M14|M15, MarkerWeight, AutoSize?(0|1)
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Line Color	Line_Color	see Color
Line Style	Line_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Area Series Properties

Object|Property	Argument	Syntax
Area Series		
Series Options	Series_Options	DataBlock, LabelBlock, Legend
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1| S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Column Graph Properties

Object|Property	Argument	Syntax
Column Graph		
Label Options	Label_Options	LabelSeries, Currency|Value|Percent|None, ShowTick
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Pane Properties

Object|Property	Argument	Syntax
Graph Pane		
Border Options	Border_Options	Left, Top, Right, Bottom, Grids on Top
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1| S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Setup and Background Properties

Object|Property	Argument	Syntax
Graph Setup and Background		
Graph Type		
2-D	Graph_Type	Bar|Variance|Stacked Bar|HiLo|Line|XY|Area|Column|Pie|100 Stacked Bar|Stacked Line|100 Stacked Line|Doughnut|Polar Radar
3-D	Graph_Type	3D Bar|3D Stacked Bar|2DHalf Bar|3D Step|3D Unstacked Area|3D Ribbon|3D Area|3D Column|3D Pie|3D Contour|3D Surface|3D ShadedSurface|3d100 Stacked Bar|3D Doughnut
Rotate	Graph_Type	R3D bar|R2DHalf Bar|R2D Bar|Rotated Area|Rotated Line|R2D Stacked Bar|R2D100 Stacked Bar|R2D Stacked Line|R2D100 Stacked Line|R3D Stacked Bar|R3D100 Stacked Bar
Combo	Graph_Type	Area_bar|Hilo_bar|Line_bar|Multiple 3D columns|Multiple 3D Pies|Multiple Bar|Multiple Columns|Multiple Pies
Text	Graph_Type	Text
Legend Position	Legend_Position	None|Bottom|Right
3D View	3D_View	Rotation, Elevation, Perspective, Depth, Height
3D Options	3D_Options	ShowLeft, ShowBack, ShowBase, ThickWalls
Box Type	Box_Type	No box|Single outline|Rounded corners|Double outline|Thick outline 1|Thick rounded corners|Three dimensional|Thick outline 2|Bevel out|Shadowed|Thick outline 3|Bevel in
Fill Color	Fill_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Bkg Color	Bkg_Color	see Color
Graph Button	Graph_Button	GotoSlide?, SlideName, Effect, Duration, Slow|Med|Fast, Overlay?, RunMacro?, MacroText
Name *	Name	

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Subtitle Properties

Object|Property	Argument	Syntax
Graph Subtitle		
Subtitle Color	Subtitle_Color	see Color
Subtitle Bkg Color	Subtitle_Bkg_Color	see Color
Subtitle Font	Subtitle_Font	see Font
Subtitle Style	Subtitle_Style	None|Solid|Wash|Bitmap, *Type, Shadow*
Solid	Subtitle_Style	Solid,,*Shadow*
Wash	Subtitle_Style	Wash,"Left to right", *Shadow*
	Subtitle_Style	Wash,"Right to left", *Shadow*
	Subtitle_Style	Wash,"Center to left and right", *Shadow*
	Subtitle_Style	Wash,"Top to bottom", *Shadow*
	Subtitle_Style	Wash,"Bottom to top", *Shadow*
	Subtitle_Style	Wash,"Center to top and bottom", *Shadow*
Bitmap	Subtitle_Style	Bitmap,"Crop to fit", *BitmapName, Shadow*
		Bitmap,"Shrink to fit", *BitmapName, Shadow*

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Title Properties

Object|Property	Argument	Syntax
Graph Title		
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Title Box Properties

Object|Property	Argument	Syntax
Graph Title Box		
Box Type	Box_Type	No box|Single outline|Rounded corners|Double outline|Thick outline 1|Thick rounded corners|Three dimensional|Thick outline 2|Bevel out|Shadowed|Thick outline 3|Bevel in
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Dimension	Dimension	see Dimension
Name		Name

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Graph Window Properties

Object|Property	Argument	Syntax
Graph Window		
Aspect Ratio	Aspect_Ratio	Floating Graph|Screen Slide|35mm Slide| Printer Preview|Full Extent
Grid	Grid	GridSize, DisplayGrid, SnapToGrid

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Legend Box Properties

Object|Property	Argument	Syntax
Legend Box		
Legend Position	Legend_Position	None|Bottom|Right
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Box Type	Box_Type	No box|Single outline|Rounded corners|Double outline|Thick outline 1|Thick rounded corners|Three dimensional|Thick outline 2|Bevel out|Shadowed|Thick outline 3|Bevel in
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Pie Graph Properties

Object|Property	Argument	Syntax
Pie Graph		
Explode Slice	Explode_Slice	Distance, Explode
Label Options	Label_Options	Series, Currency|Value|Percent|None, ShowTick
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Fill Color	Fill_Color	see Color
Bkg Color	Bkg_Color	see Color
Fill Style	Fill_Style	see Common Graph Object Properties
Border Color	Border_Color	see Color
Border Style	Border_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Series Label Properties

Object|Property	Argument	Syntax
Series Label		
Label Alignment	Label_Alignment	Above|Top|Center|Below for bar graphs Above|Center|Below|Left|Right for area and line graphs
Format	Format	Format, Precision|Type
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

X-Axis Properties

Object|Property	Argument	Syntax
X-Axis		
Scale	Scale	Normal|Log, Automatic, High, Low, Increment, #Minors, ShowUnits **Note:** This property only appears when inspecting an XY graph
X-Axis Series	X-Axis_Series	
Tick Options	Tick_Options	None|Below|Above|Across, DisplayLabels, NumRows, NoOverlap, #OfLabelsToSkip, Limit
Numeric Format	Numeric_Format	Format, Precision|Type **Note:** This property only appears when inspecting an XY graph
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Major Grid Style	Major_Grid_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1,Red,Green,Blue
Minor Grid Style	Minor_Grid_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1,Red,Green,Blue **Note:** This property only appears when inspecting an XY graph

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Y-Axis Properties

Object|Property	Argument	Syntax
Y-Axis		
Scale	Scale	Normal|Log, Automatic, High, Low, Increment, #Minors, ShowUnits **Note:** An additional setting, ZeroLine, appears if the graph type is Variance.
Tick Options	Tick_Options	None|Left|Right|Across, DisplayLabels, LengthLimit, Limit
Numeric Format	Numeric_Format	Format, Precision|Type
Text Color	Text_Color	see Color
Text Bkg Color	Text_Bkg_Color	see Color
Text Font	Text_Font	see Font
Text Style	Text_Style	see Common Graph Object Properties
Major Grid Style	Major_Grid_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1,Red,Green,Blue
Minor Grid Style	Minor_Grid_Style	S0W1|S0W2|S0W3|S0W4|S1W1|S2W1|S3W1|S4W1|S5W1,Red,Green,Blue

See Also

Fixed Graph Object Properties

Property Reference Overview

Property Reference

Notebook Object Properties

You can use this syntax to identify notebooks (not the objects in them):

[*Notebook*].*Property*

Notebook is the name of the notebook. *Property* is the property setting to read or change.

You can use block coordinates or names to set or study block property settings. For example, this formula returns the numeric format of the block A1..A26 on page B of the active notebook:

```
@PROPERTY("B:A1..A26.Numeric_Format")
```

You can identify floating objects (graphs, macro buttons, and so on) in the active notebook by using the name of the object (found in its Object Name property). For example, if the active page contains a macro button with its Object Name property set to Button1, this formula returns that macro button's Box Type:

```
@PROPERTY("Button1.Box_Type")
```

You can also use *PageName*:*ObjectName* to identify floating objects. For example, if the object in the previous example was on page AB, this formula returns the macro button's Box Type:

```
@PROPERTY("AB:Button1.Box_Type")
```

To view properties, arguments, and syntax for the following notebook objects, choose from the list:

Bitmap

Block

Button

Graph

Notebook

OLE

See Also

Property Reference Overview

Property Reference

Bitmap Properties

Object|Property	Argument	Syntax
Bitmap		
Border Color	Border_Color	see Color
Box Type	Box_Type	None|Thin|Medium| Thick, DropShadow
Box Type	Box_Type.Frame_Line_Style	None|Thin|Medium|Thick
Drop Shadow	Box_Type.Drop_Shadow	
Object Name	Object_Name	

See Also

Notebook Object Properties

Property Reference Overview

Property Reference

Block Properties

Object|Property	Argument	Syntax
Block		
Numeric Format	Numeric_Format	Format,Precision|Type
Font	Font	see Font
Shading	Shading	Color1, Color2, Blend
Color Blend 1	Shading.Color_1	0-15
Color Blend 2	Shading.Color_2	0-15
Select Color Blend	Shading.Blend	Blend1|Blend2|Blend3|Blend4|Blend5|Blend6|Blend7
Alignment	Alignment	General|Left|Right|Center|Center Across Block
Line Drawing	Line_Drawing	Left, Top, Right, Bottom, Vert, Horiz (Each setting above can take NoChange|Clear|Thin|Thick|Double)
Protection	Protection	Protect|Unprotect
Text Color	Text_Color	0-15
Data Entry Input	Data_Entry_Input	General|Labels Only|Dates Only
Row Height	Row_Height	Operation, Size
Set Height	Row_Height	Set Height, NewSize
Reset Height	Row_Height	Reset Height
Column Width	Column_Width	Operation, WidthInTwips, ColSpacing
Set Width	Column_Width	Set Width, NewWidthInTwips
Reset Width	Column_Width	Reset Width
Auto Width	Column_Width	Auto Width, ExtraSpace
Reveal/Hide	Reveal/Hide	Row|Column, Reveal|Hide
Style *	Style	The named style of the active block
Selection * (R)	Selection	The coordinates of the selected block
Value *	Value	The contents of the cell (as they appear on the input line)
Number Value *	Number_Value	The value in the cell
String Value *	String_Value	The label in the cell

See Also

Notebook Object Properties

Property Reference Overview

Property Reference

Button Properties

Object|Property	Argument	Syntax
Button		
Macro	Macro	
Label Text	Label_Text	
Border Color	Border_Color	see Color
Box Type	Box_Type	None|Thin|Medium|Thick, DropShadow
Box Type	Box_Type.Frame_Line_Style	None|Thin|Medium|Thick
Drop Shadow	Box_Type.Drop_Shadow	
Object Name	Object_Name	

See Also

Notebook Object Properties

Property Reference Overview

Property Reference

Graph Properties

Object|Property	Argument	Syntax
Graph		
Source Graph	Source_Graph	
Border Color	Border_Color	see Color
Box Type	Box_Type	None|Thin|Medium|Thick, DropShadow
Box Type	Box_Type.Frame_Line_Style	None|Thin|Medium|Thick
Drop Shadow	Box_Type.Drop_Shadow	
Object Name	Object_Name	

See Also

Notebook Object Properties

Property Reference Overview

Property Reference

Notebook Properties

Object|Property	Argument	Syntax
Notebook		
Recalc Settings	Recalc_Settings	Automatic|Manual| Background, Natural| Column-wise|Row-wise, Iterations,<CompileFormulas?(0|1)>,<AuditErrors? 0|1)>
Zoom Factor	Zoom_Factor	25-200
Palette	Palette	Color1, Color2, ..., Color16
Color 1	Palette.Color_1	see Color
Color 2	Palette.Color_2	
Color 3	Palette.Color_3	
Color 4	Palette.Color_4	
Color 5	Palette.Color_5	
Color 6	Palette.Color_6	
Color 7	Palette.Color_7	
Color 8	Palette.Color_8	
Color 9	Palette.Color_9	
Color 10	Palette.Color_10	
Color 11	Palette.Color_11	
Color 12	Palette.Color_12	
Color 13	Palette.Color_13	
Color 14	Palette.Color_14	
Color 15	Palette.Color_15	
Color 16	Palette.Color_16	
Display	Display	VertScroll, HorScroll, Tabs
Vertical Scroll Bar	Display.Show_VerticalScroller	
Horizontal Scroll Bar	Display.Show_HorizontalScroller	
Page Tabs	Display.Show_Tabs	
Macro Library	Macro_Library	
Password Level	Password_Level	None|Low|Medium|High
System	System	Yes|No
Group Mode *	Group_Mode	On enables Group Mode; Off disables group mode
Password *	Password	The password of the active notebook

See Also

OLE Properties

Object|Property	Argument	Syntax
OLE		
Object settings	none	none
Link settings	Link_Settings	UpdateType
Border Color	Border_Color	see Color
Box Type	Box_Type	None|Thin|Medium|Thick, DropShadow
Box Type	Box_Type.Frame_Line_Style	None|Thin|Medium|Thick
Drop Shadow	Box_Type.Drop_Shadow	
Object Name	Object_Name	

See Also

Notebook Object Properties

Property Reference Overview

Property Reference

Graphs Page Icon Properties

When the Graphs page is active you can change the properties of icons on the Graphs page instead of the objects they represent. For example, this macro sets the Show Pointer property of a slide show named Presentation2 to Yes:

```
{SETOBJECTPROPERTY "Presentation2.Show_Pointer", "Yes"}
```

This table lists properties, arguments, and syntax for Graphs page objects:

Object|Property	Argument	Syntax
Graph Icon	Name	
Slide show Icon		
Name	Name	
Default Effect	Default_Effect	Effect, DisplayTime, Slow|Med|Fast, Overlay?
Show Pointer	Show_Pointer	
Dialog Icon		
Dimension	Dimension	see Dimension
Title	Title	
Position Adjust	Position_Adjust	Depend, LeftRel, TopRel, RightRel, BottomRel, CenterHor, CenterVer
Grid Options	Grid_Options	Gridsize, GridShown, GridEnabled
Name	Name	
Disabled	Disabled	
Enabled *	Enabled	
Value *	Value	

See Also

Property Reference Overview

Property Reference

Insert Dialog Box

The Insert dialog box lets you insert entire or partial rows, columns, or pages. Wherever you insert, existing data is pushed down, to the right, or to the back of the notebook to make room. The following options are available:

Block	the size of the block to be inserted; the upper left cell is the first cell entry to be shifted right, down, or back
Columns	to insert entire or partial columns, moving existing data in the active block to cells to the right
Rows	to insert entire or partial rows, moving existing data in the active block to cells below
Pages	to insert entire or partial pages, moving existing data in the active block to subsequent pages in the notebook
Entire	to insert whole rows, columns, or pages
Partial	to insert only partial rows, columns, or pages (a block)

See Also

Inserting Rows or Columns

Inserting Pages

Inserting Blocks

Inserting a File into a Notebook

Delete Dialog Box

The Delete dialog box lets you delete entire or partial rows, columns, or pages. Wherever you delete, remaining rows, columns, or pages move to take up the deleted space. The following options are available:

Block	the block to be deleted
Columns	to delete entire or partial columns, moving remaining data from cells to the right into the active block
Rows	to delete entire or partial rows, moving remaining data from cells below into the active block
Pages	to delete entire or partial pages, moving remaining data from subsequent pages in the notebook into the active block
Entire	to delete whole rows, columns, or pages
Partial	to delete only partial rows, columns, or pages (a block)

Caution: If you delete space that is within the boundaries of a named block or a block referenced by a formula, Quattro Pro adjusts all references to the block. However, if one of the deleted cells is a coordinate cell that defines a block, the block becomes invalid and any formulas or names referencing the block show ERR. Any formulas that reference a cell within a deleted column, row, or page also appear as ERR.

See Also

Deleting Rows or Columns

Deleting Pages

Deleting Blocks

Using Named Blocks

SpeedFormat Dialog Box

The SpeedFormat dialog box contains a list of formats you can choose to apply to the active block. Each format consists of a predefined set of properties for different parts of a block.

Each format applies settings for Numeric Format, Font, Shading, Text Color, Alignment, Line Drawing and Column Width (Auto-Width option) properties. To prevent SpeedFormat from applying a particular property throughout the block, uncheck that property.

In addition, each format applies properties to these parts of the active block: the Column headings, Column totals, Row headings, Row totals. To prevent SpeedFormat from applying any properties to a particular part of the block, uncheck that block part.

The Example block shows the affect the current dialog box selections will have on the active block.

See Also
Using SpeedFormat

Hotlinks Dialog Box

When you open a file that contains notebook or DDE links, Quattro Pro checks whether all supporting notebooks and/or DDE server applications are open. If not, a dialog box with these options appears:

Open Supporting	opens all supporting notebooks. If those notebooks contain links to closed notebooks, those notebooks open, too, until all supporting notebooks are open.
Update References	accesses linked values in closed supporting notebooks without opening them. This takes up less memory than opening the notebooks. If you later decide to open the supporting notebooks, choose Tools|Update Links|Open Links.
None	temporarily replaces links to closed supporting notebooks and server applications with the value NA (not available). This lets you examine or edit notebooks when you don't need the linked values. If you later want to replace the NA placeholders with the actual link values from supporting notebooks, use Tools|Update Links|Refresh Links or Open LInks to update values, or open the notebooks, respectively. You can use these commands at any time for notebook links.

Note: If you have DDE links in the notebook, selecting either Open Supporting or Update References causes the server application to open and run minimized; if you select None and later want to access DDE links, save the notebook, reopen it, and enable the DDE links when this dialog box appears.

See Also

Creating Links

Maintaining Links

Creating DDE Links

Edit Palettes Dialog Box

The Edit Palettes dialog box lets you create or change the color palettes available from the Palette List in the graph window SpeedBar. The new palette definition is based on the colors in the active palette.

Begin by modifying the color squares in the graph window SpeedBar to display the set of colors you want to include in a new or changed palette definition. See Creating Custom Color Palettes for details.

In the edit field at the top of the dialog box, type a new name for a palette, or choose an existing name to modify from the list below.

The Edit Palettes dialog box contains these options:

New	creates a new palette with the new name
Replace	permanently redefines the palette that appears in the edit field
Delete	permanently deletes the selected palette

Caution: Once you replace or delete a palette, you can recover it only by redefining the colors individually. See Creating Custom Colors for details.

Create Slide Show Button

The Create Slide Show dialog box lets you enter a name for a new slide show. If any graph icons are already selected, they are automatically included in the new slide show. After you choose OK, a slide show icon appears on the Graphs page so you can edit or run the slide show.

See Also

Creating a Slide Show
Editing a Slide Show
Running a Slide Show
Slide Show Shortcuts
Slide Show Properties

Graphs Page Light Table

The Light Table dialog box shows all the graphs in the slide show in miniature form, in the order in which they will appear. In this dialog box, you rearrange slides and edit visual effects.

To move a slide to a new position, drag the miniature of the graph to a new location. When you're rearranging slides in this way, the pointer changes to a hand. When you release the mouse button, the graph moves to the new position. The light table scrolls to reveal more slides if you drag the pointer to the top or bottom edge.

To delete a graph from the Slide Table, select it and press Del.

Options in the lower left corner of the dialog box let you control the transitions between slides as the slide show runs:

Effect	Controls the way the selected graph appears. The default transition is a quick "cut" to the next slide. See Editing a Slide Show for details on the transition effects you can choose. (The way the graph disappears from the screen depends on the Effect assigned to the next slide in the slide show.)
Display Time	Specifies how long the slide remains on the screen, in seconds (up to 3600). Enter zero when you want the slide to stay on the screen until the viewer clicks the mouse or presses a key.
Overlay	Causes the selected slide to overlay the previous one instead of replacing it
Slow/Med/Fast	Control the speed of the effect. Try each effect at the Fast speed initially, then slow it down if necessary.

Here are some tips on using visual effects:

Use wipes to simulate turning a page because the second slide appears to be underneath the first one. Use tilts to give the effect of one image pushing away another.

Avoid slower effects like the dissolve--2x2 pixels unless there's good reason to keep your viewers waiting (perhaps to increase suspense). Moderate dissolves and fade out/fade in can be restful, but fast dissolves (when run at full speed) can appear almost violent.

Use overlays to build slides from other ones. This is often effective when discussing a list of bulleted items. The first slide shows only the first bulleted item. The second slide then builds on the first by adding the second item. The subsequent slides continue building until all items are onscreen.

Take care to match the effects to the overall tone of your presentation. Showy effects like spirals, sides to center, center to sides, vertical stripes, or diamonds are best in small doses; they can look out of place in serious board rooms.

See Also

Creating a Slide Show

Running a Slide Show

Slide Show Shortcuts

Slide Show Properties

Select Slide Show Dialog Box

The Select Slide Show dialog box lets you choose a slide show to run (if you chose Graph|Slide Show or if you clicked the Run Slide Show button) or edit (if you clicked the Edit Slide Show button).

Choose a slide show name from the list and choose OK. Then the slide show runs, or the Light Table dialog box appears so you can edit the slide show.

See Also

Creating a Slide Show
Editing a Slide Show
Running a Slide Show
Slide Show Shortcuts
Slide Show Properties

File Already Exists Dialog Box

You are about to replace a file on disk with the file currently displayed. Your options are:

Replace	Overwite the file on disk with the file you are working on.
Backup	Make a backup copy of the file on disk (with a .BAK extension); then replace the file on disk with the file you are working on.
Cancel	Abandon the attempt to save the file you have been working on.

File Not Found Dialog Box

Quattro Pro was unable to find the file you specified in the notebook link. Use this dialog box to locate the file you want to link to. See File-Handling Options for an explanation of the dialog box's controls.

Read/Write Formatted File Dialog Box

You are attempting to load or save a file that is currently in Impress or Allways format. Your options are as follows:

Impress	Loads or saves the file in Impress format, and maintains the file's page layout settings
Allways	Loads or saves the file in Allways format, and maintains the file's page layout settings
None	Loads or saves the file in standard .WK1 format, and abandons any page layout specified in the Impress or Allways file

Application Basics

The following topics introduce key concepts and skills for all developers of custom applications.

What's an Application

A Quattro Pro application is a combination of components that work together to make a task easier. The final look and feel of an application can be radically different from the standard Quattro Pro notebook.

When you build an application, you can create custom components, such as:

- dialog boxes
- SpeedBars
- menus

After you create these components, you assemble them into an integrated application notebook. This application notebook contains the macros, dialog boxes, and menus that make up the application.

The Developer vs. the User

Notice that the Application Building sections of Quattro Pro documentation address "you" as the developer:

- Developer refers to the person who is building an application, and who is interested in creating an application.
- User refers to the person who runs a completed application, and who is interested in using the application to perform tasks.

See Also

Looking at a Sample Application

Dialog Window Properties

Looking at a Sample Application

To get a sense of how components work within a Quattro Pro application, open BUDGTRAK.WB1, a sample application in your Quattro Pro program directory.

You can see how different this application's user interface (UI) is from the standard Quattro Pro UI. The screen differs from Quattro Pro in these ways:

- The Budgeteer appears in the title bar
- A custom dialog box appears onscreen
- The names of menus on the menu bar are new
- A custom SpeedBar resides below the menu bar

The following topics explain the Budgeteer application in detail:

Using the Budgeteer

Budgeteer Menus

Budgeteer SpeedBar

How the Application Works

See Also

Dialog Window Properties

Using the Budgeteer

The Budgeteer manages a set of databases stored in a separate notebook. This notebook is called a budget file, and is created by the Budgeteer. Each database in the budget file stores a set of expenses. Each database contains expenses that occur at a certain interval. For example, if you (as a user) pay your car insurance annually, that expense is stored in the Annual database; monthly expenses like groceries are stored in the Monthly database, and so on. The Budgeteer lets you set up an expense-tracking system without having to understand databases.

After loading a budget file (using Budget|Open or the opening screen of the Budgeteer) a record window is displayed showing the records in the database you're working on (by default, the Monthly expense database is displayed). If the database being viewed is empty (or you've created a new budget file using the opening screen or Budget|New), the window is empty. To specify the database to work on, use View on the SpeedBar.

You can use the Budgeteer SpeedBar to specify the expense database to work with and to search for specific records in that database. Commands on the Budgeteer menu bar let you save, load, and print budget files. Click one of the following topics for specific directions on performing basic Budgeteer tasks.

Creating a New Budgeteer File

Adding Expenses to a Budgeteer Database

Adding Budgeteer Records

Viewing a Budgeteer Expense Database

Searching for Specific Budgeteer Records

See Also

Budgeteer Menus

Budgeteer SpeedBar

Dialog Window Properties

Creating a New Budgeteer File

To create a new budget file,

1. Load the Budgeteer and choose New from the initial dialog box. This displays an empty record window. (You can also choose Budget|New if the Budgeteer is already running.)
2. Use Budget|Add Expense to add expenses to the databases in this budget file (see Adding Expenses to a Budgeteer Database for details).
3. Use View and Add on the Budgeteer SpeedBar to add records to the databases in this budget file.
4. Choose Budget|Save As, then enter a name for the budget file.
5. Choose OK to save the budget file. By default, budget files are saved with the file extension BDG.

See Also

Budgeteer Menus

Budgeteer SpeedBar

Dialog Window Properties

Adding Expenses to a Budgeteer Database

You can use Budget|Add Expense to add expenses to any database in the active budget file. To add an expense,

1. Choose Budget|Add Expense.
2. Click a button in the Frequency group box to choose the name of the database you want to add an expense to. By default, the database being viewed is used.
3. Choose New Item and enter the name of the expense to add. Current Items shows the existing expenses.
4. Choose OK to add the expense to the database.
5. Save the budget file using Budget|Save or Budget|Save As.

See Also

Budgeteer Menus

Budgeteer SpeedBar

Dialog Window Properties

Adding Budgeteer Records

To add records to a database,

1. Use View on the Budgeteer SpeedBar to specify the database you want to add a record to.
2. Choose Add from the Budgeteer SpeedBar. This displays a block containing the names of all expenses in the database. Date is also added to specify the entry date of the record; the Budgeteer uses this date when searching databases. It defaults to the current system date.
3. In the cell below each expense name, enter the cost of that expense. You can use arrow keys to move between the expenses.
4. To add the record to the expense database, press Enter (with nothing on the input line) or Esc.
5. Repeat the previous four steps for each record you want to add.
6. Use Budget|Save As or Budget|Save to save your changes.

See Also

Budgeteer Menus
Budgeteer SpeedBar
Dialog Window Properties

Viewing a Budgeteer Expense Database

The Budgeteer displays a record window that shows the records in an expense database. You can use the Budgeteer SpeedBar to view a different database (using View) or reduce the number of records that display (using Day, Month, and Year, discussed next). You can use the scroll bars to scroll through records.

To view expense databases in a different budget file, use Budget|Open to load the budget file containing the databases.

See Also

Budgeteer Menus

Budgeteer SpeedBar

Dialog Window Properties

Searching for Specific Budgeteer Records

By default, the Budgeteer shows you all the records in the database specified by View. You can use Month, Day, and Year to view records entered during a specific time period; if you don't want one of these controls to affect the search, set that control to All. For example, to view only records entered in June of 1993,

1. Set Month to June.
2. Set Day to All to ensure that it doesn't affect the search.
3. Set Year to 1993.

The Budgeteer automatically updates the record window to display only those records that meet the search criteria specified in the SpeedBar. You can graph the information in these records (using Graph on the SpeedBar), print them in a report (using Print|Current), or display statistics about them (using Statistics).

See Also

Budgeteer Menus

Budgeteer SpeedBar

Dialog Window Properties

Budgeteer Menus

The Budgeteer menu bar provides commands to perform global operations like saving budget files, printing, and displaying statistics. For example, here's what the Budget menu commands do:

New	creates a new budget.
Open	loads an existing budget file.
Save	writes current data to the budget file.
Save As	writes current data to a different budget file.
Add Expense	creates a new expense column. You can use this command to add expense categories to any database in the active budget file, not just the database being viewed.
Quit	exits the application and restores standard Quattro Pro settings.

You can use commands on the Print menu to print the contents of the active budget file. You can use Print|Current to print the records currently displaying in the record window, or All to print the contents of all expense databases in the active budget file.

The Statistics command displays statistics on the records displaying in the record window, including the total expense cost, average expense cost, greatest cost, and smallest cost.

See Also

Using the Budgeteer

Budgeteer SpeedBar

Dialog Window Properties

Budgeteer SpeedBar

When the Budgeteer is active, the standard Quattro Pro SpeedBar is replaced by the Budgeteer SpeedBar, which lets users display the precise view of the data they need.

Here's what the SpeedBar options do:

View	specifies the database to display in the record window. You can use Month, Day, and Year to search for and display specific records.
Month	lets you display records entered during a specific month. For example, set it to June to display any records entered for June. To view all months, set Month to All.
Day	lets you display records entered on a specific day of the week. For example, set it to Tuesday to display any records entered on a Tuesday. To view all days, set Day to All.
Year	lets you display records entered during a specific year. For example, set it to 1994 to display any records entered during 1994. To view all years, set Year to All.
Add	adds a new record to the expense database being viewed. Remember, a new record only displays if it meets the search conditions specified by the SpeedBar.
Graph	creates a graph using the records in the record window. Each expense is plotted as a separate series.

Note: A budget file contains multiple expense databases. Use View to specify which database you're working on.

See Also

Using the Budgeteer

Budgeteer Menus

Dialog Window Properties

How the Sample Application Works

When you finish exploring the Budgeteer as a user, choose Budget|Quit to close the application. Notice that the standard Quattro Pro UI is restored.

You can stop the Budgeteer's autoload macro before it runs to study the application as a developer and see how it works. To stop the autoload macro:

1. Choose Tools|Macro|Debugger.
2. Open BUDGTRAK.WB1 using File|Open or File|Retrieve.
3. Choose Terminate from the debug window's menu bar to stop the autoload macro that runs the Budgeteer.

The Budgeteer notebook is ready for your review. One advantage of notebooks is that you can organize the components of your application. The following table lists some of the pages of the Budgeteer and their purpose.

Page(s)	Purpose
Startup	This page contains the macro that changes the Quattro Pro UI and runs the Budgeteer. It also contains the macro that closes down the Budgeteer.
Menus	This page contains the menu block that defines the Budgeteer menu bar.
Help	This page contains help messages displayed by the Budgeteer, and developer notes.
*Task*_Macros	These pages contain macros specific to a Budgeteer function. For example, ChangeDB_Macros contains macros to change expense databases; Query_Macros contains the macros to search expense databases.
Expense_View	This page is the record window that the user sees. It's in the budget file.
Stats	This page is the window that the user sees after choosing Statistics. It's in the budget file.
Input	This page contains the form that users manipulate to add records. It's in the budget file.
Daily...Annually	These pages contain the expense databases when the Budgeteer is running. Each page contains one database. They're in the budget file.
Graphs	This page contains the dialog boxes used by the Budgeteer. (Discussed next.)

The Budgeteer uses several dialog boxes, as listed in the next table.

Dialog Box	Purpose
AddExpense	Displayed by Budget|Add Expense. It lets the user view expenses in a specific database and add a new expense.
InitOpen	Displayed by many commands on the Budget menu. It's used to get the name of a file to load or save. The title in this box is often changed by macro commands. The New button is hidden by macro commands when it's not needed.
Warn	Displayed by many Budgeteer commands. It's used to confirm some action the user is about to perform. The warning message is often changed by macro commands.

Many of the macro commands in the Budgeteer notebook have comments listed to their right. These comments help explain how the Budgeteer works.

Note: You can study the Budgeteer SpeedBar by loading BUDGTRAK.BAR, a file in your Quattro Pro program directory.

See Also

Using the Budgeteer

Dialog Window Properties

Understanding Application Components

In a Quattro Pro application, each component is used for a specific purpose:

Dialog boxes prompt users for information and let the user make choices.

SpeedBars give users shortcuts to common application tasks and perform operations that affect the active data. They also display information about the active window or object.

Menus provide custom lists of operations the user can choose from the menu bar. Menu commands usually perform more global operations (like saving a file).

You use macros to assemble these components into an application. You can also use macros to automate Quattro Pro operations, change Quattro Pro's appearance, and display dialog boxes.

You can also use link commands to assign actions or connections to controls in a dialog box or a SpeedBar. A link command can get or change an object's property settings, or run macros.

Each part of an application involves specific design issues, as described in the following topics:

Designing Dialog Boxes

Designing SpeedBars

Designing Menus

Designing Macros

See Also
Dialog Window Properties

Designing Dialog Boxes

When you want to create a dialog box (or a custom SpeedBar) you can choose Tools|UI Builder to work in a dialog window. When a dialog window is active, the SpeedBar changes to provide tools that create dialog controls, objects you can create in a dialog box or a SpeedBar.

Displaying Dialog Boxes

After you create a custom dialog box, you can display it using the {DODIALOG} macro command. The most important argument in {DODIALOG} is the third argument, which is a block that contains the initial settings of the dialog box controls.

Using Link Commands

Link commands perform dynamic actions in response to events, which are things the user does in an application. For example, a link command can be assigned to a control so it will get an object property when the control is clicked, or display additional controls.

Events recognized by Quattro Pro include

 clicking

 right-clicking

 pressing a key

 entering a new value

Link commands also let you quickly connect a control to another control or a notebook cell.

See Also

Building Custom SpeedBars

Dialog Window Properties

Designing SpeedBars

When specific tasks are frequently repeated in an application, you can create a custom SpeedBar to provide user shortcuts. SpeedBars are also suited to operations that affect the current data, and can display information about the current data. For example, the Budgeteer SpeedBar contains common database operations such as defining a data query and graphing the data; it also displays the name of the database being viewed.

A SpeedBar is a special type of dialog box; here's how SpeedBars are different from dialog boxes:

SpeedBars only display at the top of the Quattro Pro window.

Titles aren't displayed in SpeedBars.

SpeedBars are saved as separate files with a .BAR extension.

The {DODIALOG} macro command can't display a SpeedBar.

Displaying SpeedBars

After you create a custom SpeedBar, you can use the SpeedBar Control menu (right-click the standard SpeedBar any place outside a button or field) or the macro commands {Application.SpeedBar} or {SpeedBar.*Option*} to display the SpeedBar in an application. You can either display a custom SpeedBar below the standard Quattro Pro SpeedBar, or by itself, without the standard Quattro Pro SpeedBar.

Displaying a custom SpeedBar is a two-step process. First you hide the standard Quattro Pro SpeedBar, then you display your custom SpeedBar. See Creating a SpeedBar for details on creating, removing, and displaying SpeedBars using Tools|UI Builder. (You can also create SpeedBars using Tools|SpeedBar Designer; see Building Custom SpeedBars.)

See Also

{DODIALOG}

Dialog Window Properties

Designing Menus

The operations performed by SpeedBars often overlap with the operations performed by menus; try to use menus for operations that affect the data in a global way (like saving a file, printing a database, and so on).

You can add custom menus and menu commands to the menu bar, and replace the menu bar with one command. Menu commands can

- initiate actions
- run macros
- display dialog boxes
- display other menus

To create a menu, you describe the menu with a block of labels.The following table lists macro commands you can use to modify menus, menu commands, and the menu bar:

Macro Command	Description
{DELETEMENU}	Removes a menu from the current menu bar (including submenus like Tools|Macro).
{DELETEMENUITEM}	Removes menu commands from the specified menu.
{ADDMENU}	Adds names to the current menu bar.
{ADDMENUITEM}	Adds menu commands to the specified menu.
{SETMENUBAR}	Replaces the active menu bar with the menu described by the block.
{SETOBJECTPROPERTY}	Changes existing menu commands.

See Also
Dialog Window Properties

Designing Macros

Macros are the heart of every application; they're the primary method you use to pull all of your application components together. In addition to the {DODIALOG} macro command, which displays dialog boxes, macros are most commonly used in an application to

- automate tasks
- duplicate keyboard and mouse actions
- change the active menus
- affect the screen display
- select, reposition, and resize objects
- prompt users for input

The remainder of this topic lists some of the uses of macros in an application.

Notebook Startup Macros

Developers typically use a notebook startup macro to set up the application and present a unique look for the application. For example, the startup macro in the Budgeteer hides parts of the standard Quattro Pro UI, changes the menu bar to the Budgeteer menus, and displays the initial Budgeteer dialog box.

Changing Properties with Macro Commands

Quattro Pro offers numerous macro commands that developers can use to modify the standard look of Quattro Pro or to change property settings.

Some of these macro commands are

- {Application.Display}
- {GETOBJECTPROPERTY}
- {GETPROPERTY}
- {Notebook.Display}
- {Page.Borders}
- {Page.Grid_Lines}


{SETOBJECTPROPERTY}


{SETPROPERTY}

See Also
Dialog Window Properties

Preparing to Build an Application

As a developer, it can be hard to resist jumping right in and building an application. But before doing this, take the time to carefully plan and design the application. This approach makes the application easier to debug and revise.

Planning an Application

As a developer, consider these fundamental issues:

User needs. Start by determining what tasks users need to accomplish. Encourage users to suggest ways to make the proposed application useful to them.

Examples. Find existing features or applications in other products that perform tasks similar to the ones you plan to design. Ask users if they can think of any way to improve these examples.

Simplicity. Try to make the application as easy to use as possible. Try to provide help messages and help lines for users within the application.

Design guidelines

As a developer, consider using the following tips:

Involve users. Let users review your work on the application when early working versions are ready. Use their suggestions to improve and refine the design.

Think small. Keep all macros, dialog boxes, and menus for a single application organized in one notebook. Also, try to limit the number of supporting files (notebooks, SpeedBars, graphics, text, and so on). This way, you'll have fewer files to install and maintain on user's computers.

Be consistent. Define and follow a set of rules for naming and storing components of your applications. For example, you could begin all component names with the same three-letter code. Also, keep your application files in a specific directory and instruct your users to do the same.

Document your work. Since you might need to update or revise the application later, be sure to include development notes in the application. For example, you can add specific comments and labels to macros, or general comments for an entire notebook page of menu items.

Protect yourself. Limit user access to critical components of the application by hiding UI elements such as the tab cluster or by protecting cells from editing (see Enabling Block Protection for details). This will keep unexpected maintenance of the application to a minimum.

Avoid traps. If you change the standard menu bar, you might be unable to run a macro by choosing Tools|Macro|Execute. When developing applications, run Quattro Pro in Developer mode (see Using Developer Mode); then you can press Ctrl+Shift+N to restore the standard menu bar and reveal hidden UI components. You can also write a macro to restore Quattro Pro to its original state and assign the macro to a key (see Executing Macros for details). Handy macros like this one can be saved to a macro library, and the macro library can be open when testing or developing.

Clean up. Remember to restore the standard Quattro Pro settings after users exit your application.

See Also

Dialog Window Properties

Using Developer Mode

Starting Quattro Pro with the command line switch /D runs Quattro Pro in Developer mode. Developer mode adds special properties to Object Inspector menus and lets you use a shortcut key, Ctrl+Shift+N, as described in the following tables. You don't have to be in Developer mode to change or read these property settings; macro commands can always access them.

The application Object Inspector gains two additional properties in Developer mode.

Property	Description
Enable Inspection	Lets you disable right-clicking, the Property menu, and the Object Properties key (F12). Use this to disable Object Inspector menus.
Title	Lets you change the text that displays in the Quattro Pro title bar.

In Developer mode, all drawn objects in a graph window gain the following properties.

Property	Description
Dimension	Lets you move and resize graph objects created with the SpeedBar in the graph window.
Name	Used by macro commands, link commands, and @functions to read or change property settings.

For a list of all Quattro Pro object properties (including more hidden properties), see Property Reference.

The Developer mode shortcut key, Ctrl+Shift+N, varies in function depending on the current program state:

State of program	Description
Debugging a macro	Stops macro execution and exits Debug mode.
All other states	Restores the standard Quattro Pro menu bar, status line, input line, and SpeedBar. Also sets the Enable Inspection property back to Yes, which re-enables Object Inspector menus.

See Also
Dialog Window Properties

Advanced Editing

Grouping Notebook Pages

You can act on several notebook pages at the same time by grouping the pages. Then any changes you make to one page affect all pages in the group. For example, you can enter column headings and make formatting changes on multiple pages simultaneously by first grouping the pages.

After you create a group, turn it on and off with the Group button next to the page tabs. The group of pages is indicated by a blue line under the page tabs in the group.

Creating a Group

Group Mode Tips

See Also
Using Named Blocks

Creating a Group

To create a group of notebook pages,

1. Choose Tools|Define Group.
2. Enter the first and last pages of the group in the dialog box. If you preselected a series of pages before choosing the command, they appear in the dialog box.
3. Type a name for the group and choose OK. You can use letters and numbers in the name, as well as the following special characters:

   ~ ` ! % _ | \ ' ?

   You cannot use spaces or any other special characters. Also, you cannot use a name you have previously assigned to a page in the same notebook.
4. To activate this group, click the Group button. A blue line appears below the tabs in the group. From now on, changes made to blocks in the group affect all similar blocks in other pages in the group.

Also, if you point to blocks in the group from dialog boxes or in formulas while Group mode is on, the group name is used. For example, a formula would read `1stQtr:A1:A10` instead of `Jan..Mar:A1:A10`.

To deactivate the group, click the Group button again. The blue line disappears.

You can have multiple groups in one notebook. Changes to one page in a group affect only the other pages in the same group while you are in Group mode. You cannot have overlapping groups; that is, any given page can belong only to one group.

To delete a group, choose Tools|Define Group. Then choose the group name and choose Delete. Choose OK.

Using a Group

Any 2-D selection that you make on a page turns into a 3-D selection when you use Group mode.

For example, if you group pages A through D of the same notebook, you can apply a heading style and a currency format style to all pages at once. Just select A1 on any page and choosing Heading 2 from the Style list, and then select A2..B3 on any page and choose Currency from the Style list.

Entering Data into a Group

You can "drill" an entry into the same cell in all grouped pages by combining grouping with the Ctrl and Enter keys.

To drill the label "Income" into cell D1 on pages A through C, select D1 on page A, group pages A through C, and type the label. Instead of pressing Enter to complete the entry, hold down the Ctrl key while you press Enter.

You can also delete through the pages in a group with Ctrl+Del. Just select a cell or block on any page in a group while Group mode is on. All entries in corresponding cells in the group are deleted.

See Also

Group Mode Tips

Using Named Blocks

Group Mode Tips

Remember that every action you take affects all pages in the group. This means that many operations require you to switch Group mode on and off.

For example, to copy data from one page in a group to the remaining pages in the group,

1. Click the Group button to turn off Group mode.
2. Select the source data to be copied, and click the Copy button in the SpeedBar.
3. Click the Group button to turn Group mode back on.
4. Select the destination cell in any page in the group, and click the Paste button in the SpeedBar.

You can also turn Group mode on and off while writing formulas that involve 2-D and 3-D references. Watch the input line as you point to references; with Group mode on, you get 3-D references, with Group mode off, you point to 2-D references.

The SpeedSum button is ideally suited to use with groups. Just select the cells on any page in the group to create the desired totals on that page, then click SpeedSum. The appropriate formula is written into each page in the group.

Other operations that are particularly useful in Group mode are:

- most block property settings
- most page property settings
- column width and row height changes, either with the mouse or the Fit button
- block move and copy within similarly formatted pages
- inserting and deleting rows and columns

Caution: Be careful when using Edit|Cut, Clear, or Clear Contents when Group mode is on. Data will be deleted from all pages in the group.

See Also

Creating a Group

Using Named Blocks

Variations on Pasting

Quattro Pro uses the Windows Clipboard to copy data and paste it in various forms. By default, using Edit|Copy or Cut followed by Edit|Paste places the most complete version of the data into the pasted area whether it comes from another application or from within Quattro Pro itself.

Sometimes you may want to paste only a subset of the data. You can do this in the following ways:

With Quattro Pro cell data in the Clipboard, you can use Edit|Paste Special to paste only a block's properties instead of its values, or vice versa (see Using Paste Special for details).

With data in the Clipboard from a participating DDE application, you can use Edit|Paste Link to create a live data link. This type of link causes data changes from the other application to be constantly reflected in Quattro Pro (see Creating DDE Links for more information).

With data from many other types of applications in the Clipboard, you can use Edit|Paste or Paste Format to create many kinds of data types, including embedded OLE objects, picture objects, and bitmap objects (see Choosing the Data Type to Paste for details). Many of these types of data reside in Quattro Pro objects that float above the surface of the page, as floating graphs do (see Creating Floating Objects for details).

Without using the Clipboard, you can run an OLE server application from within Quattro Pro to create embedded OLE objects.

You can, of course, paste Quattro Pro data into other applications, as limited by the formats the other application can read. In addition, this section describes how to copy images of blocks of cells into bitmap format and paste them into other applications.

See Also

Dragging and Dropping

Copying and Moving with the Windows Clipboard

Copying Formulas

Using Block Commands for Copying and Moving

Using Paste Special

Ordinarily, when you paste a block, you paste the source block's properties too (such as its font, alignment, and numeric format). To paste the data without the properties, or vice-versa, use Edit|Paste Special.

You can also use Paste Special to transpose rows and columns or to avoid copying blank cells. Edit|Paste Special is available only when you're pasting Quattro Pro for Windows data from the Clipboard.

To control the paste operation,

1. Use Edit|Copy to copy a selection to the Clipboard.
2. Select the destination for the pasted cells.

Caution: If the destination already contains data, you'll overwrite it when you choose OK.

3. Choose Edit|Paste Special.
4. Check the options you want:


Properties pastes the properties from the copied selection.


Contents pastes the data from the copied selection. If you choose Contents, the default setting is Formulas. If you choose Values Only, only the values resulting from the formulas are pasted. Choosing Values Only accomplishes the same thing as using Block|Values (see Converting Formulas to Values for details).


Transpose rows and columns switches the position of entries so that data listed in columns is placed in rows and vice versa. Be sure to allow enough room in the destination area for the changed size of the data. This option is equivalent to choosing Block|Transpose. See Transposing Columns and Rows for an example and further explanation.


Avoid Pasting Blanks avoids erasing data in the destination selection that otherwise would be replaced by blanks pasted from the source.

5. Choose OK.

The source selection is pasted into the destination.

See Also

Creating DDE Links

Choosing the Data Type to Paste

Creating DDE Links

Pasting data from other Windows applications that use DDE lets you set up live links. Links are live when changes made to one application are automatically reflected in the other. Quattro Pro can either receive data, acting as the client application, or send data, acting as the server application.

To create a DDE link from a DDE server application to Quattro Pro, copy the data to the Clipboard in the other application. Then, in Quattro Pro, select the location for pasting the link and choose Edit| Paste Link.

To create a DDE link from Quattro Pro to a DDE client application, copy the graph or data with the Copy button in the SpeedBar. Then follow the instructions in the documentation for the client application to paste the link. Formatted data can be pasted in RTF (Rich Text Format) format to preserve existing type styles, sizes, and column widths.

When you open a file that contains DDE or notebook links, Quattro Pro checks whether all supporting DDE server applications and/or notebooks are open. If not, the Hotlinks Dialog Box appears.

See Also

Pasting Links

Choosing the Data Type to Paste

OLE and DDE Overview

Choosing the Data Type to Paste

Ordinarily, using Edit|Paste puts the most complete version of Clipboard data into Quattro Pro. If you want to create a particular type of data, use Edit|Paste Format to choose from these types:


BIFF3. Excel format data.


WK1. Lotus 1-2-3 versions 2.x format data.


Embedded OLE. A floating object containing data linked from another application.


Metafile. A graphic format that creates a floating picture object in a notebook.


Bitmap. A graphic format that creates a floating bitmap object in a notebook.


Text. Text data is entered as labels into cells, and numeric data is entered as values.


Linked OLE. A floating object displaying data linked from another application.


Link. A DDE link, which is the same as the link created with Edit|Paste Link.


WK3. Lotus 1-2-3 version 3.x format data.


Paradox Table. Paradox for Windows or Database Desktop data.


DIB. A Device Independent Bitmap format that creates a floating bitmap object.

Not all of these formats are available in all situations. Depending on the data copied to the Clipboard and the location within Quattro Pro that's selected, only a subset of the list of formats may be available.

To paste in a particular format, copy the data from the other application to the Clipboard. Then, in Quattro Pro, select the location to paste the data, and choose Edit|Paste Format. After you select the format type from the list, choose OK. Depending on the data type, the data is entered into cells, or as a floating object above the surface of the notebook page.

The Embedded OLE and Linked OLE formats create similar objects in Quattro Pro. The difference is that you'd create an embedded OLE object if you want to store the data for the object within a Quattro Pro file. Creating a linked OLE object leaves the data stored in the server application file. For both types of objects, you can restart the server application to further manipulate the object by double-clicking the floating object.

See Also

Creating Floating Objects

Inserting OLE Objects

Using Paste Special

Creating DDE Links
OLE and DDE Overview

Creating Floating Objects

You can create floating objects containing a variety of data types. These objects exist in a layer above the spreadsheet cells, but are part of the spreadsheet page in which they appear. They obscure entries in cells underneath them, but you can move a floating object to reveal cells underneath by selecting it and dragging it with the mouse.

Although a floating object isn't attached to particular cell borders, it prints when the block underneath it is selected, and it is saved with the notebook.

You can create floating objects by using Edit|Insert Object, by pasting Clipboard data, by using Graph|Insert, or by using tools in the SpeedBar.

Edit|Insert Object creates an embedded OLE object created by a OLE server application that you run from within Quattro Pro.

By pasting Clipboard data, you can create these types of objects:


Embedded OLE or linked OLE objects


Picture objects


Bitmap objects

Using tools in the SpeedBar, you can create these types of objects:


Floating graphs created with the Graph tool (see Creating a Floating Graph for details)


SpeedButtons created with the SpeedButton tool (see Creating Macro SpeedButtons for details)

You can also create floating graphs with Graph|Insert.

Note: Another way to create a floating object that contains a bitmap or metafile image is to import the file into a graph window and insert the contents as a floating graph. You can import files in many popular graphics formats. See Importing Graphics for details.

See Also

Inserting OLE Objects

Using Paste Special

Creating DDE Links

OLE and DDE Overview

Inserting OLE Objects

Quattro Pro can act as a client for OLE server applications. This means that you can use OLE server programs from within Quattro Pro to create and insert images linked into Quattro Pro notebooks. You can do this with Edit|Paste Format, but an easier way is with Edit|Insert Object.

This is the quickest way to create an embedded OLE object in Quattro Pro as long as one of the available servers will produce the data you want. To insert an object using an OLE server,

1. Select a cell in the spreadsheet page approximately where you want to place the object (you can always move it later).
2. Choose Edit|Insert Object. A list of object types you can create appears. This is the list of OLE servers available on your system.
3. Choose one of the listed OLE servers, then choose OK.

   A new window opens in which the OLE server is running. Use the commands in its title bar to create and manipulate the data you want.
4. When you're finished, there's no need to save the file or copy the data to the Clipboard; just choose File|Update or File|Exit.

An embedded OLE object appears in the spreadsheet page, and the OLE server closes. Now you can move or resize the object or change its properties.

Moving and Resizing Floating Objects

Once a floating object is selected, you can resize it or move it. Click anywhere in an object to select it; handles appear around the border.

To resize only the height or width of an object, drag a side handle. To resize height and width at the same time, drag a corner handle diagonally.

To move an object, drag it from any area except its handles.

See Also

Creating Floating Objects

Changing OLE Object Properties

Using Paste Special

Creating DDE Links

OLE and DDE Overview

Changing OLE Object Properties

You can right-click floating objects and change their properties. The Object Inspector's title bar identifies the object type. All floating objects contain the Border Color, Box Type, and Object Name properties. OLE objects each have one additional property, called Object Settings (for embedded OLE objects) and Link Settings (for linked OLE objects).

To change embedded OLE object properties, right-click the object, choose the Properties command, then choose Object Settings. The options are


Change to picture. Use this to convert an embedded OLE object to a picture object. Once this has been done, you will no longer be able to edit the object.


Edit, Play, or whatever action is appropriate to the server application. You can run a video, play a sound, edit the object, and so on. When you are finished working in the server application, choose File|Update or File|Exit.

To change linked OLE object properties, right-click the object, choose the Properties command and choose Link Settings. The options are


Update method. Choose Automatic to make data always reflect edits in the server application. Choose Manual to update data only when Update Now is chosen.


Edit. See the previous explanation for embedded OLE objects.


Update now. Choose this when the Manual update method is chosen to copy the latest data from the server application.


Unlink. Choose this to remove the link from the server application. This converts the OLE object to a picture object, which you can no longer edit.


Change link. If the file to which you are linked is moved to different directory, choose this to relink to the file in its new location.

See Also

Graph Object

File-Handling Options

Macros

Layering Floating Objects

After pasting some types of data, a floating object appears in the notebook page. The types of objects created by pasting data are

- Embedded OLE or linked OLE objects
- Picture objects
- Bitmap objects

You can also create floating objects with these tools in the SpeedBar:

- Create floating graphs with the graph tool
- Create SpeedButtons with the SpeedButton tool

As you include additional floating objects on a page, you can place them in layers one above the other. To manage the placement of floating objects, use Block|Object Order.

To reorder the layering of overlapping objects, select an object--it has handles around it when selected--and choose one of the Block|Object Order commands:

- Bring Forward moves the object one layer closer to the top.
- Send Backward moves the object one layer closer to the bottom.
- Bring to Front brings the object to the front or top of the layers.
- Send to Back sends the object to the bottom, or the layer closest to the page.

See Also

Building Graphs

Macros

Copying Quattro Pro Images

When you copy a block to the Clipboard, it is automatically stored in bitmap format in addition to the usual data format. This means you can paste this bitmap image to another application, such as Windows Paintbrush, and enhance it further.

To paste a Quattro Pro bitmap image into Paintbrush:

 Select the block you want to include in the image. The block must be contiguous.

 Click the Copy button in the SpeedBar. (Make no further changes in Quattro Pro until you paste this image.)

 Start Paintbrush (or activate it if it's already started).

 Choose Edit|Paste.

The image appears in Paintbrush just as it looks in Quattro Pro.

See Also

Creating Floating Objects

Using Paste Special

Defining Styles

A Quattro Pro style is a combination of Block property settings gathered under a style name. You apply a style by selecting it from the Style list in the SpeedBar. The predefined styles are described in Using Styles.

You can create entirely new styles, or you can modify Quattro Pro's predefined styles. In either case, you can copy property settings from a formatted cell or from an existing style to add to your new style.

To revise or create a style,

1. Choose Edit|Define Style.
2. To revise a style, choose it from the Define Style For list. To create a new style, type a new name in the Define Style For edit field.
3. To add or modify properties that make up the style, check the box next to the property name. Then choose the property button, modify the setting, and choose OK. To remove a property from the style definition, uncheck the box next to it.

   Note: Only checked properties are included in the style you are defining. The unchecked properties will have no effect in the cells to which you apply this new style. This lets you create partial styles that affect only a few properties at a time and apply them in combination with properties already existing in cells.
4. To copy property settings from a preformatted cell or another style, choose Merge. Then in the Merge Style dialog box, choose Style and choose the style name from the Select Style box to merge another style. Or, choose Cell and point to or type the cell address in the Select Cell box. The choose OK to return to the Define/Modify Style dialog box.
5. Choose OK to complete the style definition.

To use the new style, select the cells to which you want to apply it and select the name from the Style list in the SpeedBar.

If you redefine a style that is already in use in the notebook (such as Normal, which is applied to all cells by default), all cells using that style change to reflect the new property settings. Cells in which you set individual properties with the block Object Inspector don't change. This is because styles act as default settings that are overridden by block property settings.

To delete a style, choose Edit|Define Style, and choose the style name in the Define Style For box. Then choose Delete and choose OK. Cells to which you had applied the deleted style revert to Normal style. You can delete any style except the Normal style.

To copy a style to another notebook, copy a cell formatted in that style, and paste it into the new notebook. The style name then appears in the Style list for that notebook.

See Also

Active Block Properties

Object Inspection

Defining Custom Numeric Formats

In addition to the numeric formats built into Quattro Pro, you can create your own numeric formats to further customize the appearance of numbers, dates, and times. This process requires the use of numeric format codes, which are discussed in the following topics:

Using Numeric Format Codes

Number Format Symbols

Date and Time Format Symbols

Miscellaneous Format Symbols

To create a numeric format,

1. Right-click the block to which you first want to apply the format and choose Block Properties. Numeric Format is already selected.

   Choose User Defined and then any format from the Formats Defined list. This format will be the basis from which you create the new format.
2. Enter a format code, using the symbols and syntax described in the next section.
3. Choose OK to complete the format and apply it to the active block at the same time.

   To apply the new format to a block, right-click the block and choose Block Properties, then choose Numeric Format. Check User Defined and choose the new format from the Formats Defined list.

Saving Custom Formats

Every time you create a User Defined format, it's saved in the Formats Defined list in the User Defined option of the block Numeric Format property.

The format is also added to the list of all open notebooks. It is saved with any of these additional notebooks only if you apply the format to a cell in that notebook.

If you paste a formatted cell into another notebook, the custom format definition is copied also.

See Also

Block Properties

Using Numeric Format Codes

Using special codes, you can create custom formats to display values. You can also create formats that display values differently depending on whether they're positive, negative, or zero. Finally, you can add text at the beginning, middle, or end of your format to be displayed with whatever value is in the formatted cell.

Formats are divided into two groups: Number formats and Date/Time formats. You create these formats using codes that represent formatted values in each character position in the formatted cell.

You can add any other characters, which will be inserted "as is" in the displayed entry within the formatted data.

The following code displays a number to three decimal places:

```
N9.000
```

For example, 53.97 would appear as 53.970. The N code begins all Number formats. The 9 code displays as many digits as there are in the value to the left of the decimal point. The zero codes require that three digits to the right of the decimal point be displayed.

The next code displays a date and time:

```
TWeekday, H:M:S
```

For example, 1/9/92 at 2:30am would appear as Thursday, 2:30:0. The T code begins all Date/Time formats. The Weekday code displays the name of the day, and the H, M, and S codes display the hour, minute, and second. The colons aren't codes at all; they are simply inserted into the formatted value according to their position in the format.

See Also

Number Format Symbols

Date and Time Format Symbols

Miscellaneous Format Symbols

Number Format Symbols

Symbol	Action
N or n	Begins a Number format (as opposed to a Date or Time format).
0	Always displays a digit. If the number doesn't have a digit in this position, a 0 appears. If there are more digits in the number than 0s in the format code, extra digits in the integer portion are displayed, and the fraction portion is rounded.
9	Displays a digit if the number has one. If there are more digits in the number than 9s in the format code, extra digits in the integer portion are displayed, and the fraction portion is rounded.
%	Displays the number as a percentage.
, (comma)	Inserts a thousands separator (a comma unless specified otherwise with the application International property). If not part of a Number code, the comma is inserted wherever it's positioned in the format.
. (period)	Inserts a decimal separator, which is a period unless specified otherwise with the application International property.
;(semicolon)	Sets up different formats for positive and negative values. If the format contains two parts separated by one semicolon (part1;part2), the first part formats positive numbers or zero and the second part formats negative numbers. If the format contains three parts, the first part formats positive numbers, the second formats zero, and the third formats negative numbers.
E- or e-	Displays the number in scientific notation, preceding negative exponents with a minus sign. If the format includes at least one 0 or 9 following this symbol, Quattro Pro displays the number in scientific notation and inserts E or e. If the exponent contains more digits than 9s or 0s following this symbol, the extra digits are displayed.
E+ or e+	Displays numbers in scientific notation, preceding negative and positive exponents with a minus or plus sign, respectively. If the format includes at least one 0 or 9 following this symbol, Quattro Pro displays the number in scientific notation and inserts E or e. If the exponent contains more digits than 9s or 0s following this symbol, the extra digits are displayed.

The next table shows examples of user-defined numeric formats.

Number examples

Cell contents	Format code	Appears as
6435	N$9,999.99	$6,435.
4.3	N9,900.99	04.3
234	N9;Nnil;N(9.00)	234
0	N9;Nnil;N(9.00)	nil
-3	N9;Nnil;N(9.00)	(3.00)
556966830	N999-99-9999	556-96-6830
4155554828	N(999)999-9999	(415)555-4828
37.2	N99.00 ft	37.20 ft
102089	NConfidential	Confidential

.75	N9.99%	75.%
.75	N9.99'%'	.75%
1234567	N0E+999	1E+6

See Also
Using Numeric Format Codes

Date/Time Format Symbols

Symbol	Action
T or t	Begins a Date/Time format (as opposed to a Number format).
d or D	Displays the day of the month as a one- or two-digit number (1-31).
dd or DD	Displays the day of the month as a two-digit number (01-31).
wday, Wday, WDAY	Displays the day of the week as a three-character abbreviation all lowercase, or with the first letter capitalized, or all uppercase.
weekday, Weekday,WEEKDAY	Displays the day of the week all lowercase, or with the first letter capitalized, or all uppercase.
m or M	Displays the month as a one- or two-digit number (1-12). If preceded by an hour code (h, H, hh, or HH), displays the minute as a one- or two-digit number (1-59).
mm or MM	Displays the month as a two-digit number (01-12). If preceded by an hour code (h, H, hh, or HH), displays the minute as a two-digit number (01-59).
Mo	Displays the month as a one- or two-digit number (1-12).
MMo	Displays the month as a two-digit number (01-12).
mon, Mon, MON	Displays the month as a three-character abbreviation all lowercase, or with the first letter capitalized, or all uppercase.
month, Month,MONTH	Displays the name of the month all lowercase, or with the first letter capitalized, or all uppercase.
yy or YY	Displays the last two digits of the year (00-99).
yyyy or YYYY	Displays all four digits of the year (0001-9999).
h or H	Displays the hour as a one- or two-digit number. If the format includes ampm or AMPM, the number will be between 1-12. If ampm or AMPM is not included, 24-hour format is used (0-23).
hh or HH	Displays the hour as a two-digit number. If the format includes ampm or AMPM, the number will be between 01-12. If ampm or AMPM is not included, 24-hour format is used (00-23).
Mi	Displays the minute as a one- or two-digit number (1-59).
MMi	Displays the minute as a two-digit number (01-59).
s or S	Displays the second as a one- or two-digit number (1-59).
ss or SS	Displays the second as a two-digit number (01-59).
AMPM	Displays the time in 12-hour format with characters for morning (AM) or afternoon (PM).

The next table shows examples of user-defined date and time formats.

Date and time examples

Cell contents	Format code	Appears as
2/10/92	TMonth D, yyyy	February 10, 1992
2/10/92	TWeekday	Monday

@TODAY	TMM/DD/YY	02/10/92
@NOW	THH:MM:SS AMPM	02:36:42 PM

See Also
Using Numeric Format Codes

Miscellaneous Format Symbols

Symbol	Action
*	If the formatted entry is shorter than the column width, fills the column by repeating the character to the right of the asterisk.
' '	Literalizes characters enclosed in the single quotation marks and displays them wherever they're placed within the format. After you apply a format, single quotation marks are automatically inserted in the format around characters that aren't format codes. To make the actual text of a format code appear in a cell (instead of making it act as a code), surround it with single quotation marks (for example, 'WEEKDAY').
\	Works like the quotation mark symbols described above, except it literalizes only the character after the backslash.

See Also
Using Numeric Format Codes

Maintaining Block Names

As explained in Using Named Blocks, you can create names for blocks of cells and later refer to those names instead of using cell references. This saves time and promotes accuracy in formula references. The following topics describe the commands available for maintaining lists of block names in notebooks.

Naming Single-Cell Blocks from Labels

Naming Blocks and Cells from Labels

Making a Table of Named Blocks

Deleting Block Names

See Also

Naming Blocks

Using Block Names in Formulas

Naming Single-Cell Blocks from Labels

With Block|Names|Labels, you can name single-cell blocks using the labels adjacent to them. This is useful in data entry forms, where cells usually have identifying labels beside them, and in macros, where subroutines are named.

To assign names to cells using adjacent labels, select the labels you want to use as names and choose Block|Names|Labels. Indicate the position of the cells you want to name in relation to the labels. For example, if the cells you're naming are below the labels, choose Down. Then choose OK.

Quattro Pro uses each label in the specified block (up to 15 characters) as a name for the adjacent cell. It ignores numeric values in the block; you can use only labels as block names.

Subsequent changes to the labels themselves don't affect the block names.

Before you use Block|Names|Labels, check the labels you plan to use. If there are duplicate labels or labels that duplicate existing block names, Quattro Pro overwrites previous assignments. Use a choice list, or make a table of named blocks (described in the next section) to see a list of existing block names.

Note: If there are leading or trailing spaces in a label, they're included in the block name. For example, a name that appears as "INCOME" may have been entered as "INCOME " (with a space at the end).

You can also use the Create Field Names option of Data|Query to assign names to the cells in the first row of a database.

See Also

Naming Blocks and Cells from Labels
Searching for Records

Naming Blocks and Cells from Labels

With Block|Names|Labels, you can name single cell blocks using the labels adjacent to them. This is useful in data entry forms, where cells usually have identifying labels beside them, and in macros, where subroutines are named.

Now you can automatically name blocks with more than one cell. To do this:

1. Choose Block|Names|Auto Generate; the Generate Block Names dialog box appears.
2. Specify the block to name, including labels.
3. Check the location of the cells to name; you can check more than one, depending on whether the labels are in the top row, the leftmost column, the bottom row, or the rightmost column of the block.
4. If you want to name individual cells, check at least two of the location boxes, then check Name Cells at Intersections.
5. Click OK.

Quattro Pro uses each label in the specified block (up to 15 characters) as a name for the adjacent cell. It ignores numeric values in the block; you can use only labels as block names.

Subsequent changes to the labels themselves don't affect the block names.

Before you use Block|Names|Auto Generate, check the labels you plan to use. If there are duplicate labels or labels that duplicate existing block names, Quattro Pro overwrites previous assignments. Use a choice list, or make a table of named blocks (described in the next section) to see a list of existing block names.

Note: If there are leading or trailing spaces in a label, they're included in the block name. For example, a name that appears as "INCOME" may have been entered as "INCOME " (with a space at the end).

You can also use the Create Field Names option of Data|Query to assign names to the cells in the first row of a database.

See Also

Naming Single-Cell Blocks from Labels

Searching for Records

Example: Naming Blocks and Cells from Labels

The label in each row or column is applied to a block containing all selected cells in that row or column. If you checked more than one location box, a set of names is created for each set of labels. For example, if you select block A1..C4 in the next figure and check the first two boxes in the Generate Block Names dialog box, Quattro Pro creates five blocks: Jan., B2..B4; Feb., C2..C4; Housing, B2..C2; Utilities, B3..C3; and Food, B4..C4.

	A	B	C
1		Jan.	Feb.
2	Housing		
3	Utilities		
4	Food		

If you check Name Cells at Intersections, each cell has an additional name created by combining the two sets of labels. Each name can have up to 15 characters. Here is a table of cell names for this example.

Cell	Name
B2	JAN._HOUSING
B3	JAN._UTILITIES
B4	JAN._FOOD
C2	FEB._HOUSING
C3	FEB._UTILITIES
C4	FEB._FOOD

Creating a Block Name Table

In addition to using a choice list to see a list of named blocks (see Using Block Names in Formulas for details), you can create a table that lists the same information as a permanent part of the notebook.

To create a table of named blocks,

1. Select the top left cell of the block where you want the table.

> **Caution:** Make sure there is enough room for a two-column table, with one row for each block name. Quattro Pro overwrites existing data in cells it uses for the table.

2. Choose Block|Names|Make Table, and choose OK.

A two-column table appears. The first column lists block names alphabetically. The second indicates the corresponding block coordinates.

Quattro Pro doesn't update a named block table. If you add, change, or delete block names, you must re-create the table to reflect the changes. Putting a SpeedButton next to this table is a handy way of updating the table quickly.

See Also
Macros

Deleting Block Names

You can delete a block name without affecting the block itself. Formulas referencing the name then change to reference the block's coordinates; their results don't change.

For example, if a formula references a block named TOTAL, such as @SUM(TOTAL), and you delete TOTAL from the list of names, the formula adjusts to reference the cell coordinates: @SUM(B1..B20).

To delete a block name,

1. Choose Block|Names|Delete. You see a list of existing names in the dialog box.
2. Choose a name from the list, or type a name in the Name box, then choose OK.

To delete all block names in the notebook, choose Block|Names|Reset. Then choose OK to confirm that you want to delete all block names.

Although choosing Block|Names|Reset deletes all block names in the notebook from memory, the notebook itself is not affected. Formulas that reference named blocks change so they refer to block coordinates instead of names.

See Also
Using Named Blocks

Reformatting Text Entries in a Block

Block|Reformat lets you adjust word wrapping in a series of label entries as though they were in a paragraph. You can enter text as a long label or a series of labels, then reorganize the text as a paragraph, taking up as many cells or columns as you want.

To reformat text in a block of cells,

1. Enter the text in one or more cells of the same column. You can include up to 1022 characters in a single cell. There can be no blank cell between the cells containing text, and the entries must begin in the same column.
2. Select the cell containing the first cell to reformat, and choose Block|Reformat.
3. For the Block field, specify the columns or rows where you want the reformatted text to appear. If you specify cells in the active row only, the text fills up as many rows as necessary to display the text within the columns containing those cells. If you specify both columns and rows, Quattro Pro reformats the text within the block if there is enough room. Choose OK.

If an entry is longer than the column width, it's truncated or runs into the cell to the right (if empty). If you specify the reformat block by indicating cells in the first row only, be sure there are enough blank rows underneath for the reformatted text. If the destination block contains data, the data is moved down out of the way.

Remember, even though the text may appear reformatted in several columns or rows, the data is stored in the leftmost cells of each row.

After you reformat a label, you can reverse the formatting with Edit|Undo, if enabled. You can also reformat the text to fill the same space as it did before (some line breaks may change).

If you reformat entries in different font sizes, the font size of the reformatted text is determined by the size of the entry in the first row. This may cause row heights to change.

Block|Reformat works only on existing text. If you reformat an empty block, it has no effect on data you enter later.

See Also
Undoing Mistakes

Transposing Columns and Rows

Whether you set up a page with entries listed in rows or columns, you can reverse the data in columns and rows with Block|Transpose.

Block|Transpose copies data to another area and transposes the columns and rows. Although you can transpose to the same location as the existing block, it's not wise because the transposed data will usually take a different shape than the original data and won't overwrite it completely. It's safer to specify a different part of the notebook for the destination of Block|Transpose.

Caution: Don't transpose cells containing formulas. Cell references aren't adjusted properly when you use Block|Transpose.

To copy and transpose data, select the block and choose Block|Transpose. For the To edit field, indicate the top left cell of the block to which you want to copy the transposed data, then choose OK. Don't specify a cell within the source block--you will disarrange the data if you do.

Caution: If there is data in the destination block, Quattro Pro overwrites it with the new data.

See Also
Copying and Moving Data

Converting Formulas to Values

Block|Values copies the values calculated by formulas without copying the formulas. Resulting values are usually numeric, but they can also be string values, depending on your formulas. You can copy these values to another part of the notebook or copy them over the formulas that computed them.

To copy a block of formula values, select the block and choose Block|Values. For the To field, indicate the destination block for the copy. To replace the formulas in the block with their values, specify the same block for the destination block. To copy the values of the formulas to another part of the notebook, specify the top left cell of the block you want to copy them to.

To copy formulas and their values to another part of the notebook, use Block|Copy or Edit|Copy and Paste. To convert a single formula to its result, select the cell containing it, press F2, press F9, then press Enter. The result replaces the formula.

To copy formula values into a separate file, use the Values option with Tools|Extract.

See Also

Copying and Moving Data

Extracting Part of a Notebook

Moving Formulas or Referenced Cells

To maintain the accuracy of your data, Quattro Pro changes formula references in a logical manner when you move cells containing formulas, or cells referenced by formulas. Here are the specifics:


If you move a formula without moving the cells it references, the references remain intact, regardless of whether they're absolute or relative. Cell references adjust only when you copy a formula. (See Copying Formulas for a discussion of absolute and relative addresses of cells)


If you move a formula and the cells it references at the same time, the references do adjust. For example, if A2 contains +A1+1, and you move A2 alone to B1, the formula still reads +A1+1. But if you move A1 and A2 at the same time, the formula reads +B1+1.


If you move a formula to another notebook without moving the cells it references, Quattro Pro links the two notebooks so the formula still references the cells in the original notebook. If you move the formula back to the first notebook, the links are removed.


If you move a cell without moving formulas that refer to it, formulas update to refer to the new location, even if you specified the reference as absolute. For example, if you move the contents of cell B4 to B6 in page A, Quattro Pro revises any formula referencing B4 to reference B6. If you move A:B4 into A:B6 of a notebook named TAX, the formula's reference changes to [TAX]A:B6.


If you move a cell from within a block without moving formulas that refer to the block, formulas still refer to the same block and no longer refer to the moved cell. However, if you move one of the block's coordinate cells (the upper left and lower right cells), the references to the block extend or contract to reflect the new location. For example, if you define the block BILLS as B1..B7, then move cell B7 to B12, BILLS changes to B1..B12.


If you move a coordinate (corner) cell into another notebook, references to the block aren't adjusted.


If you move an entire named or referenced block, Quattro Pro updates the block name or the reference in formulas.

Moving into a Referenced Block

If you move data into a coordinate cell of a referenced block or into a single referenced cell, the formula containing the reference becomes invalid. References in formulas to the single cell or block are replaced with ERR, and cells containing those formulas display ERR. If you named the block, the block coordinates in the block names list show as ERR.

ERR values alert you to cells you've overwritten. This ensures that formulas don't display the wrong values if you accidentally move data into cells they reference.

To reverse a move operation that causes errors, use Edit|Undo immediately after the move. Using Block|Move again to move the data back to where it was won't salvage the formulas.

To move data into a cell referenced by a formula or named block, use Block|Copy, then erase the original data. Block|Copy won't turn cell or block references into ERR values.

See Also

Copying and Moving Data

Analytical Graphing

Analytical graphing lets you create graphs from analytical calculations based on notebook data. While you can perform several types of analysis and graph the results, your data remains unchanged.

Analytical graphing relates to other Quattro Pro analysis tools. Aggregation uses eight @functions. Moving Average relates to Tools|Analysis Tools|Moving Average. Linear Fit performs the same analyses as Tools|Analysis Tools|Regression, and Tools|Advanced Math|Regression. Exponential Fit relates to Tools|Analysis Tools|Exponential Smoothing.

See Also

Data Analysis

Advanced Analysis Tools List

Building Graphs

Customizing Graph Properties

Enhancing Graphs

Guidelines for Analytical Graphing

When you've decided what data to analyze and the relationship to investigate, follow these steps:

1. Choose Graph|New to display the new graph Object Inspector.
2. Highlight the 1st series edit field, and enter the first block that contains information to analyze.

   If you want to analyze additional data blocks, enter them as additional series. Click OK to create the graph.
3. Choose Graph|Type and select the type of graph you want. Line graphs are usually appropriate.
4. When the graph window appears, right-click a graph series and choose the Properties command, or choose Property|Series|series number to display the series properties Object Inspector.
5. Click Analyze and choose an analysis type: Aggregation, Moving Average, Linear Fit, or Exponential Fit. None is the default.
6. Click OK to display the analysis results in the graph window. You can press F11 or choose Graph|View to view the graph full-screen.

To graph the original values for a series, choose Analyze|None when viewing the series properties Object Inspector.

To create a table of analysis data, choose Analyze when viewing the series properties Object Inspector. Then, enter the upper left cell of the table's location in the Table edit field.

See Also

Analytical Graphing

Result Tables

Aggregation

Aggregation lets you combine multiple data points and plot them as a single point that may be the sum, average, standard deviation, minimum, or maximum of the data. Plotting aggregates transforms a graph series and reveals relationships not immediately apparent in the spreadsheet, such as weekly averages for information recorded daily.

To perform an Aggregation analysis, follow the guidelines for analytical graphing and choose Analyze| Aggregation in the series properties Object Inspector. See Aggregation Options and Time Periods for details on dialog box settings.

You can create a table of the results (see Result Tables for details).

See Also

Analytical Graphing

Guidelines for Analytical Graphing

Aggregation Graph Types

Aggregation Graph Types

Line graphs work well with aggregation because they easily show trends. Bar graphs are also impressive when plotting aggregated data over time, although trends are not as easy to see when you graph multiple series in the same graph. X-axis labels are created in the same way as for line graphs.

When using aggregation with bar or line graphs, Quattro Pro can create x-axis labels but doesn't aggregate the x-axis series, even if you enter the analyzed series in the X-Axis series edit field. To aggregate the x-axis, enter the block containing data for the x-axis in the X-Axis series edit field and choose Graph Type|XY. Then, right-click the x-axis, choose the Properties command, and choose Analyze in the x-axis properties Object Inspector. Enter the same Aggregation Period, Series Period, and Function settings as you did for the other series in the graph.

See Also

Analytical Graphing

Aggregation

Graph|Type

Aggregation Options and Time Periods

Aggregation Options and Time Periods

When you choose Analyze|Aggregation in the series property Object Inspector, the dialog box offers these options: Series Period, Aggregation Period, and Function.

The Aggregation commands are based on one day representing a period of 1. For the purposes of aggregation, Quattro Pro uses the following standards:

A week has	A month has	A quarter has	A year has
7 days	30 days	90 days	360 days
4 weeks	12 weeks	51 weeks	
3 months	12 months		
4 quarters			

See Also
Analytical Graphing
Aggregation

+K

K Series Period 

Series Period

Series Period indicates what the series values represent, or what period exists in the series. For example, if the series contains weekly information, choose Weeks.

 Days (the default) represents a period of 1.

 Weeks represents a period of 7.

 Months represents a period of 30.

 Quarters represents a period of 90.

 Years represents a period of 360.

See Also

Analytical Graphing

Aggregation

Aggregation Options and Time Periods

+K

K Aggregation Period 

Aggregation Period

Aggregation Period indicates the number of data values to combine for each point on the graph. Choose one of the following or type in a number to use an arbitrary period:

 Weeks (the default) shows weekly data, or aggregates by 7.

 Months aggregates by 30.

 Quarters aggregates by 90.

 Years aggregates by 360.

If you enter an arbitrary period, the Series Period is automatically set to Days. You can reset Series Period to another option, if you want. For example, enter 3 to group information in 3-day periods with Series Period set to Days. If you specify a standard Aggregation Period, say 7, the setting will show the standard (Weeks) instead of the number (7).

Arbitrary Aggregation Period

Remember that the aggregation settings are based on a day representing a period of 1, but this doesn't mean you can use only daily information in an aggregation graph. Regardless of what the series represents (days, weeks, months, quarters, or years) if the period you want is other than one of the settings on the Aggregation Period menu, always enter the arbitrary period in terms of days.

For example, if the spreadsheet contains daily data and you'd like to see biweekly averages, set Aggregation Period to 14 and press Enter. If you have weekly data and want to see biweekly averages, set Series Period to Weeks, set Aggregation Period to 14 and press Enter.

Note: The aggregation period must be larger than the series period. For example, if you choose Series Period|Months and Aggregation Period|Weeks, an error message tells you the series period interval must be smaller than aggregation period interval.

See Also

Analytical Graphing

Aggregation

Aggregation Options and Time Periods

Function

Function indicates how to transform the data:

SUM (the default) uses @SUM(list) to total the data.

AVG uses @AVG(list) to average the data.

STD uses @STD(list) to calculate the population standard deviation.

STDS uses @STDS(list) to calculate a sample standard deviation.

MIN uses @MIN(list) to specify a series minimum.

MAX uses @MAX(list) to specify a series maximum.

VAR uses @VAR(list) to calculate the population variance.

VARS uses @VARS(list) to calculate a sample variance.

Think of the Aggregation commands as handy standards for simple periodic aggregation. For example, when you choose Series Period|Days, Aggregation Period|Months, and Function|AVG, Quattro Pro will average 30 data points, plot the average, then average the next 30 points, plot the average, and so on.

See Also

Analytical Graphing

Aggregation

Aggregation Options and Time Periods

@Functions

Moving Average

Moving Average analysis smooths fluctuating data points by plotting progressive averages. Starting with the first point in the series, Quattro Pro calculates and plots the average for a specified previous number of points, called a period. At each following point, Quattro Pro maintains the specified period: it drops the oldest value, the one farthest from the new point, so the number of points averaged is always the specified period. Quattro Pro then calculates and plots the new average, and continues in this way.

For example, if daily sales vary widely, you may want to determine a general trend by smoothing data points. With a period of 3,


Quattro Pro starts with the first day's sales and plots this value.


For the second average, Quattro Pro moves to the second day, averages the first and second days' totals, and plots the average.


Quattro Pro moves to the third day and calculates the average of the third, second, and first days' sales, then plots the average.


At the fourth day, Quattro Pro drops the first day's sales and averages the second, third, and fourth days' sales.

With Period set to 3, Quattro Pro averages each day's sales with the previous two. In this way, each point is tempered by previous points and data is smoothed to show a general trend.

To perform a Moving Average analysis, follow the guidelines for analytical graphing and choose Analyze|Moving Average in the series property Object Inspector.You can perform a standard or weighted moving average and can create a table of the results. For details, see Moving Average Options and Result Tables.

Moving Average Graph Types

Line and area graphs are most effective for displaying moving averages. They highlight trends better than bar graphs, although bar graphs are not inaccurate.

See Also

Analytical Graphing

Guidelines for Analytical Graphing

Graph|Type

Moving Average Options

To perform a Moving Average analysis, follow the guidelines for analytical graphing and choose Analyze|Moving Average in the series property Object Inspector. The dialog box offers these options:

Period	Indicates the number of points to average.
Weight	Lets you select a weighted moving average. The default is None. When you select Standard, Quattro Pro numerically places greater emphasis, or weight, on recent points (closest to the point it's working with) and less weight on older points (farthest from the most recent point). The oldest point is multiplied by 1, the next multiplied by 2, and so on; the most recent point is multiplied by the specified Period. For example, with Period set to 3, the oldest point is multiplied by 1, the next one by 2, and the most recent point is multiplied by 3. Then, the weighted moving average is determined by dividing the sum of the weighted points by the sum of the weights (in this case, 1+2+3).

See Also

Analytical Graphing

Moving Average

Linear Fit

Using simple linear regression, Analyze|Linear Fit generates a line that best fits the data. Not only is Linear Fit useful for showing a general trend among fluctuating points, it is easier to grasp regression information in graph form than in a table.

To perform a Linear Fit analysis, follow the guidelines for analytical graphing; choose Line for the graph type.

Linear Fit calculates and plots regression information in a line, even if the data does not have a general trend.

You can create a table of the results and use the table to project hypothetical values.

See Also

Analytical Graphing

Guidelines for Analytical Graphing

Graph|Type

Result Tables

Exponential Fit

Exponential Fit generates a curve to fit data that increases or decreases geometrically--through multiplication rather than addition. For this feature to work, all values in the series must be nonzero and of the same sign, either all negative or all positive.

To fit data with a curve, follow the guidelines for analytical graphing; choose Line for the graph type.

You can create a table of the results.

See Also

Analytical Graphing

Guidelines for Analytical Graphing

Graph|Type

Result Tables

Result Tables

Graphs show trends, but sometimes you need exact data for further analysis. Follow these steps to generate a table of analytical graphing results in the spreadsheet:

1. Right-click the series you want to analyze and choose the Properties command, or choose Property| Series.
2. Choose Analyze and specify the type of analysis (Aggregation, Moving Average, Linear Fit, or Exponential Fit).
3. When the Analyze dialog box appears, specify a block for the table in the Table edit field. You can indicate an entire block, or specify only one cell as the top left cell of the table.

If you specify one cell, Quattro Pro extends the block downward as needed.

Caution! Check that there's room for the table; otherwise, Quattro Pro overwrites unprotected existing data. Try specifying a block below and to the right of data.

If you specify an entire block, depending on how you want to view the table values, you can specify a block one row wide, one column wide, or composed of multiple rows and columns. Quattro Pro fills the block as in Edit|Fill: it starts with the upper left cell, fills the column, then starts at the top of the next column, and so on.

If the block you specified is too small, Quattro Pro displays only what the specified block will hold. To see all the values, choose Table again and specify either one cell as the top of the table (Quattro Pro then extends the block downward as needed), or specify a larger block.

Note: If the type of analysis is Linear Fit or Exponential Fit and you specify a series block larger than the number of existing data points, Quattro Pro projects additional values based on the plotted line or curve and enter them after the others. This applies only to standard line graphs, not XY graphs.

See Also

Analytical Graphing

Guidelines for Analytical Graphing

Multiple Relationships in One Graph

Quattro Pro makes it easy to view more than one relationship in the same graph without redesigning your spreadsheet and without repetitive calculations. Just follow the guidelines for analytical graphing and enter the same block for several series. Then, perform different analyses on each series.

You can view multiple Aggregation relationships in the same graph if you aggregate by the same period for each analysis.

See Also
Analytical Graphing
Guidelines for Analytical Graphing

Data Analysis

These topics describe how to use Quattro Pro's many tools for analyzing notebook data:

See Also

Advanced Analysis Tools List

Analytical Graphing

Using the Scenario Manager

Scenarios are "snapshots" of data models. They show changing data or variable values plugged into a model and the values that result. The following topics give an overview of scenario management in Quattro Pro and explain how to use Tools|Scenario Manager to create and display scenarios:

See Also

Data Analysis

Scenario Management Overview

Scenarios are "snapshots" of data models. They show changing data or variable values plugged into a model and the values that result. For example, a worst-case scenario shows what you can expect from the least desirable set of variable values; a best-case scenario uses the most desirable values as input. The Scenario Manager can also be used for tracking versions of notebooks, with changes as minor as changing the fonts or numeric format of blocks (for details, see Tracking Versions). Several commands, such as Block|Copy and Tools|Scenario Manager, help you set up and save scenarios.

The Model Copy option of Block|Copy lets you copy models to different locations in a notebook without affecting formula references. Then you can change values in each copy to produce different scenarios. The Model Copy method of scenario management is most suitable when you want to build a set of scenarios for further analysis--with Tools|Optimizer or Solve For, for example. You can place each scenario on a separate page labeled with a unique name, perform a variety of analyses, and print the results whenever you like.

For a more powerful and interactive approach to scenario analysis, use the Scenario Manager. It highlights value changes for new data, so it's easy to compare and summarize scenarios. At any time you can name and save scenario variations in report form.

See Also
Using the Scenario Manager

Planning Scenarios

Before using the Scenario Manager, build a data model. Enter formulas in your notebook spreadsheet and add any supporting data. Then,

1. Decide how much of the notebook to include in the scenario group: a block, a page, or the entire notebook. The Scenario Manager tracks all changes within that area, so it's best not to include more than you need. See The Capture Area for details.
2. Decide which information to change with each scenario. These are the changing cells.
3. Determine which formula cells reference the changing cells and contain the results you're interested in. These are the result cells.

   See Scenario Cells for more information on the changing and result cells.
4. If you want, use Block|Names--or press Ctrl+F3--to give each of the changing and result cells a meaningful name. These names become labels in the Scenario Summary Report. If you have column or row labels, you can use those for identification in the Scenario Summary Report without actually naming scenario cells.

   See Scenario Summary Report for report details.

See Also

Scenario Manager Guidelines

Using the Scenario Manager

Scenario Manager Guidelines

Follow these steps to set Scenario Manager options; create, display, and delete scenarios; and produce summary reports:

1. Choose Tools|Scenario Manager to display the Scenario Manager SpeedBar.
2. Set a capture area. Page (the current page) is the default, but you can choose Notebook or Block; adjust other options as needed.
3. Name and save the scenario baseline.
4. Name, save, and edit scenarios.
5. Rename the scenario group (optional); Group1 is the default.
6. View the Scenario Summary Report.

As a shortcut, you can build a model, then use the Scenario Expert to name and save scenarios. Click the Expert button in the Scenario Manager SpeedBar and follow the directions.

See Also
Using the Scenario Manager

The Scenario Manager SpeedBar

Use the Scenario Manager SpeedBar to identify changing cells and result cells, set scenario options, save and retrieve scenarios, and view the Scenario Summary Report. Choose Tools|Scenario Manager to display the SpeedBar.

For a description of each SpeedBar object, point to it. Its name appears at the bottom of the notebook window. To display Object Help with more information, press Ctrl and right-click the mouse while pointing to the object.

To close the Scenario SpeedBar, click the X at its right end.

See Also
Using the Scenario Manager

The Capture Area

The capture area is the block, page, or whole notebook where the Scenario Manager tracks changes. Once you set the scenario baseline, the Scenario Manager tracks all changes in the capture area and interprets them as changing cells and result cells.

The default capture area, Page, is usually suitable. To verify or change the capture area, click the Scenario Manager SpeedBar button labeled Group. The Scenario Group Control dialog box appears, with these Capture Area options:

Notebook	Tracks changes within the entire notebook.
Page	Tracks changes in the active page.
Block	Tracks changes in the specified block (2-D or 3-D).

Caution! If you edit data within the capture area then display an existing scenario, any changes not yet saved to a scenario are lost--unless you immediately use Edit|Undo (or Ctrl+Z) to restore them. Choose a capture area carefully to avoid possible data loss or extra changing and result cells; the capture area applies to all scenarios in the current group.

If you choose Notebook, scenario changes are tracked and displayed on all pages except the Scenarios page and Graphs page. Use this setting carefully to avoid unexpected results. For example, suppose you change page B, save a scenario, change page C, then display the scenario you last saved. All data on page B appears but the changes on page C disappear.

Only set Capture Area to Notebook if you're interested in tracking changing and result cells across several pages and don't intend to use the notebook for anything else but storing related scenarios.

Choose Page if you'll be changing data on a single page but don't want to specify where. You can choose Page for most purposes.

For maximum control, choose Block then follow the instructions in Scenario Cells to identify scenario changing and result cells. The specified block must be contiguous, either 2-D or 3-D.

See Also
Using the Scenario Manager

The Scenario Baseline

The Scenario Manager tracks value changes against a baseline. If you're working with an empty notebook, enter any labels and formulas before setting the baseline. If you'll be changing data that's already in the notebook, set the baseline before starting the changes.

To set the baseline,

1. Click the Capture Scenario button. Think of it as a camera that's taking a snapshot of current data in the capture area.
2. When the Capture Scenario dialog box displays, name the baseline scenario.

   The default baseline name is Base Scenario. To use another, type a new name in the Scenario Name edit field. Then click OK.

Now, Quattro Pro tracks all changes within the capture area until you save another scenario.

See Also
Using the Scenario Manager

Scenario Cells

The Scenario Manager tracks two types of scenario cells: changing cells (which generally contain new data entries) and result cells (formula cells that reference the changing cells).

Once you set the baseline, you can identify scenario cells automatically or manually. For details, see

Finding Scenario Cells Automatically

Manually Identifying Scenario Cells

Changing cells are shaded yellow on color monitors; result cells are green. You can use the Toggle Highlights button to switch cell highlighting on and off.

When you save a scenario and generate a Scenario Summary Report, the report includes values in all the shaded cells. You may want to focus on fewer changing and result cells than Quattro Pro has shaded. To exclude extra cells of either type, select them and click the Remove Scenario Cells button.

See Also
Using the Scenario Manager

Finding Scenario Cells Automatically

To find scenario cells automatically, enter new values into the changing cells. Click the Find Scenario Cells button. The Scenario Manager identifies all cells in the capture area that have had direct entries, property changes, or formula edits since the baseline was set. These are changing cells. They can contain values or formulas. Result cells are all cells containing formulas that have had value changes but no formula edits since the baseline was set.

Changing cells are shaded yellow on color monitors; result cells are green. You can use the Toggle Highlights button to switch cell highlighting on and off.

When you save a scenario and generate a Scenario Summary Report, the report includes values in all the shaded cells. You may want to focus on fewer changing and result cells than Quattro Pro has shaded. To exclude extra cells of either type, select them and click the Remove Scenario Cells button.

See Also

Scenario Cells

Using the Scenario Manager

Manually Identifying Scenario Cells

To identify scenario cells manually, select one or more changing and result cells, then click the Add Scenario Cells button. All selected cells with values are identified as changing cells; cells with formulas are result cells.

Changing cells are shaded yellow on color monitors; result cells are green. You can use the Toggle Highlights button to switch cell highlighting on and off.

When you save a scenario and generate a Scenario Summary Report, the report includes values in all the shaded cells. You may want to focus on fewer changing and result cells than Quattro Pro has shaded. To exclude extra cells of either type, select them and click the Remove Scenario Cells button.

See Also

Scenario Cells

Using the Scenario Manager

Naming and Saving Scenarios

When scenario cells and values are ready to save, click the Capture Scenario button. Enter the new scenario name in the Scenario Name edit field (or use the default name) and click OK. If you haven't identified scenario cells manually, the Scenario Manager performs an automatic identification (described in the previous section) and saves the information under the name you entered.

Continue until you have added as many scenarios as you want. They are saved permanently when you save the notebook.

See Also
Using the Scenario Manager

Scenario Groups

A set of scenarios is called a group. Each notebook can have as many scenario groups as you need, each with its own baseline. To create or rename a group, click the button labeled Group. The Scenario Group Control dialog box appears. It displays the name of the active group (Group1 is the default name for the first group) and offers these buttons:

New	Adds a new group. Click it, then type the name of the new group in the edit field and click OK. When you create a new group, the first scenario you capture will be the baseline.
Rename	Gives the active group a new name. Click Rename, type the new name in the To edit field, and click OK.
Delete	Erases the active group and all scenarios in it.
Options	Lets you change block tracking (see Tracking Versions for details).

For information on the other settings in this dialog box, see The Capture Area.

See Also
Using the Scenario Manager

Displaying and Editing Scenarios

To display a scenario, click its name in the Scenario Name list. If you have several scenario groups, select the target group in the Group Name list before selecting the scenario.

When you display a scenario, its changing cell and result cell values replace those currently onscreen. If you want to add or delete changing or result cells, see Scenario Cells. You can change the contents of any cell with standard editing procedures.

To save edits to the active scenario, click the Capture Scenario button. When the Capture Scenario dialog box appears, type the name of the active scenario to replace it with current cells and values. You can type another name to replace another existing scenario or create a new one. Click OK to accept the scenario name. Unless you've manually identified scenario cells, any edits appear in the Scenarios page when you generate a report.

See Also
Using the Scenario Manager

Scenario Summary Report

Each time you add a scenario, its changing and result cell values are stored. You can view and print them whenever you want.

Use the Report button to display the Scenario Summary Report. The Scenario Report dialog box offers several options. When checked, they have these effects:

Report All Groups	Displays a report for each scenario group, not just the active group.
Use Left Labels	Identifies scenario cells with text in cells immediately to their left.
Use Top Labels	Identifies scenario cells with text above them.
Find Empty Page	Puts the Scenario Summary Report on the first empty page in the notebook. To place it somewhere else, uncheck Find Empty Page and enter a page and block in the edit field.

Since these are check boxes, you can check all, any, or none. Click OK to display the Scenario Summary Report.

If you've checked a labels option or named each changing and result cell, a label or name displays next to each value in the Scenario Summary Report.

See Also
Using the Scenario Manager

Deleting Scenarios

To remove a scenario from the Scenario Name list and Scenarios page, select it in the Scenario Name list and click the Delete Scenario button. If it's in another scenario group, click the Scenario Group button and select the group name first. If you're viewing the Scenario Summary Report when you delete a scenario, it still displays until you click the Report button to generate a new report.

See Also
Using the Scenario Manager

Deleting Scenario Groups

To delete a group and all scenarios in it,

1. Select the group in the Group Name list.
   Click the Scenario Group button.
2. Click Delete.

See Also

Using the Scenario Manager

Tracking Versions

Sometimes you need to keep track of different versions of a block, page, or whole notebook--if you're building a complicated budget notebook or macro application, for example.

To use the Scenario Manager for version tracking,

1. Display the Scenario Manager SpeedBar.
2. Click the Group button, then

Set the capture area to suit your tracking needs; keep the area as small as possible.

Click Options to change the Update on Block Operations setting. For complete version tracking, you'll probably want to uncheck it.

Update on Block Operations indicates whether to update all scenarios in the active group when blocks are moved, inserted, or deleted. When that setting is checked, the Scenario Manager tracks all block operations internally; no scenario changes are recorded. If unchecked, the scenarios register any moves, insertions, and deletions.

3. Click the Capture Scenario button to set the baseline.
4. Make changes in the capture area until you're ready to save a version, then click the Capture Scenario button again and enter a new version name.
5. Continue editing and saving versions until your work session is over. Be sure to save a final version before closing the Scenario Manager or Quattro Pro.

To view previous versions, display the Scenario Manager SpeedBar, choose the target group in the Group Name list if its not active already, then choose the target version in the Scenario Name list.

Caution! If you've been editing anywhere in the capture area, be sure to save the current version in the Scenario Manager before displaying a previous version. Otherwise, edits may be lost. Remember, if Undo is enabled, you can use Edit|Undo or Ctrl+Z to restore an unsaved scenario as long as no edits have been made since displaying the current scenario.

See Also

Using the Scenario Manager

Using the Consolidator

The Consolidator is similar to Tools|Combine; it combines data from multiple blocks into one using your choice of @function operators. The following topics give an overview of the Consolidator and explain how to create and use consolidations:

Consolidator Overview

Consolidator Guidelines

The Consolidator SpeedBar

Source Blocks

Operators

Destination Blocks

Consolidator Results

Consolidator Options

Displaying Results as Formulas

Consolidating with Labels

Naming and Saving Consolidations

Deleting Consolidations

See Also
Data Analysis

Consolidator Overview

The Consolidator is similar to Tools|Combine; it combines data from multiple blocks into one using your choice of operators. Unlike Tools|Combine, the Consolidator offers a choice of @functions as operators and lets you save consolidation setups. You can specify source and destination blocks plus the @function operator used to combine them, then name that setup and save it for future use.

While the Consolidator is often used for permanent calculations, its flexibility makes it useful for model analysis. For example, you can define several source blocks, then use any or all of the operators with different destination blocks. If you save each setup, it's easy to change a few values and recompute with a single click.

If you are working with a database, you can choose blocks with field names at the top of the block or to the left and automatically sort combined data in the destination block.

See Also

Using the Consolidator

Consolidator Guidelines

Consolidator Guidelines

Follow these steps to set up and perform consolidations:

1. Define two or more source blocks and add them to the Source Block list.
2. Choose an operator.
3. Define a 2-D destination block and enter it in the Destination Block field.
4. Verify and change option settings.
5. Name the consolidation and add it to the Consolidation Name list.
6. Perform the consolidation.

The following topics contain details on each step.

Source Blocks

Operators

Destination Blocks

Consolidator Results

Consolidator Options

Displaying Results as Formulas

Consolidating with Labels

Naming and Saving Consolidations

Deleting Consolidations

As a shortcut, you can use the Consolidator Expert to create and save consolidations. Click the Expert button in the Consolidator SpeedBar and follow the directions.

See Also

The Consolidator SpeedBar

Consolidator Overview

Using the Consolidator

The Consolidator SpeedBar

Choose Tools|Consolidator to display the Consolidator SpeedBar. You use it to define source and destination blocks, specify operators, set options, and save or delete consolidation setups.

To identify each SpeedBar object, point to it and read its name at the bottom of the window. For a description, display Object Help; press Ctrl and right-click the mouse while pointing to an object.

To close the SpeedBar, click the X at its right end.

See Also

Consolidator Overview

Consolidator Guidelines

Source Blocks

Consolidator source blocks are the blocks to combine. For example, you might want to count or average all cells in the source blocks. Since the chosen operator works on equivalent cells in each source block, source blocks usually have the same number of rows and columns.

To add two or more blocks to the Source Block list, select each and click the Add Source Block button. Remember, to enter discontiguous blocks press Ctrl while selecting cells. Each sub-block of discontiguous blocks is treated as a separate block. If you enter a 3-D source block, the block on each page is treated separately and combined with those on the other pages.

To remove a block from the Source Block list, select it in the list and click the Remove Source Block button.

See Also

The Consolidator SpeedBar
Consolidator Overview
Consolidator Guidelines

Operators

The Operator list offers these operators: SUM, AVG, COUNT, MIN, MAX, STD, STDS, VAR, and VARS. They work like statistical @functions of the same name on equivalent cells of each source block.

For example, if you choose SUM to perform the consolidation in the following example, the results display in F2..F4.

	A	B	C	D	E
1					
2	1	4	6		11
3	2	5	7		14
4	3		8		11
5					

Cell E2 is the sum of cells A2, B2, and C2; cell E3 is the sum of A3, B3, and C3; and so on. With columns of different lengths, cell E4 is the sum of A4 and C4 only.

See Also

Consolidator Overview

Consolidator Guidelines

Destination Blocks

You can specify a 2-D destination block to hold consolidation results. Select the block and click the Add Destination Block button. To edit your entry, select another block and click the Add Destination Block button again.

See Also

The Consolidator SpeedBar

Consolidator Overview

Consolidator Guidelines

Consolidator Results

When you've specified blocks and an operator, click the Consolidate button to perform the consolidation. By default, consolidation results display as formulas. Consolidator Options describes how to display them as values.

The dimension of the destination block affects the dimension of the consolidation result. If the source blocks and destination block have the same number of columns and rows, the Consolidator displays all results in their corresponding position in the destination block. For example, the upper left cell of one source block is combined with the upper left cell of a second source block; the results appear in the upper left cell of the destination block. This is true even when those upper left cells are in different rows.

When the source and destination blocks have different dimensions, the Consolidator combines equivalent cells when possible. Cells without equivalents are handled differently depending on the operator. If the destination block is smaller than a source block, results are limited to the size of the destination block.

See Also

The Consolidator SpeedBar

Consolidator Overview

Consolidator Guidelines

Consolidator Options

Option settings determine whether consolidation results appear as formulas or values and whether data field labels are used in the consolidation. Click the Options button to display the Consolidator Options dialog box. These topics explain its options:

Displaying Results as Formulas

Consolidating with Labels

See Also

The Consolidator SpeedBar

Consolidator Overview

Consolidator Guidelines

Displaying Results as Formulas

You can display results as values or formulas. Check Output with Formulas, the default, to enter results as formulas in the destination block. Otherwise, they display as values.

See Also

Consolidator Options

The Consolidator SpeedBar

Consolidator Guidelines

Consolidating with Labels

Suppose you need to consolidate several blocks of equivalent data entered in different row and column orders. As long as matching data has identical row and/or column labels, the Consolidator can combine it and sort the results in any order you specify.

Check Top Row to use labels in the top row of Consolidator source and destination blocks; check Left Column to use labels in the left column. You can check either or both options.

The Consolidator matches column labels, then matches row labels within columns to find equivalent cells for consolidation. Then it locates cells in the destination block with the same column and row labels and puts the results there. If the destination block doesn't contain labels, data is entered according to label order in the first source block. The destination block is formatted to match the first source block.

See Also

Consolidator Options
The Consolidator SpeedBar
Consolidator Guidelines

Naming and Saving Consolidations

You can save consolidation setups for future use:

1. Click the Save Consolidation button.
2. When the Consolidation Name dialog box displays, type a name in it.
3. Click OK to enter the name in the Consolidation Name list.

To reuse a saved consolidation, click its name in the Consolidation Name list then perform the consolidation. Current contents of the source blocks are combined; new results appear in the destination block or blocks.

If you want to save a "snapshot" of consolidation source and destination block contents, use the Consolidator within the Scenario Manager; display the Scenario Manager SpeedBar and set a scenario baseline before performing the consolidation (see Using the Scenario Manager for more information).

See Also

The Consolidator SpeedBar

Consolidator Overview

Consolidator Guidelines

Deleting Consolidations

To remove a consolidation setup from the Consolidation Name list, click the name in the list then click the Delete Consolidation button. Notebook data isn't changed, but that consolidation setup is no longer available.

See Also

The Consolidator SpeedBar

Consolidator Overview

Consolidator Guidelines

Creating Frequency Distributions

Frequency distributions show how many values in a block fall within given value intervals. To create a frequency distribution table:

1. Set up a bin of ranges to analyze in a single column with a column of blank cells to the right. The ranges must be in ascending order.
2. Choose Data|Frequency.
3. Specify the value block(s), the block of values to analyze; separate multiple blocks with commas (,).
4. Specify the bin block, the block of values you set up in step 1.
5. Choose OK or press Enter. Quattro Pro calculates the frequency distribution of values within each bin range and enter the results to the right of the bin block.

See Also

Setting Up a Frequency Bin Block

Data|Frequency

Data Analysis

Setting Up a Frequency Bin Block

A bin block contains the range of intervals you want analyzed. The block must be a single column with a column of blank cells to its right (where the results will be written).

You can use Block|Fill to create the bin. Numbers must appear in ascending order, from top to bottom of the column.

Each value in the bin block represents all values from it down to the previous value. (The first value represents any value less than or equal to itself.)

The result block will be one cell longer than the bin block. Its last cell contains the number of values found that were greater than the final number in the bin.

See Also

Creating Frequency Distributions

Data|Frequency

Other Basic Analyses

You can supplement the statistical information in your notebook with appropriate @functions. For example, you could include these formulas at the end of the weekly sales figures, contained in a block named SALES:

`@SUM(SALES)`	calculates total sales.
`@COUNT(SALES)`	calculates the number of sales in the column.
`@AVG(SALES)`	calculates the average sales amount.

See Also

Creating Frequency Distributions

Statistical @Functions

Graphing Frequency Distributions

To create an XY graph of a frequency distribution, specify the bin block as the x-axis series and the results as the first series of values.

See Also

Creating Frequency Distributions

XY and High-Low Graphs

Performing Regression Analysis

You can create a regression analysis table showing how sets of independent variables can affect a certain set of dependent variables.

1. Choose the variable data you want to analyze. You can have one dependent variable that you believe is influenced by independent variables; for example, quarterly revenues. You can have up to 150 independent variables that seem to have a positive or negative effect on the dependent variable; for example, advertising expenditures and number of sales calls.
2. Make sure the data blocks you want to use in the regression analysis have the same number of rows.
3. Choose Tools|Advanced Math|Regression. Quattro Pro displays the Regression dialog box.
4. Specify the column of dependent data.
5. Specify the column(s) of independent data; they may be in discontiguous columns.
6. Specify the upper left cell of the output block where you want Quattro Pro to write the regression information.
7. If you want to force the y-intercept value to zero, set Y Intercept to Zero.
8. Choose OK or press Enter.
9. Regression data displays in the regression output block.

Regression tables aren't automatically updated. If you alter the values in the independent and dependent blocks, use Tools|Advanced Math|Regression again to see the new results.

See Also

Regression Dialog Box

Advanced Math Commands

Data Analysis

Introducing Matrix Operations

A matrix is a rectangular array of numbers. Each column in the matrix corresponds to one variable and each row corresponds to a linear constraint.

Matrix operations can help you solve sets of linear formulas and equations. For example, suppose you have these four unique equations describing variables w, x, y, and z:

```
1w+1x+2y+3z=10

3w+2x+2y+1z=20

1w+0x+3y+4z=15

1w+1x+0y+1z=6
```

You can express the coefficients of these equations (the numbers multiplying the variables w, x, y, and z) in this "4 by 4" coefficient matrix:

```
1 1 2 3

3 2 2 1

1 0 3 4

1 1 0 1
```

And you can express the results in this "4 by 1" constant matrix (the first variable is at the top of the list, the second is next, and so on):

```
10

20

15

 6
```

You can use matrix multiplication and inversion to find the value of each variable in the second matrix that satisfies the constraints in the first matrix.

See Also

Multiplying Two Matrices

Inverting a Matrix

Solving Matrix Problems

Data Analysis

Multiplying Two Matrices

Matrix multiplication multiplies the values in two matrices and displays the results in a third. The number of columns in the first matrix must equal the number of rows in the second matrix. To multiply matrices,

1. Choose Tools|Advanced Math|Multiply.
2. Specify Matrix 1, the first matrix you want to multiply.
3. Specify Matrix 2, the second matrix.
4. Specify the Destination, the top left cell of the area where you want to write the resulting matrix.
5. Choose OK or press Enter. Quattro Pro multiplies the two matrices and displays the results in the Destination block.

See Also

Matrix Multiplication Dialog Box

Inverting a Matrix

Solving Matrix Problems

Inverting a Matrix

When you invert a matrix, you create another matrix with this feature: When you multiply a matrix by its inverse, the resultant matrix is an identity matrix (all 1s and 0s, with only a single diagonal of 1s). To invert a matrix,

1. Choose Tools|Advanced Math|Invert. The Invert dialog box displays.
2. Specify the Source, the matrix block you want to invert.
3. Specify the Destination, the upper left cell of the block where you want to write the inverted matrix. If you specify the same block as the invert block, the inverted matrix overwrites the existing matrix.
4. Choose OK or press Enter. Quattro Pro inverts the matrix and writes the results into the Destination block, overwriting any data in the block.

See Also

Matrix Inversion Dialog Box

Multiplying Two Matrices

Solving Matrix Problems

Solving Matrix Problems

You can use matrix arithmetic to solve sets of "square" linear equations, those with the same number of rows and columns in the coefficient matrix. For example, consider this set of linear equations:

```
2x+3y=31
 x+2y=19
```

The coefficient and constant matrices are

Coefficent	Constant
2 3	31
1 2	19

Follow these steps to solve them with matrix inversion and multiplication:

1. Invert the coefficient matrix to get the invert matrix.
2. Multiply the invert matrix by the constant matrix.
3. The solution matrix contains the results (x=5 and y=7):

```
5
7
```

If you multiply the coefficient matrix by the solution matrix, the result is the constant matrix:

```
2  3        5          31
     times        equals
1  2        7          19
```

See Also

Matrix Multiplication

Matrix Inversion

Introducing Matrix Operations

Introducing "What-if" Tables

"What-if" tables show the results of substituting a number of values for one or two variables in a formula.

Also called sensitivity tables, what-if tables can answer questions like these: "What if my company's expenses increase 10%, 20%, 30%? And, what if production increases 10% to 25% at the same time?" What-if tables can also be used to look up values, such as postal charges for different parcel weights.

You can create basic tables that use data you supply, or tables can refer to data already entered in the notebook.

These topics describe how to build what-if tables:

Creating One-Variable "What-if" Tables

Creating Two-Variable "What-if" Tables

See Also
Data Analysis

Creating One-Variable "What-if" Tables

One-variable what-if tables are created by varying a single value that is referenced by one or more "what-if" formulas. You can build a basic table that uses only the data you supply, or you can build a table that refers to data already entered in the spreadsheet. To create a one-variable what-if table,

1. Set up a column of values to substitute in one or more formulas.
2. Set up one or more formulas that refer to a blank cell (the input cell). Enter these in the row above the first substitution value, starting in the first column to the right of the substitution column.
3. Choose Data|What-If.
4. Check One Free Variable to indicate the type of table to build.
5. Specify the Data Table block containing both formulas and substitution values.
6. Specify Input Cell, the blank cell referenced in the formulas.
7. Choose Generate. Quattro Pro calculates each formula using the values in the substitution column and fills in the column below each formula with calculated values.

See Also

Data|What-If Dialog Box

Introducing "What-if" Tables

Creating Two-Variable "What-if" Tables

Creating Two-Variable "What-if" Tables

Two-variable what-if tables show the results of varying two values in a "what-if" formula. You can build a basic table that uses only the data you supply, or you can build a table that refers to data already entered in the spreadsheet. To create a two-variable "what-if" table,

1. Create a column of substitution values.
2. In the row above and to the right of the substitution column, enter a second set of substitution values.
3. In the top left cell of the table, enter the formula to be used in calculating table values. It should refer to two blank cells.
4. Choose Data|What-If.
5. Check Two Free Variables to indicate the type of table to build.
6. Specify the Data Table block, including the formula and both substitution ranges.
7. Specify Column Input Cell and Row Input Cell, the two blank cells referenced in the what-if formula.
8. Choose Generate to create a table of what-if values to the right of the substitution column, below the substitution row.

See Also

Data|What-If Dialog Box

Introducing "What-if" Tables

Creating One-Variable "What-if" Tables

Goal Seeking with Solve For

Tools|Solve For lets you calculate a formula backward, starting with the result and solving for a variable that produces that result. You just specify the result you want, then indicate which variable Quattro Pro can adjust to reach that result. To find a solution,

1. Enter the formula to be solved. It should reference one variable cell.
2. Choose Tools|Solve For.
3. Choose Formula Cell, and specify the cell containing the formula.
4. Choose Target Value, and enter the result you want from the formula, or reference a cell containing the target.
5. Choose Variable Cell, and enter the cell Quattro Pro can vary to achieve the result you want.
6. Change the calculation options, if necessary: Max Iterations determines how many passes Solve For can make to solve the formula, and Accuracy specifies how close Solve For must get to the Target Value (the default is 0.0005).
7. Choose OK to solve the formula. The solution displays in the variable cell.

See Also

Tools|Solve For Dialog Box

Creating Formulas

Data Analysis

Finding Solutions with Optimizer

The Optimizer helps find optimum solutions for linear and nonlinear problems. You can

- evaluate formulas with one or more variables
- solve sets of linear and nonlinear equations and inequalities
- find a minimum or maximum solution, or meet an exact goal
- find values that satisfy limits, or constraints

Once you define a problem, you can save Optimizer settings for future use. You can print reports of interim and final problem solutions.

To solve a problem with Optimizer,

1. Define the problem. You can
   - maximize or minimize a single formula, or make it meet a target value
   - solve a set of equations or inequalities
2. Assign a cell to each variable in the problem.
3. If you're solving a single formula, enter it into a cell. The formula should reference any variable cells you've assigned.
4. If you're solving a set of expressions, enter their left terms into cells; you may enter the right term into adjacent cells, or enter the value into the Optimizer directly when adding constraints.
5. Choose Tools|Optimizer.
6. If you have a goal-seeking problem with one formula, enter its location in the Solution Cell input box. Choose Max to maximize the formula, Min to minimize it, or None if you're not specifying a solution cell. If you want the solution cell formula to reach an exact value, check Target Value and enter the value in the Target Value edit field.
7. Enter the variable cell or cells in the Variable Cell(s) input box. You can enter them as blocks (A1..A4) or lists (A1, B1, C1).
8. If you're solving a set of expressions, add them as constraints. You can also enter constraints when goal seeking to set upper and lower limits for the Solution Cell formula. For instructions, see Entering Constraints.
9. Choose Solve to solve the problem; the solution displays in the variable cells.

For best results, start with reasonable values in the variable cells.

See Also

Tools|Optimizer Dialog Box

Setting Optimizer Options

Loading and Saving Optimizer Models

Producing Optimizer Reports

Data Analysis

Entering and Editing Constraints

Constraints are expressions that limit the values of Optimizer solution and variable cells. They require these cells to fall within upper or lower bounds, such as a budgeted figure you can't exceed or a minimum you must manufacture. Constraints can also specify a target, such as a production order you must fill. You can assign up to 100 constraints.

To enter a constraint, choose Add on the Optimizer dialog box. The Add Constraint dialog box appears, with space to enter each part of the constraint:

Cell	The left side of the expression; a cell containing a formula that must be bounded, or that must equal a target value. You can select a single cell, or select a block of cells that must all have the same bound.
Operator	The operator that connects both terms of the constraint and describes their relationship: less than or equal to (<=), equal to (=), greater than or equal to (>=), or Integer (Cell must be a whole number).
Constant (constraint value)	The right side of the expression; a cell that contains the target or bound, or a value you type in the space. If you specify a cell, it must contain a value, not a formula. You can specify a block of cells equal to or larger than the number of cells included on the left side of the constraint. Each cell on the left is compared with the corresponding cell on the right. A single cell on the left is compared with the upper left cell of a block on the right.

To edit a constraint, select it and choose Change. To delete a constraint, select it and choose Delete.

See Also

Finding Solutions with Optimizer

Tools|Optimizer Dialog Box

Loading and Saving Optimizer Models

The last Optimizer problem you defined is always saved with the notebook. To save more than one problem, or model,

1. Choose Tools|Optimizer|Options|Save Model.
2. Identify a block to hold the model. The block must be three columns wide with enough rows to hold the model. You can specify the cell in the upper left corner of the block.
3. For convenience, use Edit|Names|Create to name the block. Use a meaningful name to make it easier to load when the time comes.

To re-use the model,

1. Choose Tools|Optimizer|Options|Load Model.
2. Specify the block containing the model you want to use.
3. Choose OK.

See Also

Finding Solutions with Optimizer

Tools|Optimizer Dialog Box

Producing Optimizer Reports

Producing Optimizer Reports

To produce an Answer or Detail Report, or both,

1. Define the Optimizer problem.
2. Choose Tools|Optimizer|Options|Reporting.
3. Specify a blank block, or just the upper left cell of a blank block where Optimizer can write either or both reports.
4. Choose OK to return to the Options dialog box. Choose OK to return to the Optimizer dialog box. Choose Solve to generate the report(s).

See Also

Tools|Optimizer|Options|Reporting

Tools|Optimizer Dialog Box

Finding Solutions with Optimizer

Advanced Analysis Tools List

Tools|Analysis Tools lets you perform a number of advanced statistical, numerical, and financial analysis tasks. For a list of Analysis Tools tasks, see Using the Advanced Analysis Tools.

Available tools are listed below in the order they appear on the Analysis Tools SpeedBar, left to right.

Amortization Schedule

Mortgage Refinancing

Exponential Smoothing

Fourier

Histogram

Moving Average

Random Number

Anova: One-Way

Anova: Two-Way with Replication

Anova: Two-Way Without Replication

Correlation

Covariance

Descriptive Statistics

F-Test

Rank and Percentile

Advanced Regression

Sampling

t-Test

z-Test

Each of the analysis tools has an equivalent macro command.

See Also
Data Analysis
Analytical Graphing

Using the Advanced Analysis Tools

In addition to other Tools menu options and @functions, Quattro Pro offers a number of advanced analysis tools. You can use them to analyze statistical, engineering, and financial data. See Advanced Analysis Tools List for a list of available tools.

To use the advanced analysis tools, choose Tools|Analysis Tools to display the Analysis Tools SpeedBar. These topics describe the analyses and how to perform them:

Analysis Tools Guidelines

Arranging Your Data

Describing Data

Testing the Relationship Of Samples

Correlation

Covariance

Smoothing Data

Creating Histograms

Generating Random Numbers

Ranking Data

Performing Linear Regression

Sampling Data

Testing the Means Of Small Samples

Testing the Means Of Large Samples

Analyzing Variance

Comparing Variances Of Two Samples

Generating an Amortization Schedule

Refinancing a Mortgage

Performing Fourier Analysis

Analysis Tools Guidelines

Follow these steps to use analysis tools in Quattro Pro:

1. Choose Tools|Analysis Tools to display the Analysis Tools SpeedBar.
2. Click an analysis tool.
3. In the dialog box for the analysis tool, specify the input block, output block, and other options. You can also click the Help button for information about the dialog box.
4. Choose OK to generate the output block beginning at the specified cell.

As a shortcut, you can use the Analysis Expert to perform an analysis. Click the Expert button in the Analysis SpeedBar and follow the directions.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Arranging Your Data

Before you use an analysis tool, make sure the input block you're analyzing is arranged properly and contains the right kind of data (that is, numeric data, not strings). The analysis tools have varying restrictions on the size and contents of the input block; read the description of the input block option for each tool in the next sections.

For many analysis tools, the input block can contain column or row labels. If the input block you specify contains labels, check the Labels options in the tool dialog box.

Caution: Many of the analysis tools create large output blocks. Make sure you specify the output block in a place where the tool won't overwrite existing data.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Describing Data

Use the Descriptive Statistics tool to characterize a sample. It computes the mean, variance, median, mode, standard deviation, kurtosis, skewness, range, minimum, and maximum of a distribution.

Input Block	is the block of data you want to analyze.
Output Block(s)	identifies the upper left cell where Quattro Pro places the resulting output table(s).
Nth Largest	displays a separate row in the output table for the nth largest value. Enter 1 to display the largest value.
Nth Smallest	displays a separate row in the output table for the nth smallest value. Enter 1 to display the smallest value.
Confidence	displays the confidence level of the means for the input block; the default is 95%.
Group By	specifies whether data in the input block is grouped by Row or Column.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.
Summary	displays the following statistics for each data set in the input block:

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Testing the Relationship Of Samples

You can use the Correlation and Covariance tools to analyze the relationship between two sets of data by producing matrices of correlation and covariance.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Correlation

The Correlation tool computes the correlation coefficient of the numeric values in two or more sets of data. Correlation coefficients range in value from -1.0 to 1.0. A correlation coefficient of zero means that two variables are not correlated. A variable correlated with itself returns a correlation coefficient of 1.0.

Input Block	is the block of data to analyze. The block must contain two or more sets of data arranged in columns or rows.
Output Block	identifies the upper left cell where Quattro Pro starts the resulting output table; the table will extend to the right and downward as needed.
Group By	specifies whether data in the input block is grouped by Row or Column.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Covariance

The Covariance tool returns the covariance matrix of two or more data sets. A positive covariance indicates that as values increase in one data set they increase in another (the sets move together). A negative covariance indicates that as values increase in one data set they decrease in another (the sets move apart). Unrelated data sets have a covariance approaching zero.

Input Block	is the block of data to analyze. The block must contain two or more sets of data arranged in columns or rows.
Output Block	identifies the upper left cell where Quattro Pro starts the resulting output table; the table will extend to the right and downward as needed. The diagonal of the output table shows the variance of each range.
Group By	specifies whether data in the input block is grouped by Row or Column.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

Note: Covariance, unlike correlation, is scale dependent; related data sets yield different covariances if their scales differ.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Smoothing Data

You can use two analysis tools to smooth out variations in a series of values: Moving Average and Exponential Smoothing.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Moving Average

The Moving Average tool generates forecast values by finding the average over a specified number of preceding intervals for each value in a data set. Unlike an average of all the values in the set, the moving average lets you focus on rising and falling trends in the data.

Input Block	is the block of values to analyze. The block must be either a single column or row; it must not contain labels.
Output Block	identifies the upper left cell where Quattro Pro starts the resulting output table; the table will extend to the right and downward as needed. If there aren't enough values to forecast a value, Quattro Pro returns NA (Not Available).
Interval	specifies the number of values to include in the moving average; the default is 3.
Standard Errors	specifies whether to include a column in the output table for standard error values.

See Also

Advanced Analysis Tools List
Using the Advanced Analysis Tools

Exponential Smoothing

The Exponential Smoothing tool, like the Moving Average tool, smooths out variations in a data series. After using the Moving Average tool to create a set of forecast values, you can use the Exponential Smoothing tool to smooth each forecast value and adjust for error in the prior period.

Input Block	is the block of values to smooth. The block must be either a single column or row; it must not contain labels.
Output Block	identifies the upper left cell where Quattro Pro starts the block of smoothed values. If there aren't enough values to forecast a value or calculate a standard error, Quattro Pro returns NA (Not Available).
Damping Factor	specifies the constant used to smooth the data. The Damping Factor indicates the percentage for error that the prior forecast can be adjusted; the default is 30%.
Standard Errors	specifies whether to include a column in the output table for standard error values.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Creating Histograms

You can use the Histogram tool to view the shape of a sample population's distribution. It calculates the probability and cumulative distributions for a sample population, based on a series of bins you define. Histograms are often used to graph the data in frequency distribution tables.

Input Block	is the block of data to analyze. The block can consist of one or more columns or rows of numeric values. (The block should not contain labels.)
Bin Block	is a block of values that defines the bin ranges for the data. If you don't specify a bin block, bins are distributed evenly from the minimum to the maximum value in the input block, with the number of bins equal to the square root of the number of values in the input block.
Cumulative Percentage	specifies whether to generate a column for cumulative percentages in the output table.
Pareto	specifies whether to create an output table with data in both descending frequency order and ascending Bin Block order. (If you don't check Pareto, the output table displays data only in ascending Bin Block order.)

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Generating Random Numbers

The Random Number tool generates random variables from various distributions. You can select one of seven types of distributions.

Output Block	identifies the upper left cell where Quattro Pro starts the resulting output table; the table will extend to the right and downward as needed.
Variables	(columns) identifies the number of columns in the output block.
Points	(rows) identifies the number of rows in the output block. (Points is not an option for the Patterned Distribution Type.)
Seed	is a starting value for the random number generation algorithm. You can use Seed to generate the same sample of random numbers again.

The remaining options in the dialog box vary depending on the Distribution Type you select:

Distribution	Description
Uniform	Every value has an equal probability of being selected. Specify the upper bound and lower bound limits for the distribution.
Normal	Has the qualities of a symmetrical, bell-shaped curve. Specify two parameters of the distribution: the mean and standard deviation.
Bernoulli	Has two possible outcomes, failure or success, represented by 0 and 1, respectively. Specify the probability of the outcome of the Bernoulli trial.
Binomial	Represents the distribution of successful outcomes in a given number of independent Bernoulli trials; for example, a coin toss experiment produces a binomial distribution. Specify the probability of the outcome of each trial and the number of trials.
Poisson	The distribution of values in any interval depends on the length of the interval and the constant Lambda, the expected number of occurrences in an interval.
Patterned	Produces a pattern of repeated values and sequences. Specify lower and upper bounds for the values, a step value (an increment between the lower and upper bounds), the number of times each number is repeated, and the number of times each sequence is repeated.
Discrete	Generates random numbers drawn from a sample block of values; each sample value has a discrete probability value, and the sum of the probability values equals 1.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Uniform Distribution Options

Every value has an equal probability of being selected. Specify the upper bound and lower bound limits for the distribution.

Lower Bound	a value indicating the lower bound on the set of numbers to generate; the default is 0
Upper Bound	a value indicating the upper bound on the set of numbers to generate; the default is 1

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Normal Distribution Options

Has the qualities of a symmetrical, bell-shaped curve. Specify two parameters of the distribution: the mean and standard deviation.

Mean	a value indicating the mean of the set of numbers to generate; the default is 100
Std Deviation	a value indicating the standard deviation of the set of numbers to generate; the default is 10

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Bernoulli Distribution Options

Has two possible outcomes, failure or success, represented by 0 and 1, respectively. Specify the probability of the outcome of the Bernoulli trial.

Probability	a value indicating the probability of success on each trial; the value must be greater than or equal to 0% and less than or equal to 100%; the default is 50%

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Binomial Distribution Options

Represents the distribution of successful outcomes in a given number of independent Bernoulli trials; for example, a coin toss experiment produces a binomial distribution. Specify the probability of the outcome of each trial and the number of trials.

Probability	a value indicating the probability of success on each trial; the value must be greater than or equal to 0% and less than or equal to 100%; the default is 50%
Trials	the number of trials to run; the default is 10

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Poisson Distribution Options

The distribution of values in any interval depends on the length of the interval and the constant Lambda, the expected number of occurrences in an interval.

Lambda	a parameter to the Poisson distribution representing the expected number of occurrences in an interval; the default is 1

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Patterned Distribution Options

Produces a pattern of repeated values and sequences. Specify lower and upper bounds for the values, a step value (an increment between the lower and upper bounds), the number of times each number is repeated, and the number of times each sequence is repeated.

From	a value indicating the lower bound on the set of numbers to generate; the default is 1
To	a value indicating the upper bound on the set of numbers to generate; the default is 100
Step	increment used to generate numbers between From and To; the default is 1
Number Repeat Rate	the number of times to repeat each value; the default is 1
Sequence Repeat Rate	the number of times to repeat each sequence of values; the default is 1

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Discrete Distribution Options

Generates random numbers drawn from a sample block of values; each sample value has a discrete probability value.

Probability Range	a two-column block containing numbers in the first column and probability values (between 0 and 1) in the second column. The sum of the probability values must equal 1.

See Also

Generating Random Numbers

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Ranking Data

The Rank and Percentile tool computes the ordinal and percent rank of each value in a sample.

Input Block	is the block of data you want to analyze.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Group By	specifies whether data in the input block is grouped by Row or Column.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

Note: To determine the relative ranking and the percentage ranking of values in a sample, you can also use the separate @functions @RANK and @PERCENTRANK, respectively.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Performing Linear Regression

The Advanced Regression tool performs linear regression analysis, which shows the relationship between a set of independent variables and a certain dependent variable. In addition to creating a summary output table, the Advanced Regression tool creates output tables for residuals and probability data. For simpler regression analysis, see Performing Regression Analysis

Y Range	is the block of y values (the dependent variables). The y values should be in a single column.
X Range	is the block of x values (the independent variables). The x values should be in one or more columns.
Labels	indicates whether labels are in the first row of both Y Range and X Range.
Confidence	displays the confidence level of the means for the input block; the default is 95%.
Summary Output	identifies the upper left cell of the resulting summary table. (When you specify the block, allow at least seven columns.)
Residual Output	identifies the upper left cell of the resulting residuals output table. (When you specify the block, allow at least four columns.)
Probability Output	identifies the upper left cell of the resulting probability output table. (When you specify the block, allow at least two columns.)
Residuals	specifies whether you want to include residuals in the residuals output table.
Std Residuals	specifies whether you want to include standardized residuals in the residuals output table.
Y-intercept 0	specifies whether you want the regression line to pass through the y-axis at zero. If you don't specify the y-intercept as a zero, Quattro Pro generates a y-intercept from an internal set of variables.

Note: Output blocks can't overlap; make sure the ones you specify allow for the recommended number of columns. Try placing output blocks side by side instead of on top of one another.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Sampling Data

The Sampling tool creates a random or periodic sample from a given population of values. Use the Sampling tool to work with a subset of data from a large database. If the population consists of periodic data, you can create a periodic sample to select data from a particular cycle.

Input Block	is the block of data to sample. For periodic sampling of an input block with multiple columns, Quattro Pro samples each column in order from left to right.
Output Block(s)	identifies the upper left cell where Quattro Pro places the resulting output table(s).
Sample Type	specifies the type of sample, Random or Periodic; the default is Random. In a random sample, every value in the input block has an equal probability of being selected. A periodic sample selects every nth value from the input block.
No. of Samples	(for random sampling only) specifies the number of values in the output block; the default is 1.
Interval	(for periodic sampling only) specifies the periodic interval used to sample data; the default is 1. For example, 3 means every third value from the input block is selected.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Testing the Means Of Small Samples

t-Tests test the hypothesis that means from two small samples are equal. Quattro Pro provides t-Tests for samples with matched pairs, for independent samples with equal variances, and for independent samples with unequal variances.

t-Test Type	specifies the type of t-Test. Select either Paired Two-Sample for Means, Two-Sample with Equal Variances, or Two-Sample with Unequal Variances.
Variable 1 Input	is the first column or row of numeric data to compare. The block should be a single column or row.
Variable 2 Input	is the second column or row of numeric data to compare. The block should be a single column or row.
Alpha	is the significance level of the test; the default is 5%.
Difference	is a value indicating the hypothesized difference (the null hypothesis) in the means between Variable 1 Input and Variable 2 Input; the default is zero. Difference is not applicable if the t-Test Type is Two-Sample with Unequal Variances.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

t-Test:Paired Two-Sample For Means

The t-Test: Paired Two-Sample for Means tool performs a paired two-sample Student's t-Test for means. Each value from one data set is paired with a value from another set; both sets must have the same number of values.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

t-Test: Two-Sample with Equal Variances

The t-Test: Two-Sample with Equal Variances tool performs a two-sample Student's t-Test for means using two independent (rather than paired) samples with equal variances.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

t-Test: Two-Sample with Unequal Variances

The t-Test: Two-Sample with Unequal Variances tool performs a two-sample Student's t-Test for means using two independent (rather than paired) samples with unequal variances.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Testing the Means Of Large Samples

z-Tests test the hypothesis that the means of two large samples (or populations) are equal. The z-Test: Two-Sample for Means tool performs a two-sample z-test. You must specify the known variances for each sample.

Variable 1 Input	is the first column or row of data to compare.
Variable 2 Input	is the second column or row of data to compare.
Variable 1 Variance	is a value indicating the variance of Variable 1 Input.
Variable 2 Variance	is a value indicating the variance of Variable 2 Input.
Alpha	is the significance level for the test; the default is 5%.
Difference	is a value indicating the hypothesized difference (the null hypothesis) in the means between Variable 1 Input and Variable 2 Input; the default is zero.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Analyzing Variance

Analysis of variance (Anova) tools help determine if two or more samples come from the same population by calculating the F-statistic, the ratio of the mean variance between samples to the mean variance within samples. The Anova tools are commonly used in cases involving more than two samples (where t-Tests cannot be used).

Quattro Pro provides three analysis tools for analyzing the variance between data sets:


Anova: One-Way


Anova: Two-Way with Replication


Anova: Two-Way Without Replication

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Anova: One-Way

The Anova: One-Way tool performs a single-factor analysis of variance with equal numbers of observations.

Input Block	is the block of data to analyze. The block must contain two or more sets of data arranged in columns or rows.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Alpha	is the significance level at which to evaluate values for the F-statistic; the default is 5%.
Group By	specifies whether data in the input block is grouped by Row or Column.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Anova: Two-Way with Replication

The Anova: Two-Way with Replication tool performs a two-factor analysis of variance using more than one sample for each group of data.

Input Block	must contain two or more sets of data arranged in columns. The first row must contain labels for each group. The first column must contain row labels to indicate the beginning of each sample.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Rows Per Sample	indicates the number of rows in each sample.
Alpha	is the significance level at which to evaluate values for the F-statistic; the default is 5%.

See Also

Advanced Analysis Tools List
Using the Advanced Analysis Tools

Anova: Two-Way Without Replication

Anova: Two-Way without Replication performs a two-factor analysis of variance using only one sample for each group of data.

Input Block	is the block of data to analyze. The block must contain two or more sets of data arranged in columns or rows.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Alpha	is the significance level at which to evaluate values for the F-statistic; the default is 5%.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Comparing Variances Of Two Samples

The F-test tool compares the population variances of two data sets. If the variances of two data sets are significantly different you can hypothesize the samples were drawn from different parent populations.

Variable 1 Input	is the first column or row of numeric data to compare.
Variable 2 Input	is the second column or row of numeric data to compare.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Labels	specifies whether the first row or column of the input block contains labels. If the input block you specified does not contain labels, leave Labels unchecked; Quattro Pro generates appropriate column and row labels for the output table.

See Also

Advanced Analysis Tools List
Using the Advanced Analysis Tools

Generating an Amortization Schedule

The Amortization Schedule Generator tool creates a table showing the amortization schedule for a mortgage. The table shows principal and interest payments, and interest paid from start to date and from year to date.

Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Interest Rate (%)	is the yearly interest rate; the default is 12%.
Term	(years) is the number of years of the loan; the default is 30.
Original Balance	is the original loan balance; the default is $100,000.
Ending Balance	is the balance at loan completion; the default is $0.00.
Last Year	is the last year through which the amortization table is generated; the default is the value entered for Number of Years, which generates a table through the end of the loan.

You can enter fractions for Number of Years and Last Year to represent months. For example, 18+5/12 represents 18 years and 5 months; 100/12 represents 100 months.

After you choose OK, wait for the tool to generate the amortization table. The table takes longer to generate for long-term loans.

The table consists of two sections. The top section is the input section of the table. Most of the values in the input section correspond to options in the Amortization Schedule Generator dialog box. One additional value, 1st PMT, sets the first payment date for the mortgage. Change any of these input values to customize the bottom section of the table.

The bottom section of the table consists of 10 columns:

Column	Description
Pmt #	Number of payments from 1 to the last payment of the loan or the last one in the last year specified by Last Year
Date	Date on which the corresponding payment number is due (mortgage payments are typically due on the first day of the month following the month covered by the payment); the default date format is Month-Year (MMM-YY)
Yr Rate	Equivalent yearly interest rate for the period; for fixed rate loans this rate does not change; for variable rate loans, specify an estimated average interest rate when you create the table, then replace individual monthly rate values later by overwriting the calculated values
P&I Payment	Shows the principal and interest payment for the corresponding payment period. This will not vary for fixed loans, but would vary with variable interest loans or other variable loans such as graduated payment loans. In the case of "graduated payment" loans, you can type in the actual P&I payment to replace the calculated value. Even if this results in a negative amortization, the new Balance and other columns calculate correctly. (Negative amortization occurs when the P&I payment is insufficient to cover the interest charged for the period, and so the loan balance increases by the shortage amount.)
Principal	Principal portion of the P&I payment.
Interest	Interest portion of the P&I payment.
Extra Prin	Amount of extra principal per pay period. If you intend to pay extra principal for some portion of the loan, enter the amount in this column at the row of the corresponding payment number. The extra principal amount is included in every

	row that follows. If extra principal payments vary month to month, type in the actual extra principal amount wherever it changes.
New Balance	Balance after each payment. The balance decreases with each payment, and eventually reaches zero (or Ending Balance, if it is not zero). The amortization table stops short of zero if Last Year precedes the end of the loan.
Cum. Interest	Cumulative interest paid through the corresponding payment
Yearly Total Int	Cumulative interest paid for each calendar year. Use this column to calculate the yearly interest paid for tax purposes.

When you create the amortization table, you may need to increase the width of columns in the table to display the data. You can use the Fit button in the notebook SpeedBar.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Refinancing a Mortgage

The Mortgage Refinancing tool creates a table of information allowing you to compare prospective loans with the current mortgage you want to refinance.

Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Current Loan: Remaining Term	is the number of years remaining in the loan; the default is 30.
Current Loan: Balance	is the current loan balance; the default is $100,000.
Current Loan: Rate	is the current yearly interest rate; the default is 12%.
Candidate Loan: Rate	is the annual interest rate on the candidate loan; the default is 10%.
Candidate Loan: Fees (%)	are percentage fees ("points") for the candidate loan; the default is 0.

The table created by the Mortgage Refinancing tool consists of two sections: Current Loan and Candidate Loans.

In the Current Loan section, Current Balance, Current Rate, and Rem. Term take values from the Mortgage Refinancing dialog box. If you've paid extra principal at any time during the current loan, overwrite the value calculated for Equivalent P&I with the value of your actual monthly principal and interest. Enter a value for Future Value if the mortgage requires a balloon payment. Enter any loan-related fixed monthly fees in the Fixed Mo. Loan Fees cell, such as private mortgage insurance; do not include taxes or hazard insurance fees, which are not loan-related.

The Candidate Loans section of the table consists of 19 columns:

Column	Description
Loan Description	Name of the candidate loan.
Current Balance	Balance of the current loan.
Cash Out	Cash removed from equity (a positive value), or cash paid up front into the loan to pay down balance or to prevent financing or loan fees (a negative value).
Pct Fees	Percentage fees ("points") charged by the lender (for example, 1.75%).
Fixed Fees	Fixed fees involved in the refinancing; if a loss (for example, a loan prepayment penalty), enter a positive value; if a gain (for example, a prorated return of prepaid Private Mortgage Insurance from payoff of current loan), enter a negative number.
Loan Amount	Amount of the candidate loan.
Rate	Annual interest rate of the candidate loan.
Term (yrs)	Loan term in years.
Future Value	Future value of the candidate loan; enter a positive value if the loan requires a balloon payment.
P&I Payment	Principal and interest payment for each period.
Fixed Mo. Loan Fees	Fixed monthly loan fees (such as private mortgage insurance).
Loan Pmt	Monthly payment (Loan Pmt = P&I Payment + Fixed Mo. Loan Fees).
Savings: Gross	The difference in the monthly payment of the candidate loan and the current loan; if negative, the candidate's monthly payment is higher; if positive, the candidate's monthly payment is lower and all gross savings are

	applied as extra principal.
Term +/-	Portion of gross savings due to lengthening the term of the loan; if the term is lengthened, Term +/- is negative; if the term is shortened, Term +/- is positive.
Cash I/O	Portion of gross savings due to closing costs paid up front (a negative number), or the amount of gross savings that has already been reduced by cash taken out (a positive number enclosed by "><", which does not affect the Net amount).
Net	Sum of savings from Gross, Term +/-, and Cash I/O.
Payback (Years): When	The initial estimate of the number of years it will take for the candidate loan to pay for the cost of refinancing; displays "Never" if the refinance loses money.
Loan Life	The number of years you plan to keep the property before selling it or refinancing.
Bal chg	The difference in the balance at the end of Loan Life of the candidate loan and the current loan due to extra principal paid on the candidate loan; a negative number represents money saved.

After you create the table, enter values for the candidate loan such as Loan Description, Cash Out (if you removed cash from equity or paid additional principal), Pct Fees (points for the candidate loan), Fixed Fees, Rate, Term, Future Value, Fixed Monthly Loan Fees, and Loan Life. All other values in the table are calculated.

Note: To ensure accurate calculation of the percentage fees and loan amount for the candidate loan, increase the number of recalculation iterations for the notebook from 1 to 5; right-click the notebook's title bar to display the notebook Object Inspector.

In the Payback section of the table, enter a number of years for Loan Life to indicate how long you plan to keep the candidate loan before selling or refinancing. Payback occurs when the balance of the candidate loan would be equal to that of the current loan, that is, when Bal Chg equals zero; for equal comparison of the loans, any cash put into the candidate loan is alternatively considered a lump sum extra principal on the current loan. You can accurately determine the payback period by trying alternate estimates in the Loan Life column until Bal Chg is near zero. (Bal Chg is a negative number when Loan Life is greater than the payback period.)

To compare several candidate loans with the current loan, copy the row for the first candidate loan into one or more rows below it, then enter information for each loan.

Gross savings indicates how much lower the monthly payment is for the candidate loan compared to the current loan; all gross savings are applied as extra principal. Net savings indicates money saved. It's possible for a candidate loan to have a lower monthly payment but still not save money. For example, a candidate loan can have a lower monthly payment simply by lengthening the loan term, but this doesn't result in net savings.

In general, there are three ways to handle the savings that result from refinancing:

1. Use the savings to pay extra principal for the life of the candidate loan.

2. Use some of the savings to pay extra principal on the candidate loan.

3. Keep the savings and pay no extra principal on the candidate loan.

By default, the Mortgage Refinancing tool uses the first method. The Net savings cell shows the savings for each payment period that is applied as extra principal. To use some of the savings to pay extra principal, reduce the amount in the Gross Savings cell (overwrite the formula). For example, if the Gross savings cell shows a value of $125, enter $100 to use $100 of the savings to pay extra principal each month (the monthly payment of the candidate loan is still $25 lower than the current loan payment). To keep the savings and pay no extra principal, enter zero in the Net savings cell (overwrite the formula).

When you create a table using the Mortgage Refinancing tool, you may need to increase the width of columns in the table to display the data. You can use the Fit button in the notebook SpeedBar.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Performing Fourier Analysis

The Fourier tool performs a fast Fourier transformation on a block of data. Fourier transforms are commonly used to solve partial differential equations.

Input Block	is the block of values to transform. The block can contain real or complex numbers. (Complex numbers can be in x+yi, x+iy, x+yj, or x+jy format.) The number of values in the input block must be a power of 2 (for example, 2, 4, 8, 16,...) between 2 and 1024, inclusive; you can pad the input block with zeros so that the number of values is a power of 2.
Output Block	identifies the upper left cell where Quattro Pro places the resulting output table.
Inverse FFT	specifies whether to perform the inverse transformation on data in the input block. If the input block has already been transformed, check Inverse to set the values back to their original state.

See Also

Advanced Analysis Tools List

Using the Advanced Analysis Tools

Array Formulas

Array B2..D3 in the next figure contains data in three columns and two rows that can be worked with independently, not just as a whole block.

	A	B	C	D
1				
2		3	5	7
3		9	11	13
4		12	16	20

There are two ways you can get column totals for this array. You can enter @SUM(B2..B3) in B4, then copy it to C4 and D4. Or, you can select B4 and type this formula in the input line: `+B2..D2+B3..D3`.

When you enter the formula, Quattro Pro recognizes you're specifying arrays and converts the formula to an @function, @ARRAY. You can type @ARRAY and its arguments directly, if you prefer.

The formula in B7, @ARRAY(B2..D2+B3..D3), contains the same information as three @SUM formulas. It tells Quattro Pro to add the values in each column of the array and show the results in the formula cell and the two cells to its right.

Because this formula contains arrays instead of single values, it is called an array formula. The results of an array formula appear in a block of cells, called the output block or output array. The number of columns and rows in the output block depend on the dimensions of the arrays in the array formula. The following sections tell more about the output array and how to limit its size.

The following mathematical and logical operators determine how one value relates to a second: +, -, *, /, ^, =, <>, <=, >=, <, >, #AND#, #OR#, and &. Either or both values can be a block array or array constant.

See Also

Array Features

@ARRAY

Block Arrays

Block arrays are arrays given as variables, block references instead of exact values. To use block arrays in formulas, enter a block and another block or value separated by a mathematical or logical operator. The operator indicates the operation to perform on matching cells of each block or the relationship between them. For example, - (minus) means "subtract each cell of the second block from the equivalent cell of the first block." You need to enter only the first block (preceded by a +), the operator, and the second block. Quattro Pro adds the @ARRAY @function and parentheses. Block arrays can be contiguous or noncontiguous, but they must be 2-D.

When you're working with rows, array results appear in the @ARRAY formula cell and the two cells below it. For columns, the results appear beside the @ARRAY formula cell. You can mix columns and rows. You can enter more than two row or column blocks and use multiple operators.

As with other formulas, calculations are performed according to operator precedence: multiplication and division first, then addition and subtraction. For example, the following formula means "multiply C2 by D2 and add B2, multiply C3 by D3 and add B3, then multiply C4 by D4 and add B4:"

```
@ARRAY(B2..B4+C2..C4*D2..D4)
```

To change the calculation order, use parentheses.

You can use block names instead of references to specify arrays. If the block name is redefined to cover a different block of cells, then all @ARRAY formulas that reference it will automatically change.

See Also

Array Formulas

Array Features

Array Constants

Arrays can be block references, but they can also be array constants--series of exact values separated by row and column delimiters and enclosed in braces {}. Numbers in the same row are separated by semicolons or the argument separator specified in the application Object Inspector (through the Punctuation selection of the International property). Rows are separated by the pipe symbol (|), Shift+\. For example, the values in the next figure are indicated by this array constant: {1;2;3;4|5;6;7;8|9;10;11;12}.

```
 1   2   3   4
 5   6   7   8
 9  10  11  12
```

This array constant specifies values in the second row of the previous figure: {5;6;7;8}. This array constant specifies values in the third column: {3|7|11}.

Array constants can be used together in formulas or mixed with blocks.

See Also
Array Features

Arrays as @Function and Macro Arguments

You can use block arrays and array constants as arguments for other @functions besides @ARRAY. Virtually any @function that accepts block arguments can handle array operations, although different types of @functions handle them differently.

@SUM, @AVG, and other statistical @functions always return a single value, no matter how large an array you specify. For example, `@SUM(C4..C8*D4..D8)` means "multiply each value in C4..C8 by the equivalent value in D4..D8 and add the results."

The database @functions, such as @DSUM, take three arguments: a database block, the column to operate on, and a search criteria block. The first and third arguments must be blocks, not array constants. The second argument can be an array; the @function is calculated once for each value in the array.

@ABS, @SQRT and many other @functions normally take one argument which is a single value or an expression which results in a single value, not a block. If you enter a block or array constant for these @functions, Quattro Pro converts the @function statement into an argument for @ARRAY and provides a value for each row or column of the array.

Using arrays as arguments for index, lookup, and related @functions can be convenient when writing and debugging macros. For example, `@ARRAY(@CELLPOINTER(A1..A16)` returns a whole column of information about E1 when you select that cell and press F9 to recalculate; cells A1 through 16 contain @CELLPOINTER target attributes. While the previous formula referes to an attribute list in the notebook, you can also enter attributes as an array constant; for example, `{"contents";"type";"prefix"}`.

See Also
Array Features

Customizing Graph Properties

To customize graphs, right-click each graph element and display its Object Inspector. Each of the topic jumps listed below displays information on the properties in each Object Inspector.

See Also

Building Graphs

Enhancing Graphs

Analytical Graphing

Displaying Graph Object Inspectors

Graph Object Inspectors are available only in the graph window. There are three ways to bring a graph into the graph window:


Double-click a floating graph on the spreadsheet page.


Double-click the graph icon on the Graphs page.


Choose Graph|Edit anywhere in the notebook, select the graph name from the list that appears, and choose OK.

To display an Object Inspector, right-click the part of the graph you want to change and choose the Properties command in a SpeedMenu, or select the graph element, then choose Property|Current Object. You can use other commands in the Object Inspector to quickly display the Graph Window, Graph Setup and Background (called Graph Setup in the menu), Graph Pane, Graph Legend, and X-axis and Y-axis Object Inspectors.

Object Inspectors have a standard format: On the left is a list of the properties available for the selected object. When you choose a property from the list, its options appear on the right. The best way to use an Object Inspector is to select a property from the list, change settings, select another property from the list, change settings, and so on. When you've made all the modifications you want, choose OK to apply all the changes simultaneously. To help you track changes, Quattro Pro highlights the property name in blue whenever you alter an option setting.

See Also
Changing Graph Title and Subtitle Properties

Changing the Legend Position

The Legend Position property controls legend display. The left option hides the legend. The middle option creates a horizontal legend and places it below the graph. The right option creates a vertical legend and places it to the right of the graph. To change the legend position,

1. Right-click the graph background and choose the Properties command to display the Object Inspector.
2. Click Legend Position.
3. Choose a legend position, then choose OK.

Once you display a legend, you can drag it to any position.

See Also
Legend

Changing 3-D Options

You can hide or reveal the walls and base of a 3-D graph by changing 3-D options. There are four options that toggle off and on:

Show left wall hides or reveals the left wall.

Show back wall hides or reveals the back wall of the graph.

Show base hides the area under the series display.

Thick walls gives the graph 3-D walls. When thick walls are turned off, the walls look thin, like a sheet of paper.

A check mark next to an option indicates that display is on. To turn the option off, uncheck it. When wall or base display is on, you can right-click the area and choose the Properties command to display its properties, then change style options (see 3-D Walls and Base). When a wall or base display is off, you won't see the wall or the grid lines and you cannot display an Object Inspector.

To hide the wall, but display the grid lines, leave the wall display on (checked). Right-click the wall, choose the Properties command, choose Fill Style in the Object Inspector, then choose None. Choose OK. Grid lines on the walls of 3-D graphs are part of the Y-Axis; grid lines on the base are part of the X-Axis.

Changing Graph Background Appearance

The graph background is the area behind the title, the legend, the graph itself, and any drawing elements you added. Right-click any part of the background not covered by a graph element or drawn object and choose the Properties command, or choose Property|Graph Setup to display the Graph Setup and Background Object Inspector. Five properties in this inspector control the appearance of the graph background:

Box Type	chooses the style of box that encloses the graph background.
Fill Color	assigns a fill color to the background.
Bkg Color	assigns a second color to the background. (Bkg Color is not visible unless you choose a pattern or wash fill style.)
Fill Style	assigns a fill style to the background.
Border Color	controls the color of the box that encloses the background.

Note: On the spreadsheet page, graph background Box Type and Border Color properties are overridden by the Box Type and Border Color properties of the floating Graph Object.

Axis Properties Overview

All graphs except pie, column, and text graphs have two references for plotting data, the x-axis and the y-axis. The x-axis is the horizontal line that forms the bottom border of the graph pane. This axis usually has fixed points of reference that are often associated with time.

The y-axis is the vertical line on the left side of the graph. It contains a scale, or range of numbers against which a series is plotted. When you create a graph, Quattro Pro automatically scales the y-axis to include the highest and lowest values in all the series.

There are a few exceptions to this format:


Rotated graphs have a vertical x-axis and a horizontal y-axis. Quattro Pro reverses the axes of rotated graphs automatically.


In XY graphs (scatter diagrams), the x-axis series is data, not labels. Quattro Pro gives the x-axis a scale to match the data. When you right-click the x-axis of an XY graph and choose the Properties command, Quattro Pro displays an Object Inspector that lists the same properties as the y-axis Object Inspector. See Y-Axis Properties for a description of these properties.


If the graph is a 2-D bar, line, or area graph, you can assign any series to a secondary y-axis, which then appears on the right side of the graph (see Plotting Series on the Secondary Y-Axis for details). Quattro Pro scales this axis to accommodate the highest and lowest values in the series.

See Also

Changing X-Axis Properties

Changing Y-Axis Properties

Changing X-Axis Properties

When you right-click the x-axis of most bar, line, and area graphs and choose the Properties command, Quattro Pro displays the x-axis Object Inspector. There are seven properties; see X-Axis for a description of each property. The following topics explain how to use x-axis properties:

Selecting the Axis Label Series

Choosing a Tick Style

Controlling Label Display

Customizing Axis Labels

Choosing Major Grid Style Options

See Also

Axis Properties Overview

Changing Y-Axis Properties

Selecting the Axis Label Series

If you didn't select an x-axis series when you created the graph (see Rearranging Data Series), you can add a label to each x-axis division through the X-Axis Series property.

1. Enter axis labels in a spreadsheet page, if a suitable block of labels doesn't already exist.
2. Right-click the x-axis and choose the Properties command, or choose Property|X-Axis. The X-Axis Series property is already selected in the Object Inspector.
3. Double-click the Select Range edit field, then point to the block of axis labels on the spreadsheet page. If your labels are not in a continuous row or column, hold down the Ctrl key as you select each cell or sub-block.
4. Press Enter to return to the Object Inspector (or type the coordinates directly in the edit field), then choose OK.

See Also

Changing X-Axis Properties

Axis Properties Overview

Changing Y-Axis Properties

Choosing an X-Axis Tick Style

Tick marks are the short vertical lines that separate the major divisions of the x-axis. The Tick Style option controls the display and placement of tick marks and axis labels.

There are four tick styles displayed in the Object Inspector. To change the tick style,

1. Right-click the x-axis and choose the Properties command to display the Object Inspector.
2. Choose Tick Options.
3. Click a tick style, then choose OK.

See Also

Changing X-Axis Properties

Axis Properties Overview

Changing Y-Axis Properties

Controlling Label Display

If you have long labels or plot many series, x-axis labels might overlap. There are several ways to correct this without changing the x-axis series:

1. Right-click the x-axis and choose the Properties command to display the Object Inspector.
2. Choose Tick Options, then use one of these options to adjust label display:

Display Labels toggles label display off and on. To turn label display off, uncheck this option; to turn them on, check it again.

Number of rows places the labels in a single row, or in two or three staggered rows.

No Overlapping Labels eliminates some of the labels when label text would otherwise overlap. The default setting is on (checked). Uncheck this option to activate Skip __ Labels.

Skip __ Labels lets you choose the number of labels to eliminate, to prevent overlap. For example, if you skip two labels, the first label in the x-axis series is displayed, the second and third labels are skipped, the fourth label is displayed, the fifth and sixth labels are skipped, and so on. To specify the number of labels to skip, uncheck No Overlapping Labels, then type a number in the edit field or click the up and down arrows.

Length Limit lets you specify the maximum number of characters displayed in a label. Any additional characters will not appear in the label. To set a length limit, check this option, then enter the number of characters in the edit field that appears (or use the arrows to the right of the edit field to adjust the limit).

3. Choose OK.

See Also

Changing X-Axis Properties

Axis Properties Overview

Changing Y-Axis Properties

Customizing Axis Labels

You can change the appearance of axis label text by selecting a new text color, text background color, text font, or text style.

Text Bkg Color

Text Color

Text Font

Text Style

See Also

Changing X-Axis Properties

Axis Properties Overview

Changing Y-Axis Properties

Choosing Major Grid Style Options

Major grid lines separate the divisions of the x-axis. (They partially or fully cover tick marks.) You control the style and color of these grid lines with the Major Grid Style property. To change this property,

1. Right-click the x-axis and choose the Properties command.
2. Choose Major Grid Style. Line style options appear automatically.
3. Choose a line style. (The choice in the upper left corner turns grid lines off.)
4. Click Color.
5. Select a line color from the palette, or use the color scales to create your own (see Using Color Palettes for directions). Then choose OK.

See Also

Changing X-Axis Properties

Axis Properties Overview

Changing Y-Axis Properties

Changing Y-Axis Properties

When you right-click the y-axis of a standard graph and choose the Properties command, the y-axis Object Inspector appears. There are nine properties; see Y-Axis for a description of each property. The following topics explain how to use y-axis properties:

Changing the Y-Axis Scale

Setting a Zero Line

Using Show Units

Logarithmically Scaling an Axis

Selecting a Numeric Format

Choosing Major and Minor Grid Styles

See Also

Axis Properties Overview

Changing X-Axis Properties

Changing the Y-Axis Scale

Quattro Pro automatically adjusts the y-axis scale to fit the range of values plotted against it. You can change the High, Low, Increment, and No. (number) of Minors values to fine-tune the graph or to zoom in on a specific area.

High sets the highest value displayed on the scale. (If you set this lower than a value in a data series, the plot for the value will appear to run off the top of the graph.)

Low sets the lowest value displayed on the scale. (If you set this higher than a value in a data series, the plot for the value will appear to run off the bottom of the graph.)

Increment determines the numeric "distance" between major y-axis divisions. Y-axis labels, major tick marks, and major grid lines are placed at these divisions.

No. of Minors specifies the number of evenly-spaced minor tick marks and/or minor grid lines between the major y-axis divisions.

To adjust the scale,

1. Right-click the y-axis and choose the Properties command, or choose Property|Y-Axis. Scale options appear automatically.
2. Select the number in the High edit field, then type the highest number you want to appear on the axis. The new setting overwrites the old. Notice that Automatic is no longer selected.
3. Select the number in the Low edit field, then type the lowest number you want to appear on the axis.
4. Select the number in the Increment edit field, then enter the scale distance between major tick marks. Enter a zero if you want Quattro Pro to calculate an increment using the new High and Low settings.
5. Select the entry in the No. of Minors field, then type over it to enter the number of minor tick marks to place between major tick marks.
6. Choose OK to apply your changes.

To set a descending axis scale, reverse the high and low values. To reset the scale back to its default setting, check Automatic, then choose OK.

See Also

Axis Properties Overview

Changing Y-Axis Properties

Setting a Zero Line

Zero Line appears as a Scale option when you right-click the y-axis of a variance graph and choose the Properties command. The zero line provides the frame of reference for the "variance" that gives this graph type its name. Values less than the zero line setting project below the line; higher values extend above the line. The zero line can be set, for example, to show how temperatures vary from the freezing point on the Fahrenheit scale.

To change the zero line value, right-click the y-axis of the variance graph and choose the Properties command, or choose Property|Y-Axis. The zero line option appears under No. of Minors. Set the zero line to a value no greater than the High scale setting, and no less than the Low scale setting, then choose OK.

See Also

Axis Properties Overview

Changing Y-Axis Properties

Using Show Units

If the numbers on the y-axis scale are 1000 or greater, you can simplify the labels with the Show Units option. Show Units displays only the first characters of the label, and automatically adds the appropriate units title in parentheses along the y-axis; for example, "(thousands)" or "(millions)."

To use this option, right-click the y-axis and choose the Properties command. The scale property is automatically selected. Check Show Units. (To turn this feature off again, uncheck it.) Choose OK.

To change the font, color, or other properties of the Show Units title, right-click it and choose the Properties command to display the axis title Object Inspector (see Axis Titles for details).

See Also

Axis Properties Overview

Changing Y-Axis Properties

Logarithmically Scaling an Axis

In a logarithmically scaled axis, each major division of the axis represents 10 times the value of the previous major division. This type of scale is useful when you're plotting series with wide ranges in magnitude.

A logarithmic scale plots a steady increase or decrease as a straight line. Consequently, it is useful for demonstrating the rate of increase or decrease in values, such as population levels.

To use a logarithmic scale, right-click the y-axis and choose the Properties command or choose Property|Y-Axis. Scale options appear automatically in the Object Inspector. Under Scale Type, choose Log, then choose OK.

See Also

Axis Properties Overview

Changing Y-Axis Properties

Selecting a Numeric Format

To change the format of the values on the y-axis scale, right-click the y-axis and choose the Properties command, then choose the Numeric Format property. Select a format option, then choose OK. The options are the same as those presented in the block Object Inspector. See Numeric Format Property for a description of these options.

See Also

Axis Properties Overview

Changing Y-Axis Properties

Choosing Major and Minor Grid Styles

Major grid lines extend from each major tick point across the graph to the opposite side. Minor grid lines extend from each minor tick on the axis to the opposite side of the graph.

Grid lines appear behind the data series and help the viewer estimate series values.

The Major Grid Style and Minor Grid Style properties select new line styles and line colors for major and minor grid lines. To change a grid line style,

1. Right-click the axis and choose the Properties command to display the axis Object Inspector.
2. Choose Major Grid Style or Minor Grid Style. Line style options appear automatically.
3. Select a line style. (The blank selection in the upper left corner turns the grid off.)
4. Click the Color option and select a color from the palette, or use the scales to create a new color (see Selecting Fill, Background, and Border Colors for additional information).
5. Choose OK.

If major grid lines are too close together, choose the Scale property and increase the increment; if the lines are too far apart, decrease the increment. To change the number of grid lines between labels, increase or decrease the No. of Minors (see Changing the Y-Axis Scale for details).

See Also

Axis Properties Overview

Changing Y-Axis Properties

Fill and Border Properties

Fill and border properties are the fill color, background color, fill style, border color and border style. You find fill and border properties in the Object Inspectors for every solid object that appears in or on a graph.

A solid object has two areas, an interior and a border. The appearance of the interior is determined by the way fill color, background color and fill style properties are set. The border color and border style (or box type) control the appearance of the perimeter of the object.


Fill Style makes the box transparent (None), or fills it with a solid color, a two-color pattern, a wash, or an imported bitmap graphic.


Fill Color is the interior color of the object. If the fill style is a pattern, the fill color is the color of the pattern. If the fill style is a wash, the fill color is distributed over the background color.


Background (Bkg) Color is visible only if the fill style is a pattern or a wash. If the fill style is a pattern, the background color is the color *behind* the pattern. In a wash, the fill color is distributed over the background color.


Border Color is the color of the frame or box around the object.


Border Style sets the style or thickness of the frame around the object. (An object has either a Border Style or a Box Type.)


Box Type determines the look of the box around the object. The 12 choices range from "no frame" to several 3-D styles.

See Also

Selecting Fill, Background, and Border Colors

Selecting a Fill Style

Changing Box or Border Style

Filling an Object with a Bitmap

Making an Object Solid or Transparent

Selecting Fill, Background, and Border Colors

To select a fill color, background color, or border color for an object,

1. Right-click the object and choose the Properties command to display the Object Inspector.
2. Select Fill Color to choose the fill color, Bkg Color to select the background color, or Border Color to choose a border color. The Object Inspector displays a color palette, color scales, sliding scales for adjusting the color, and a sample box.
3. Use this color "as is," or adjust it with the color scales. (See Using Color Palettes for details).
4. Choose OK.

Selecting a Fill Style

To select a fill style for an object,

1. Right-click the object and choose the Properties command to display the Object Inspector.
2. Select Fill Style.
3. Choose a style:


None makes the object transparent.


Solid makes the object one solid color (the Fill Color).


Pattern gives you a choice of 24 two-color patterns. The color of the pattern is the Fill Color; the background behind the pattern is the Bkg Color.


Wash gives you a choice of 6 two-color wash styles. In a wash, the Fill Color is distributed over the Bkg Color.


Bitmap lets you fill the object with an imported bitmapped graphic (see Filling an Object with a Bitmap for more information).

4. Choose OK.

See Also

Fill Color

Bkg Color

Making an Object Solid or Transparent

Changing Box or Border Style

Every solid object is enclosed by either a box or border. The box or border style is controlled by the Box Type or Border Style property. To select a new box type or border style,

1 Right-click the object and choose the Properties command to display the Object Inspector.

2. Select Box Type or Border Style.

If the object has a Box Type property, you'll see 12 options, including transparent (the upper left choice), thick-sided boxes, round-cornered boxes, and 3-D styles.

If the object has a Border Style property, you'll see 9 options, including transparent (the upper left choice), dashed and dotted line styles, and solid line styles of various thicknesses.

3. Choose a style, then choose OK.

See Also

Border Color

Filling an Object with a Bitmap

To fill an object with an imported bitmapped graphic,

1. Right-click the object and choose the Properties command to display the Object Inspector.
2. Choose Fill Style.
3. Click Bitmap.
4. Type in the path and name of a bitmap file, or choose Browse and use the File Name and Directories lists to search for and select a bitmap file.
5. Choose whether you want to crop the picture (cut out all pieces that don't fit) or shrink it to fit within the object. If the object is larger than the bitmap, Shrink to Fit enlarges the bitmap to fill the object.
6. Choose OK.

Making an Object Transparent

To make a background transparent, as when you want a text box to disappear so the text appears to float on the graph background, set the Fill Style to None, and choose a blank Box Type or Border Style (click the upper left selection when box type or border style options are displayed).

Changing Graph Series Options

Series Options for 2-D and 3-D graphs appear in the Line Series Object Inspector, the Bar Series Object Inspector, and the Area Series Object Inspector. By customizing series options you can


select or change the data to plot (see Changing the Data Series for details)


place labels over bars in bar graphs, and over data point in line and area graphs (see Adding Labels to Bars and Data Points for details)


enter legend text or the coordinates where legend text is located (see Overriding a Legend Label for details)


create a combination graph (see Creating a Custom Combination Graph for details)


plot the series against a secondary y-axis (see Plotting Series on the Secondary Y-Axis for details)

To display the Object Inspector for a particular series, do any one of these:


Right-click the bar, line, or area that plots the series and choose the Properties command in the SpeedMenu.


Right-click the marker that represents the series in the legend box and choose the Properties command.


Choose Property|Series|series number.

Series Options is the first property in the Object Inspectors, so you see these settings as soon as it appears.

Changing the Data Series

When you right-click a series marker and choose the Properties command or choose Property|Series|series number, the coordinates of that series appear in the Data Series field. To assign different data to the series,

1. Double-click the edit field. The dialog box and graph move behind the spreadsheet.
2. Point to the block on the spreadsheet page that contains the labels and press Enter.
3. Choose OK. (You can also type the block coordinates directly into the edit field, then choose OK.)

Be sure to enter a block that has exactly the same number of cells as the other series in the graph.

Note: If you delete the block coordinates in the Data Series edit field, your graph will show a blank area (because you now have a series with no data). To delete a series, see Deleting a Series.

See Also

Adding a Series

Arranging the Legend Series

Graphing Grouped or Linked Data

Overriding a Legend Label

You add legend labels to a graph by defining a legend series. These labels appear in the legend box of 2-D graphs (except pie and column graphs), and in some 3-D graphs. In most 3-D graphs, legend labels appear along the z-axis. In multiple graphs, legend labels appear as "titles" of the sub-graphs that represent each series.

The Legend option lets you override the legend label for an individual data series, or enter a label when no legend series exists. To override a legend label, or enter a new one, type the label text or enter the label's cell address (A:E5, for example) in the Legend field, then choose OK.

You can change the font, style, and color of legend text using the Legend Object Inspector.

Creating a Custom Combination Graph

You can create a combination graph by overriding the graph type of a single series. To do this,

1. Right-click the series and choose the Properties command to display the Object Inspector, or choose Property|Series|series number in the menu bar. Series Options appear automatically.
2. Under Override Type, choose Bar, Line, or Area to specify how you want the series to be plotted. You can override a series on 2-D bar, line, variance, and high-low graphs, and on rotated bar and line graphs. (High-low graphs restrict the override to the fifth series or greater.)
3. Choose OK.

To return the series to the original graph type, choose Default.

Creating a multiple line or area graph

You can override the graph type on a multiple bar graph to create a multiple line graph or a multiple area graph. To do this,

1. Create a graph (see Creating a Graph in a Window for details).
2. Change the graph type to Multiple Bar (see Changing the Graph Type for details).
3. Right-click any bar in the graph and choose the Properties command to display the bar series Object Inspector, or choose Property|Series|series number. Series Options are displayed automatically.
4. Under Override Type, choose Line for a multiple line graph, or Area for a multiple area graph.
5. Choose OK.

See Also

Area Series

Bar Series

Line Series

Combination Graphs

Plotting Series on the Secondary Y-Axis

You can plot a series in 2-D bar, line, variance, and high-low graphs on a secondary y-axis. (High-low graphs restrict this feature to the fifth series or greater.) You may want to do this, for example, when the values in one series are much larger or smaller than those in other series. When there is a wide variation between series, the y-axis scale range may be so large that some series are barely visible above the x-axis. If you plot the series that is very different on the secondary y-axis, the primary y-axis adjusts to accommodate the other series.

You can also use this feature when the data is based on a different kind of scale. For example, in a high-low (open-close) volume graph, you can plot share prices on the primary y-axis, and the volume (number of stocks traded), on the secondary y-axis.

Right-click a series and choose the Properties command, or choose Property|Series|series number to display Series Options. Click the Secondary setting under Y-Axis and choose OK to plot the selected series against the y-axis on the right side of the graph.

The secondary y-axis has the same properties as the primary y-axis (see Y-Axis).

Customizing Bar Graphs

Bar Series Properties change the appearance of bar graphs. Through this Object Inspector, you can change series options, adjust the width, spacing, and overlap of all bars in the graph, and change the style attributes of bars in an individual series.

To display the Bar Series Object Inspector, right-click a bar in the series you want to change and choose the Properties command, or choose Property|Series|series number in the menu bar. The properties are:


Series Options control the way the selected series is plotted.


Bar Options control the bar width percentage, bar margin percentage, and bar overlap for all bars in the graph.


Fill Color selects the interior color of the bars.


Bkg Color is visible only if the bars have a pattern or wash fill style.


Fill Style can be none (transparent), a solid color, a pattern, a wash, or a bitmap graphic.


Border Color is the color of the frame around the bar.


Border Style chooses the line thickness of the border.

Comparison graphs have another property, Comparison Line, which chooses the color and style of the line that connects series boundaries.

Customizing Line Graphs

You can customize each line in a line graph with the Line Series Object Inspector. Right-click a line and choose the Properties command, or choose Property|Series|series number to display the Object Inspector with these properties:

Series Options control the way the selected series is plotted.

Marker Style changes the way each data point is marked.

Fill Color selects the color of the line graph background.

Bkg Color is visible only if the bars in a bar graph have a pattern or wash fill style.

Fill Style can be none (transparent), a solid color, a pattern, a wash, or a bitmap graphic.

Line Color lets you choose the color of the line that connects the points in the series.

Line Style lets you change the thickness of the line that connects the points. You can also choose from several dashed line styles.

Customizing Area and Surface Graphs

Area Series Properties change the appearance of area, surface, shaded surface, and contour graphs. To display the Object Inspector, right-click the area that represents the series you want to change and choose the Properties command, or choose Property|Series|series number in the menu bar. The properties are:

Series Options control the way the selected series is plotted.

Fill Color selects the interior color of the area.

Bkg Color is visible only if the area has a pattern or wash fill style.

Fill Style can be none (transparent), a solid color, a pattern, a wash, or a bitmap graphic.

Border Color is the color of the lines that form a border around the selected area.

Border Style chooses the thickness of the lines that surround an area.

Customizing Pie and Doughnut Graphs

Pie and doughnut graphs plot a single series. Each value in the series is a "slice" of the pie or doughnut, and is usually expressed as a percentage of the whole. To display the pie graph Object Inspector, right-click a slice and choose the Properties command. There are eleven properties:

Explode Slice applies only to the selected slice. (In multiple pie graphs, you explode the corresponding slice in each pie.)

Label Options, Text Color, Text Bkg Color, Text Font, and Text Style apply to the entire graph. (All pies are affected in multiple graphs.)

Fill Color, Bkg Color, Fill Style, Border Color, and Border Style apply only to the selected slice. (In multiple pie graphs, these properties affect the corresponding slice in each pie.)

Customizing Column Graphs

Column graphs plot a single series, and represent each value in the series as a section of the column. Right-click a column section and choose the Properties command to view its properties. There are ten:

Label Options, Text Color, Text Bkg Color, Text Font, and Text Style apply to the entire column graph. (All columns are affected in multiple graphs.)

Fill Color, Bkg Color, Fill Style, Border Color, and Border Style apply only to the selected section. (In multiple column graphs, these properties affect the corresponding section of each column.)

Customizing High-Low Graphs

High-low graphs illustrate the difference between corresponding values in two or more series. Though most often used in tracking daily stock prices, high-low graphs can be used whenever you want to compare the difference between pairs of values. To change the appearance of a high-low graph, right-click a high-low marker and choose the Properties command. The bar series Object Inspector appears (see Bar Series for details).

High-low graphs have one property that is unique: the Hi-Lo Bar Style. This property determines how high and low values are represented. There are four styles:

I-Beam	is the default style. High and low values determine each end of the I-beam. Open and close are represented by left and right tick marks, respectively.
Line	connects corresponding high and low values with a line, and shows open and close values as left and right tick marks.
Bar	give you a "bar and whisker" (or "candle") graph. A line connects high and low values. A bar spans the open and close values. When the close value is higher than the open, the bar is white. When the open value is higher than the close, the bar is blue. (Retain this light/dark relationship if you change fill colors.)
Marker	assigns different-colored markers to high, low, open, and close values, and connects each set of corresponding values with a line.

To change the style of a high-low graph,

1. Right-click any high-low marker and choose the Properties command (this property is global, so you don't have to select a specific series). The bar series Object Inspector appears.
2. Choose Hi-Lo Bar Style.
3. Select a bar style, then choose OK.

When you choose a Marker style high-low graph, you can use the line series Object Inspector to give each marker a different look.

1. Change the high-low graph type to line
2. Right-click a series and choose the Properties command, or choose Property|Series|series number to display the line Object Inspector.
3. Change the marker style, weight, and appearance.
4. Repeat this procedure for all the other series.
5. When you're finished, change the graph type back to High-Low.

Changing Graph Title and Subtitle Properties

The graph title and subtitle have three separate Object Inspectors, Graph Title Properties, Graph Subtitle Properties, and Graph Title Box Properties. To determine which Object Inspector you'll see, move (don't drag) the mouse pointer over the title and subtitle area. If you right-click when the pointer is an arrow, you can display Title Box properties. When you move the pointer over the graph title or subtitle, the pointer changes to an I-beam, and a right-click accesses title or subtitle properties.

What Is a Database?

A database is a collection of data records, for example:

An address book or phonebook (each business or person is one record)

A checkbook (each bank transaction is a record)

A notepad (each page is a record)

Each record breaks down into pieces of data called fields. A phonebook entry (record) consists of three fields: name, address, and phone number. Each field has a unique name, as shown in the table below:

Some Database Examples

Database	No. of Fields	Field Names
Phonebook	3	Name, Address, Phone Number
Address book	5	Name, Address, City, State, Postal Code
Checkbook	5	Check Number, Description, Credit, Debit, Balance
Notepad	1	Page

In Quattro Pro for Windows databases, each row is a record and each column is a field. You can use Data Modeling Desktop to create more elaborate crosstab report formats from a basic database.

See Also

Setting Up a Database

Searching for Records

Sorting Data

Setting Up a Database

Once you have an idea of the kind of data you want to include in your database, it's fairly easy to set up. The block containing the field names and data is called the database block. To speed data searches, you'll need space for a criteria table, which indicates the data you're searching for. You'll also need an output block to hold the results of data searches.

These topics describe how to set up each part of a database:

The Database Block

The Criteria Table

The Output Block

See Also

What Is a Database?

Searching for Records

The Database Block

The database block holds a Quattro Pro database, including field names and data records. Follow these rules when creating it:

The database block must be rectangular, although it can contain any number of blank cells.

The database block must be contiguous and all on one page.

Enter the field names in the top row of the database block, one name per column (each column of the database block is a field). Field names must be labels (text), less than 16 characters long, with no blank spaces at the beginning or end.

Make sure field names are different from each other. They should also be different from any block names you are using.

Each row of the database block beneath the field names is a record. Enter the first row of data directly below the field names without inserting a blank line or a row of decorative symbols.

Use the same type of data throughout each column. In other words, don't mix labels and numeric data in the same column.

See Also

Setting Up a Database

Searching for Records

Sorting Data

The Criteria Table

A criteria table consists of a row of field names, which must match one or more fields in the database block, and one or more rows of search criteria, either values or logical formulas:

Criteria in the same row indicate an AND query. Data must match all criteria.

Criteria in different rows indicate an OR query. Data can match criteria in one or more rows, but not necessarily all rows.

Criteria tables can combine AND and OR queries.

To create a criteria table:

1. In the first row of the table, copy the names of the fields you want to search through.
2. Underneath the appropriate field name(s), enter the data you want to search for. To search for an exact match, enter the value exactly as it appears in the database. To enter a condition, use a search formula.

   If you want more than one criterion met by each record, enter them all in the same row, under the appropriate field names.

   To search for records that meet any of the given criteria, use a separate row for each criterion.

See Also

AND Query Example

OR Query Example

AND and OR Query Example

Using Search Formulas in Criteria Tables

Using Wildcards in Search Criteria

Setting Up a Database

AND Query Example

Criteria in the same row indicate an AND query. Data must match all criteria.

AND Query Table

LAST_NAME	FIRST_NAME
Lyter	Kelly

Only records with LAST_NAME=Lyter and FIRST_NAME=Kelly match these criteria.

See Also

The Criteria Table

Using Search Formulas in Criteria Tables

Using Wildcards in Search Criteria

Setting Up a Database

OR Query Example

Criteria in different rows indicate an OR query. Data can match criteria in one or more rows, but not necessarily all rows.

OR Query Table

LAST_NAME	FIRST_NAME
Lyter	
	Barbara

Any record with LAST_NAME=Lyter or FIRST_NAME=Barbara matches these criteria. Records for Kelly Lyter, Barbara Lyter, and Barbara Wong all match these criteria.

See Also

The Criteria Table

Using Search Formulas in Criteria Tables

Using Wildcards in Search Criteria

Setting Up a Database

AND and OR Query Example

Criteria tables can combine AND and OR queries.

AND and OR Query Table

LAST_NAME	FIRST_NAME
Lyter	Kelly
	Barbara

Records for Kelly Lyter and anyone with FIRST_NAME=Barbara match these criteria. These records match the criteria: Kelly Lyter, Barbara Lyter, and Barbara Wong. These records are not matches: Steve Lyter, Kelly O'Toole.

See Also

The Criteria Table

Using Search Formulas in Criteria Tables

Using Wildcards in Search Criteria

Setting Up a Database

Using Search Formulas in Criteria Tables

You can use any of these operators and symbols in formulas:

= equals <> not equal

< less than > greater than

<= less than or equal

>= greater than or equal

#AND# both expressions must be true

#OR# either the first or the second expressions may be true

#NOT# the first expression is true, the second is false

You can also use wildcards in formulas to search for near matches.

See Also

Using Wildcards in Search Criteria

The Criteria Table

Setting Up a Database

Using Wildcards in Search Criteria

When you're searching for text, you can use special "wildcards" in your search formulas to search for near matches. Quattro Pro supports these wildcard symbols:

? specifies any single character in a label. For example, "t?p" finds tip, top, or tap, but not tape or stop.

***** specifies any number of characters to the end of a label. For example, "ten*" would find tender, tension, and tent, but not attention.

~ (tilde) searches for all labels in a field EXCEPT those that match the search condition. For example, CITY=~"Boston" finds all records that do not have Boston in the City field.

If you're using database forms, you can use ? and *, but not ~.

See Also

Using Search Formulas in Criteria Tables

The Criteria Table

Setting Up a Database

Using Database Forms

The Output Block

If you intend to copy matching records to another block with either the Extract or Unique command, you must first prepare the area to copy the records to (the output block):

1. Choose a blank area of the notebook that will not overlap your data or criteria table.
2. In the first row of the block, copy the name of each field that you want included in the output from the database. You can include as many field names as you want, in any order. Fields that you don't include won't be included in the output.

See Also

Setting Up a Database

Searching for Records

Searching for Records

Once you've created a Quattro Pro database, you can review records that meet specific search conditions, or search criteria. This process is called querying or searching a database.

To search a database:

1. Create a criteria table containing the conditions and exact matches to search for.
2. If you want to copy the matched records to another part of the notebook, set up an output block, an area with enough blank space to hold extracted data plus the names of the fields you're extracting. Then enter the field names in the first row of the output block in whatever order you like.
3. Choose Data|Query and specify the database block, including field names.

   If you want to specify a database block in another open notebook, use standard linking syntax. For example, `[DATA]A1..Z50` specifies block A1..Z50 in the notebook called DATA.

   You can also use linking syntax to query a database file created with Paradox. This lets you access database information without having to translate the database into a notebook file first. Just specify any block including the name of the file (and directory if different).
4. If your search formulas refer to field names, use Data|Query|Field Names to assign a block name to each field.
5. Specify the criteria table, including field names and all rows of criteria.
6. Specify the output block.
7. Choose a search option: Locate, Extract, Extract Unique, or Delete.

   Choose Locate to highlight matching records, Extract to copy them to the Output Block, Extract Unique to eliminate duplicate matching records when copying them, or Delete to erase matching records.

See Also

Setting Up a Database

Highlighting Matching Records

Copying Matching Records

Deleting Matching Records

Data|Query

Using External Databases

Highlighting Matching Records

To search through the database for all records that meet the conditions specified in the criteria table, choose Data|Query. Specify the database block, criteria table, and output block. Then, choose Locate.

Quattro Pro highlights the first record in the database block that meets the conditions. If there are no qualifying records, Quattro Pro beeps and returns you to the dialog box.

To highlight other records that meet your criteria, press Up Arrow or Down Arrow. Up Arrow moves to previously found matches, Down Arrow moves to the next match.

Press Esc to return to the Data|Query dialog box.

See Also

Copying Matching Records

Deleting Matching Records

Setting Up a Database

Data|Query

Searching for Records

Copying Matching Records

You can search through a Quattro Pro database for all records that meet the conditions specified in the criteria table and copy them to an output block:

1. Choose Data|Query. Specify the database block, criteria table, and output block.
2. Choose Extract to copy all matching records to the output block. Quattro Pro copies only those fields whose names are written on the first row of the output block.

   If you want to eliminate duplicate records (records that are identical in all fields), choose Extract Unique instead.

See Also

Highlighting Matching Records

Deleting Matching Records

Setting Up a Database

Data|Query

Searching for Records

Deleting Matching Records

You can search through a Quattro Pro database for all records that meet the conditions specified in the criteria table and delete them:

1. Choose Data|Query. Specify the database block and criteria table.
2. Choose Delete to find and delete all matching records. Quattro Pro erases all qualifying records from the database. Records underneath the deleted records move up to fill in the empty rows.

For best results, use Locate to check the result of a search before using Delete. If you accidentally delete a record, choose Edit|Undo (if enabled) to bring it back.

See Also

Highlighting Matching Records

Copying Matching Records

Setting Up a Database

Data|Query

Edit|Undo

Using @Functions with a Database

You can use a variety of @functions with Quattro Pro databases. These functions use database blocks and criteria tables like a database search.

See Also

Database @Functions

Setting Up a Database

Using Database Forms

If your notebook database has a column for each data field and a row for each record, you can create a form for entering and finding data records without programming. See these topics for details:

Creating and Editing Records

Searching for Records

Editing and Clearing Search Criteria

Creating and Editing Records

To edit an existing record or create a new one:

1. Choose Data|Form.
2. Specify the database block, including field and record labels.
3. Quattro Pro generates a form and displays the first record. The field names appear in the first column with edit fields for data beside them.
4. Edit the active record or choose another activity:

   New creates a new, blank record added after the last record.

   Delete erases the active record.

   Revert cancels any edits and restores original data for the active record.

   Go Next displays the next record.

   Go Previous displays the previous record.

   Search displays a search form (see Searching for Records for details).

   Close enters any changes from the current session and closes the form.

   Help displays a help topic.

See Also
Using Database Forms

Searching for Records

To find all records that meet a set of search criteria,

1. Choose Data|Form and enter the database block as described in the previous section.
2. Click Search to display a search form for that database.
3. Enter the search criteria for each field. You can find exact matches or look for a range of values. You type an operator followed by a value. These are acceptable operators:

=	equal to
<>	not equal to
>=	greater than or equal to
>	greater than
<=	less than or equal to
<	less than

   For example, if Housing is a field, enter >500 beside Housing to find all Housing expenses greater than $500. If you leave a field blank, Quattro Pro finds all instances of that field.

   You can search for strings as well as numbers. For example, if Month is a field and you want to find the record for February, type `February` in the edit field beside Month. Quattro Pro looks for any record that starts with the string February. Enter `=February` to find an exact match. `<February` means "Find any record that starts with a letter before F in the alphabet."

   You can use DOS * and ? wildcards (for details, see Using Wildcards in Search Criteria).
4. Click Go Next to find the first record that matches the criteria. Go Next returns the form to Edit mode. You can edit any record when it appears, then click Go Next again to find the second matching record, and so on. Click Go Previous to back up, still in Edit mode.
5. Click Close to close the form.

See Also
Using Database Forms

Editing and Clearing Search Criteria

To enter new search criteria or clear all criteria, click Search again. Then, edit the criteria or click Clear to end the search. You can click Go Next or Go Previous to display a matching record, or click Close to close the Form dialog box. See Searching for Records for more information on using these buttons.

See Also
Using Database Forms

Using External Databases

You can use linking with the Data|Query options to access information in a database file created with Paradox, Reflex, or dBASE (II, III, and IV). You can extract the information from the database without having to translate the file first. The procedure for this is basically the same as querying the current notebook:

1. Set up a criteria table and an output block in a Quattro Pro notebook. Use the exact field names as those in the database file.
2. Choose Data|Query and specify the database file itself as the database block using linking syntax, for example, [Database]A1..A2. [Database] is the name of the database, including path; for example, [C:\PARADOX\SAMPLE.DB]. The block specified after the file name can be any valid block with at least two rows. A1..A2 acts as a hook into the database. It remains A1..A2, no matter how large the database block actually is.
3. Specify the criteria table and output block.
4. Choose Extract. Quattro Pro copies the specified information from the outside database file and writes it into the output block.

To view all the records in an external database, open it with File|Open, or access it directly with the Data|Database Desktop or Data|Table Query command. To see the Database Desktop help file, click Database Desktop Help.

See Also

Setting Up a Database

Searching for Records

Sorting Data

Sorting reorders a block, called the sort block, based on sort key columns within it.

The quickest way to sort a block is to use the SpeedSort button on the SpeedBar.

Or, use Data|Sort:

1. Choose Data|Sort.
2. Specify the block, the cell block containing all the fields of the records you want to sort. (Don't include field names.)
3. Specify one to five key columns, single-column blocks to be sorted in the order indicated. 1st is the primary sort key, 2nd is the secondary key, and so on. If two entries in the first key column are identical, Quattro Pro looks at the second key, and so on.
4. Choose Ascending to sort a key in ascending order (from smallest to largest). Or, choose Descending to sort in the reverse order.
5. Reset the Data and Labels buttons if you need to change the sort order.
6. When you're ready to perform the sort, choose OK.

See Also

Sort Dialog Box

Sort Keys

Sort Order

Sorting Tips

Sort Keys

Quattro Pro uses specified columns in the sort block to reorder data. These sort keys determine which fields the sort is based on. You may specify up to five sort keys. Sorting begins with the first key. If two entries in in that field are identical, Quattro Pro looks at the second key. If the first two fields are identical, the third key determines sort order, and so on. For example, if LAST_NAME is the first and only sort key, the following list sorts as shown:

Before Sorting:

FIRST_NAME	LAST_NAME
Sam	Wong
John	Fernandez
Herman	Schwartz
Maria	Fernandez
Lawrence	Jones
Alicia	Fernandez

After Sorting:

FIRST_NAME	LAST_NAME
John	Fernandez
Maria	Fernandez
Alicia	Fernandez
Lawrence	Jones
Herman	Schwartz
Sam	Wong

If FIRST_NAME is added as the second sort key, the Fernandez records sort in this order (if Ascending is checked): Alicia Fernandez, John Fernandez, Maria Fernandez.

See Also

Sort Dialog Box
Sorting Data
Sort Order
Sorting Tips
SpeedSort Button

Sort Order

Each sort key you specify can be sorted in ascending or descending order. When you check Ascending, Quattro Pro sorts data in the following order:

- blank cells
- labels beginning with numbers (in numerical order)
- labels beginning with letters and special characters (in ASCII order)
- values (in numerical order)

When you uncheck Ascending, Quattro Pro sorts records in descending order.

The Data options (Numbers First, Labels First) specify how to sort columns containing values and labels. If Data is set to Numbers First, the values in a column are sorted and placed as a group before the labels in the column (which are also sorted). If the sort order is descending, the order of the two groups is reversed (values appear after labels). If Data is set to Labels First, the values in a column are sorted and placed as a group after the labels in the column (which are also sorted). If the sort order is descending, the order of the two groups is reversed (labels appear after values).

The Labels options (Character Code, Dictionary) specify how labels sort. Character Code, the default, sorts labels alphabetically according to the character code of their first letters; labels with special characters fall at the end; and uppercase letters sort before lowercase letters. Dictionary disregards case and looks at each letter.

One of the Application properties, International|Language, loads alternate sort tables with different sort orders.

See Also

Sort Dialog Box

Sorting Data

Sort Keys

Sorting Tips

SpeedSort Button

Sorting Tips

Dates are sorted by their date serial numbers; formulas are sorted by their results.

If a sort block contains formulas that reference cells OUTSIDE the sort block (for example, +A2), make them absolute (+$A$2) before sorting.

Pad numbers stored as labels in the sort block with leading zeros or spaces. This ensures they sort in numerical order. For example, `'5` and `'100` won't sort properly unless they are entered as `'005` and `'100`.

Formulas that refer to cells outside of their own row are not updated when records are resorted. If you have such formulas in your records, use Edit|Values to convert them to their end values before sorting them, or edit the formulas to correct the cell references after sorting.

To restore the sort block to its original order after sorting, insert an extra column in the sort block BEFORE you sort it. Use Block|Fill to fill this column with consecutive numbers. Then you can specify this column as the first sort key whenever you want to restore the original order.

If you do not specify ALL the cells of each record you sort, the resulting sorted fields will be mismatched. You will then have to retrieve your file again to restore it, or re-enter the records.

To reverse the effects of a sort operation, use Edit|Undo.

Records you enter after sorting are not entered in sort order. Sort again to include new records where they belong.

See Also

Sort Dialog Box
Sorting Data
Sort Order
Sort Keys
Edit Menu

SpeedSort Button

The quickest way to sort a block is to use the SpeedSort button on the SpeedBar:

1. Select the block to sort.
2. Hold the Shift+Ctrl keys and click the column to sort by. If you want a second sort key, click that column next, and so on.

   For example, if each record contained a last name, a first name, a middle initial, and an employee ID, you would click the first column to sort by last name.
3. Click the SpeedSort button on the SpeedBar to perform the sort and save the sort settings. The top part of the button sorts in ascending order; the bottom sorts in descending order.

To re-sort the same block, click SpeedSort again. The current settings remain until you select a different block and click SpeedSort.

See Also

Sort Dialog Box

Sorting Data

Sort Keys

Sort Order

Sort Tips

Limiting Data Entry

Data|Restrict Input limits selector movement to unprotected cells in a specific contiguous block. This is handy when designing data entry forms in macros. All data is visible, but not accessible, and you can't use the menus.

To use Restrict Input,

1. Protect the notebook page (choose Protection|Enable in the page Object Inspector).
2. Unprotect the cells in the block where you want to allow data entry and edits (choose Protection|Unprotect in the Block property list).
3. Select the block, choose Data|Restrict Input, then choose OK. INPUT appears on the status line.

To return to Ready mode and restore full notebook access, press Esc or press Enter with no data on the input line.

You can also limit the type of data that's entered into cells. Choose Data Entry Input in the Block property list, then indicate the type of data for the selected block: General, numbers and text; Labels Only, text but not numbers; or Dates Only.

See Also

Data Menu

Page Properties

Block Properties

Building Dialog Boxes and SpeedBars

Dialog Box and SpeedBar Overview

A dialog box is a box of options that you create to prompt the application user for information. It contains various controls, such as check boxes and edit fields.

A SpeedBar is a custom tool you create that remains onscreen while an application runs (just like the Quattro Pro SpeedBar does). It gives the user shortcuts for commonly used commands and procedures.

Dialog boxes are saved with the notebook; SpeedBars are saved as separate files.

From a user's point of view, the main difference between dialog boxes and SpeedBars is that dialog boxes appear at specific times during an application; SpeedBars usually remain onscreen while the application runs.

Because of this difference, you'll tend to put commonly used commands or actions on a custom SpeedBar, and reserve dialog boxes for those places in an application where you need specific information from a user.

To create dialog boxes, use Tools|UI Builder. You can create SpeedBars with Tools|SpeedBar Designer or Tools|UI Builder.

See Also

Basic Dialog Box Procedure

Basic SpeedBar Procedure

Building Custom SpeedBars

Dialog Window Objects

Tools for Building Dialog Boxes and SpeedBars

Dialog boxes and SpeedBars are similar in many ways; you use similar commands and tools to build them. When you choose Tools|UI Builder, a dialog window opens that lets you build dialog boxes and SpeedBars for a custom application. The dialog window SpeedBar includes tools to help you create each dialog box control.

Dialog Window

Dialog Window SpeedBar

See Also

Creating a Dialog Box

Creating a SpeedBar

Dialog Window Objects

Dialog Window

When you choose Tools|UI Builder, you see an empty dialog box within a dialog window. New dialog boxes contain an OK button and a Cancel button, to provide a consistent way for the user to close the dialog box. Also, a default name is automatically assigned.

When you're creating a dialog box in a dialog window, it looks slightly different from how it will appear to the user in a completed application.

The dialog window is the dialog box in an editable form; it contains all the Windows user interface elements for a window: a window border, a Control-menu box, and Minimize and Maximize buttons.

In the dialog window, the name of the dialog box appears in the title bar. You'll use this name in a macro to display the dialog box or change its property settings.

This name is different from the title of the dialog box, which is what the user sees when the application runs.

SpeedBars don't have titles, but they do have names (which, like dialog box names, appear in the title bar)

See Also

Tools for Building Dialog Boxes and SpeedBars

Active Dialog Window Properties

Dialog Window Objects

Dialog Window SpeedBar

The dialog window SpeedBar contains tools that are specifically designed to help you build dialog boxes and SpeedBars.

When you move the mouse pointer over a tool in the dialog window SpeedBar, the status line displays the name of the tool. To see an explanation of a tool, point to a tool and Ctrl+right-click to display Object Help.

These tools let you quickly create, copy, move and test controls. Controls are elements you put in a dialog box to gather information from the user or to perform a desired action.

The Cut, Copy, and Paste buttons are shortcuts for Edit|Cut, Edit|Copy, and Edit|Paste, respectively. You can copy a control to the Clipboard to duplicate it or to move it to another dialog box or SpeedBar.

The Test button tests the operation of a dialog box (or SpeedBar). In test mode, the dialog box controls behave exactly as they will when the application runs.

The Selection tool lets you select a control in the dialog window to move it, resize it, or change its properties, etc. You can select several controls at a time, to move or size them together. (You can use Edit|Select All to select all controls in a dialog window.)

The remaining tools on the SpeedBar let you add controls to a dialog box; they are described individually under Dialog Window Objects topics.

See Also

Tools for Building Dialog Boxes and SpeedBars

Dialog Window

Creating a Dialog Box

Creating a SpeedBar

Basic Dialog Box Procedure

Here are the basic steps to follow to create a dialog box:

1. Choose Tools|UI Builder. A dialog window appears. (See Opening a New Dialog Box.)
2. Enter a name for the dialog box (later, you'll use this name in a macro to make the dialog box appear).
3. Enter a title for the dialog box (the title is what the user will see).
4. Add controls to the dialog box. (See Adding Dialog Controls.)
5. Customize each control, if necessary. (See Customizing Controls.)
6. Resize the dialog box, if desired.
7. In the notebook, create a {DODIALOG} command to call the dialog box and pass initial defaults to it. (See Using the {DODIALOG} Command.)
8. Add a link command to each control, if necessary. A link command indicates what should happen when a user manipulates that control in a specific manner. (See Linking Commands to Dialog Controls.)
9. Test the operation of each control. (See Testing a Dialog Box.)
10. Save the dialog box (by saving the notebook it is in).

See Also

Creating a Dialog Box

Tools for Building Dialog Boxes and SpeedBars

Dialog Window Objects

Basic SpeedBar Procedure

Here are the basic steps to follow to create a SpeedBar with Tools|UI Builder:

1. In the notebook, select the Graphs page (click the SpeedTab button).
2. Choose the New SpeedBar tool from the Graphs page SpeedBar. A long, narrow dialog window appears--this is the blank SpeedBar. (See Opening a New SpeedBar.)
3. Add controls to the SpeedBar. (See Adding Controls to the SpeedBar.)
4. Customize each control as necessary.
5. Add a link command to each control, to indicate what should happen when the user clicks this control. (See Linking Commands to SpeedBar Controls.)
6. Test each control to make sure that it works properly. (See Using Test Mode.)
7. Save the SpeedBar. (Since SpeedBars are saved as separate files, you can use a SpeedBar with different notebooks.)
8. Display the SpeedBar using the SpeedBar Control menu. (See Displaying the SpeedBar.)

Note: You can also build SpeedBars using Tools|SpeedBar Designer; see Building Custom SpeedBars.

See Also

Creating a SpeedBar

Tools for Building Dialog Boxes and SpeedBars

Dialog Window Objects

Creating a Dialog Box

The following topics describe how to create a dialog box. The easiest way to move from topic to topic is to click the Help **>>** button above.

Opening a New Dialog Box

Adding Dialog Controls

Labeling Controls

Customizing Controls

Using the {DODIALOG} Command

Linking Commands to Dialog Controls

Testing a Dialog Box

See Also

Creating a SpeedBar

Dialog Window Objects

Opening a New Dialog Window

The first stage in building a dialog box is opening a blank dialog box and giving it a name and a title.

There are two ways to reach the dialog window. Typically, you'll choose Tools|UI Builder from within a spreadsheet page (you could click the dialog icon from the Graphs page instead).

1. Starting from a new notebook, choose Tools|UI Builder. You'll see a new dialog window.
2. To give this new dialog box a name, right-click the dialog box background and choose Dialog Properties, then choose Name. Type a name and choose OK. Notice that the title bar of the dialog box shows its new name.
3. Next, create the dialog box title. Right-click the dialog box background and choose Dialog Properties, then choose Title. Type the title and choose OK. (You won't see any visible change here; the Title is what a user sees.)
4. Dialog boxes are stored within their notebook. To save this dialog box, save its notebook--click anywhere in the notebook, choose File|Save As, then enter a file name.

There is an alternative way to create a dialog box: From the notebook, choose the Graphs page, then select the New Dialog tool from the Graphs page SpeedBar.

See Also

Creating a Dialog Box

Adding Dialog Controls

Dialog Window Objects

Adding Dialog Controls

Once you've named a dialog box and given it a title, the next stage is to add the desired controls to it.

Use the dialog window SpeedBar to add controls to a dialog box. When you move the mouse pointer over a tool in the dialog window SpeedBar, the status line displays the name of the tool. To see an explanation of a tool in the dialog window SpeedBar, Ctrl+right-click to display Object Help for a tool.

Quattro Pro offers a variety of different controls; each control is best suited for presenting (or receiving) a certain type of information. Each control displays a setting, and provides a way for users to change that setting.

Following is a brief description of the most commonly used controls (see Dialog Window Objects for full descriptions of controls and their properties).

Push Button	A push button performs a specific action when the user clicks it. The developer determines what that action is.
Check Box	Check boxes present a Yes/No choice to the user. The user checks the check box to accept the choice. Check boxes often are placed within group boxes, to present the user with mutually exclusive choices.
Radio Button	Radio buttons (also called option buttons) are usually grouped, to give the user a mutually-exclusive list. When a user clicks a radio button, its diamond darkens to indicate that it's chosen.
Label	A label clarifies to the user what a specific control does.
Edit Field	An edit field is where a user types specific information.
Spin Control	A spin control lets the user click an arrow to increase or decrease the value it displays (the user can also type a value instead).
Rectangle	A rectangle embellishes a dialog box or groups two or more controls together.
Group Box	A group box usually contains other controls such as radio buttons or check boxes. It looks like a rectangle with a title at the top.

To add controls to a dialog box, follow these steps:

1. To go to the dialog window, select the Graphs page and double-click the dialog icon. If you don't have a dialog window open, choose Tools|UI Builder.
2. Choose a tool from the dialog window SpeedBar.
3. Position the pointer near the upper left corner of the dialog box (leave room for a label on the left). Click to drop the control into place. The control appears, with its top left corner at the position you clicked.
4. To make the control bigger, drag one of the control's handles.

Note: You can create a control and size it all in one step: Instead of clicking the mouse button, move to the position where you want the control to start, then drag the pointer until the control is the desired size.

Tip: There is a fast way to fill a group box with radio buttons--just hold down the Ctrl key while you resize the group box.

See Also

Labeling Controls

Creating a Dialog Box

Labeling Controls

Labels help explain to the user what each control in a dialog box does. To label a control, follow these steps:

1. Choose the Label tool from the dialog window SpeedBar.
2. Move the pointer just to the left of a control, then click. (Move the edit field to the right if you need more room).
3. Next, add text to the label: Double-click the label, type the label text, and press Enter.

See Also

Customizing Controls

Creating a Dialog Box

Dialog Window Objects

Customizing Controls

After you've added controls to a dialog box, right-click on the controls and use the Properties command to customize them. (For full descriptions of controls and their properties, see Dialog Window Objects.)

The following topics provide examples of how to customize controls:

Customizing an Edit Field

Customizing a Spin Control

Customizing Radio Buttons and Group Boxes

When you're finished customizing the controls, save your work. Click anywhere in the Graphs page, then choose File|Save. Your dialog box is saved with the notebook.

See Also

Using the {DODIALOG} Command

Creating a Dialog Box

Dialog Window Objects

Customizing an Edit Field

This example shows how to customize an edit field and give it a range of 1000-30000:

1. Right-click an edit field in a dialog box, then choose the Properties command. You'll see its Object Inspector.
2. Choose Field Type, then choose Integer.
3. Right-click the edit field again and choose the Properties command. Choose Minimum, type 1000, and choose OK.
4. Right-click the edit field again and choose the Properties command. Choose Maximum, type 30000, and choose OK.

The edit field will not accept text strings or alphabetical characters, or values above 30000 or below 1000.

See Also

Customizing Controls

Using the {DODIALOG} Command

Creating a Dialog Box

Dialog Window Objects

Customizing a Spin Control

This example shows how to limit a spin control to the range 1-5:

1. Right-click a spin control in a dialog box, then choose the Properties command. Choose Minimum, type 1 and choose OK.
2. Right-click the spin control button again, then choose the Properties command. Choose Maximum, type 5 and choose OK.

See Also

Customizing Controls

Using the {DODIALOG} Command

Creating a Dialog Box

Dialog Window Objects

Customizing Radio Buttons and Group Boxes

This example shows how to add descriptive text to radio buttons and group boxes:

1. Double-click a radio button, type a description of the option, and press Enter.
2. Repeat this process for other radio buttons.
3. To give a group box a title, double-click the group box title, type the title, and press Enter.

To size the dialog box, grab a lower left corner of the dialog box and drag it upward, then release the mouse button.

See Also

Customizing Controls
Using the {DODIALOG} Command
Creating a Dialog Box
Dialog Window Objects

Using the {DODIALOG} Command

After you're finished creating a dialog box and customizing the controls in it, the next step is to create a macro that displays the dialog box and stores the user's choices in cells in the notebook. The macro sends initial values to each control in the dialog box. If the user changes a setting in the dialog box, that change is sent back to a cell in the notebook when the user closes the dialog box.

First, set up an area in the notebook that will send default settings to the dialog box. For example, in the following table for a loan payment dialog box, B1..B5 is a block that defines the initial dialog box settings:

	A	B
1	Pay_Every	Month
2	Loan_Amount	100000
3	Int_Rate	12.5
4	Term	30
5	End_Of_Period	No

This sets the default payment period to month, the default loan amount at 100,000, the default interest rate at 12.5%, the default term at 30 years, and specifies payment at the start of the month.

After the initial dialog box settings have been entered in the notebook, enter a macro command that will display the dialog box and send the user's choices back to the notebook. Use the {DODIALOG} command to do this. The {DODIALOG} command has several arguments, which are noted as follows:

The first argument is the name of the notebook that contains the dialog box you want to display, and the name of the dialog box. The name of the notebook is optional if the dialog box is stored in the active notebook.

The second argument specifies a cell that will store a value indicating how the user closed the dialog box. If the dialog box was canceled, 0 is stored in the cell; if the dialog box was closed by Enter or OK, 1 is stored.

The third argument specifies a block that contains initial values for the dialog box controls and receives final values from the dialog box controls when the user closes the dialog box.

The fourth argument (1 or 0) specifies whether the user should manipulate the dialog box (1) or if the macro should manipulate it (0); the default is 1.

Each cell in the block (starting at the upper left cell and proceeding row-wise to the lower right cell) sets the initial value of one control. If the user cancels the dialog box, the values in this block remain unchanged.

Here is an example of a {DODIALOG} macro used to display a loan payment dialog box using the initial settings in B1..B5:

```
{DODIALOG "LoanData",B6,B1..B5,1}
```

This command displays the dialog box and passes the initial values in cells B1 through B5 to the dialog box controls. It also sends a value of 0 to cell B6 if the user cancels the dialog box; otherwise, 1. The final argument of the command specifies that the user can manipulate the dialog box.

To make this dialog box dynamic, so that the user's changes are immediately reflected in the

notebook, add link commands to each control.

See Also

Linking Commands to Dialog Controls
Creating a Dialog Box
Dialog Window Objects

Linking Commands to Dialog Controls

Link commands are statements you attach to each control, to indicate what should happen when the user changes the control.

The {DODIALOG} command brings changed values back to the notebook when the user closes the dialog box. However, if you set up a dynamic link you don't have to close the dialog box to see the changed values. You accomplish this with link commands.

For example, you might want to add a link command to an edit field to make it write changes back to cell B2 immediately. To add the link commmand to an edit field, follow these steps:

1. Double-click the edit field.
2. Choose Dialog|Links. The Object Link dialog box appears.
3. Choose Add.

Basically, you enter link commands as statements, specifying "when a happens, do b to c." The statement reads something like a "what, when, where" statement:

"When the user changes a setting, send the new value back to a specific cell in the notebook."

For a list of link events in the Object Link dialog box, see Link Events. For complete information about link commands, see Working with Link Commands.

For an example showing how to add link commands to dialog box controls, see Adding Link Commands to an Edit Field.

By the way, there's a fast way to create a dynamic link; use Dialog|Connect.

After you add link commands, test each control to see if the dynamic links work.

See Also

Testing a Dialog Box

Creating a Dialog Box

Dialog Window Objects

Adding Link Commands to an Edit Field

Here is an example showing how to add link commands to an edit field:

1. First, specify what event triggers the command (a "when" statement): Move to the first button, which is labeled Init. This is the Event pick list button.

   Hold down the left mouse button, drag to Valuechanged, then release. (You want a value sent back to the notebook when it has been changed.)

2. Next, specify the action that should occur (a "what" statement): Move to the second button, labeled RECEIVE. Hold down the mouse button, drag to SEND, then release. (You want to send the change back to the notebook).

   The third button reads Value; since this is correct, leave it as is (you want to send the new value back to the notebook).

   Move to the fourth button, beside TO. This is the Object pick list button, where you specify the object to be acted upon. Hold down the mouse button, and choose <POINT>.

   If you're still on the Graphs page, click the SpeedTab button to return to the notebook.

   Move the Object Link dialog box out of the way if you need to.

3. In the notebook, click cell B2 (where you want the value sent). You return to the Link dialog box.

   Finally, hold down the last button, drag to Value, then release. This indicates what you want done to the cell--set its value.

4. Choose OK.

See Also

Testing a Dialog Box

Creating a Dialog Box

Dialog Window Objects

Testing a Dialog Box

To test the link commands for a dialog box:

1. Choose the Test button from the dialog window SpeedBar (or choose Test from the dialog window's Control menu). The dialog box will look just as the user will see it (except TEST MODE appears next to the dialog box title).

   In Test mode, you can't activate any other windows unless you've configured a control in the dialog box to do so.
2. Move the dialog box to the left or right, so that you can see the cells in the notebook.
3. Change the settings of controls in the dialog box and see if the notebook cells change simultaneously. For example, if the dialog box has an edit field, enter a value in the field and check to see that the linked notebook cell has changed.
4. To end Test mode, choose OK.
5. To save your work, click any cell in the notebook and choose File|Save.

See Also

Creating a Dialog Box

Dialog Window Objects

Creating a SpeedBar

SpeedBars aren't associated with a particular notebook (no icon is created on the Graphs page), so you can build a SpeedBar from any notebook.

Read each of the following topics in turn to learn how to create a SpeedBar. The easiest way to move from topic to topic is to click the Help **>>** button above.

Opening a New SpeedBar

Adding Controls to the SpeedBar

Linking Commands to SpeedBar Controls

Displaying the SpeedBar

Editing SpeedBars

Note: You can also build SpeedBars using Tools|SpeedBar Designer; see Building Custom SpeedBars.

See Also

Creating a Dialog Box

Dialog Window Objects

Opening a New SpeedBar

To open a new SpeedBar, first, open a blank dialog window:

1. Select the Graphs page.
2. Click the New SpeedBar tool on the Graphs page SpeedBar. A dialog window appears; this is the SpeedBar in an editable form.

   Unlike dialog boxes, SpeedBars don't contain an OK and Cancel button.
3. Name and save this new SpeedBar (SpeedBars are saved as external files with the extension .BAR): Choose Dialog|Save SpeedBar As, type the file name, then choose OK. This name appears in the dialog window's title bar.

If you're making a SpeedBar for a specific notebook, consider giving the SpeedBar the same name as the notebook, adding the file extension .BAR. This helps you keep track of which SpeedBar belongs with which application. Next, add controls to the SpeedBar.

See Also

Adding Controls to the SpeedBar

Creating a SpeedBar

Dialog Window Objects

Adding Controls to the SpeedBar

To add controls to a SpeedBar, choose a tool on the dialog window SpeedBar and then click on the SpeedBar.

To see an explanation of tools in the dialog window SpeedBar, Ctrl+right-click to display Object Help for a tool.

Here is an example showing how to add two buttons to a new SpeedBar.

1. Choose the Push Button tool on the dialog window SpeedBar. Then click near the left end of the SpeedBar.
2. Double-click this push button and enter `Save` as the button title.
3. To create a bitmap button beside the Save button, choose the Bitmap Button tool on the dialog window SpeedBar, then click to the right of the Save button in the SpeedBar.
4. To replace the OK image on this button's face with a small box and grid, right-click it and choose the Properties command, then choose Bitmap, choose `ldrawout` from the list box and choose OK. The button is a little too large to fit in the SpeedBar, so drag one of its top handles down to shrink it, then move the button up.
5. Click the SpeedBar's background to deselect the bitmap button. Then hold down the Shift key, click the Save button, and click the bitmap button. Both buttons are now selected.
6. To align these two buttons, choose Dialog|Align.

   This menu offers several commands to help you visually align controls in dialog boxes and SpeedBars.
7. Choose Horizontal Space, type `10` and choose OK. The buttons should now be evenly spaced. Choose Dialog|Align|Top to line up the tops of the controls.
8. Click the SpeedBar's background to deselect the objects.

Next, add link commands to the SpeedBar controls.

See Also

Linking Commands to SpeedBar Controls

Creating a SpeedBar

Dialog Window Objects

Linking Commands to SpeedBar Controls

After adding controls to a SpeedBar, you can add links to the controls. Link commands are statements that indicate what happens when a user changes or selects a control.

To add a link command to a control, follow these steps:

1. Double-click the control.
2. Choose Dialog|Links. The Object Link dialog box appears.
3. Choose Add.

For a list of link events in the Object Link dialog box, see Link Events. For complete information about link commands, see Working with Link Commands.

For example, consider a new SpeedBar that has two buttons, a Save button and a bitmap button. The following examples show how you can make the Save button save the active file and the bitmap button draw a box around the active block.

Linking a Save Button to the Save Command

Linking a Button to a Command That Draws a Box

See Also

Displaying the SpeedBar

Creating a SpeedBar

Dialog Window Objects

Linking a Save Button to the Save Command

1. Select the Save button and choose Dialog|Links. Then choose Add to create a link command.
2. First, indicate which event should initiate an action--in this case, a mouse click by the user.

   Pull down the event pick list (which reads Init) and choose Clicked.
3. Next, specify what will happen when the user clicks the button: a macro should run.

   Pull down the action pick list (which currently reads RECEIVE) and choose DOMACRO.
4. Then enter the macro commands that should run when the button is clicked. The remaining pick list buttons change to an edit field, where you enter the macro command you want this button to perform.

   Type `{FileSave Backup}` in the edit field. This macro will save the file and create a backup of the previous version.
5. Choose OK to save the link command and return to the dialog window.

See Also

Linking Commands to SpeedBar Controls
Working with Link Commands
Displaying the SpeedBar
Creating a SpeedBar
Dialog Window Objects

Linking a Button to a Command That Draws a Box

1. Select the bitmap button, choose Dialog|Links, then choose Add.
2. To specify that a mouse click should initiate the link command, set Event to Clicked.
3. To indicate the action that should happen when the user clicks the button, pull down the action pick list (RECEIVE appears on it) and choose SET (we'll set the linedraw property of a block).
4. In the edit field type:

   ```
   Thin,Thin,Thin,Thin,NoChange,NoChange
   ```

5. Specify that the link command will change a property in the active block when the button is clicked: Pull down the object pick list (<POINT> appears on it) and choose Active_Block.
6. To indicate that this setting should go to the active block's Line Drawing property, pull down the last pick list and choose Line_Drawing.
7. Choose OK to save the link and return to the SpeedBar.

Next, save the SpeedBar and display it.

See Also

Linking Commands to SpeedBar Controls

Working with Link Commands

Displaying the SpeedBar

Creating a SpeedBar

Dialog Window Objects

Displaying the SpeedBar

After you create a SpeedBar, save it and use the SpeedBar Control menu to add it to the Quattro Pro window:

1. Choose Dialog|Save SpeedBar.
2. Close the SpeedBar and activate a notebook window.
3. To display the SpeedBar Control menu, right-click the standard SpeedBar any place *outside* a button or field.
4. Choose Append to add the SpeedBar below the active bar or Insert to add the SpeedBar above the active bar.

To see how this SpeedBar works, select a block in the active notebook, then click the bitmap button. A box surrounds the block.

Note: To remove the SpeedBar, click the Close SpeedBar button. (You can also remove a SpeedBar using the SpeedBar Control menu: right-click the SpeedBar any place outside a button or field, then choose Remove.)

The application Object Inspector (right-click the Quattro Pro title bar or choose Property|Application) has a property that controls the display of the standard SpeedBar. Use the Show SpeedBar option of the Display property to specify whether the standard Quattro Pro SpeedBar appears. To hide it, uncheck the Show SpeedBar check box.

You can also display a SpeedBar using the {SpeedBar.*Option*} macro command.

To hide the standard SpeedBar, use the following command:

```
{Application.Display.Show_Toolbar "No"}
```

See Also

Editing SpeedBars

Creating a SpeedBar

Dialog Window Objects

Editing SpeedBars

To edit a SpeedBar,

1. Choose Tools|UI Builder.
2. Choose Dialog|Open SpeedBar.
3. After making any changes, choose Dialog|Save SpeedBar.

Or, go to the Graphs page, click the New SpeedBar tool, then choose Dialog|Open SpeedBar, enter the name of the SpeedBar, and choose OK.

See Also

Using Test Mode

Creating a SpeedBar

Dialog Window Objects

Using Test Mode

When a SpeedBar is in Test mode, you can activate other windows to test the SpeedBar's operation. Also, in Test mode SpeedBars aren't locked to the top of the Quattro Pro window. You can use Test mode to create "floating" SpeedBars. Unlike dialog boxes, multiple SpeedBars can be in Test mode at the same time.

If you need to exit Test mode without running any link commands, choose Stop from the dialog box's Control menu. Clicking an OK or Cancel button also exits Test mode; if you've added link commands to the dialog box that run when OK or Cancel is clicked, those link commands will run.

Pressing Esc is the same as clicking Cancel.

See Also

Creating a Dialog Box

Creating a SpeedBar

Dialog Window Objects

Working with Dialog Boxes

You can create, rename, or copy dialog boxes on the Graphs page. Work with icons on the Graphs page to perform operations listed in this table:

Operation	Step(s)
Edit	Double-click the dialog box icon. Right-clicking the icon and choosing the Properties command lets you set various dialog box properties.
Rename	Right-click the dialog box icon, choose the Properties command, then choose Name. Enter the new name.
Delete	Click the dialog box icon and choose Edit|Clear (or press Del).
Copy to another	Click the dialog box icon, choose Edit|Copy, select the destination notebook's Graphs page, and choose Edit|Paste. If both the source and the destination notebook appear on the Quattro Pro desktop, you can copy the dialog box by selecting the source notebook's Graphs page and dragging the dialog box icon into the destination notebook window.
Duplicate	Click the dialog box icon, choose Edit|Copy, then choose Edit|Paste. A duplicate can't have the same name as the original, so a unique name is automatically generated.
Move to another	Select the dialog box icon, choose Edit|Cut notebook to select the destination notebook's Graphs page, then choose Edit|Paste.

See Also

Creating a Dialog Box

Dialog Window Objects

Working with Controls

The information in this section applies, generally, to all controls used in dialog boxes. Most of these topics also apply to SpeedBar controls.

Selecting Several Controls
Copying and Moving Controls
Moving and Sizing Controls
Using the Dimension Options
Moving Overlapping Controls
Aligning Controls
Repositioning Controls
Setting a Control's Value
Changing a Control's Text
Using the Clipboard with Controls
Assigning a Key to a Control
Setting Tab Order
Group Box Selection Order
Disabling and Hiding Controls
Grouping Controls
Connecting Controls to Cells or Other Controls
Providing Hints
Creating Object Help

See Also

Creating a Dialog Box
Creating a SpeedBar
Dialog Window Objects

Selecting Several Controls

You can select more than one control at a time to move them as a group:

1. Choose the Selection tool from the dialog window SpeedBar.
2. Hold down Shift and click each control you want to select. Handles surround each control as you select it.

To deselect a control, click it again.

Another way to select multiple controls is to choose the Selection tool and drag a selection frame around the controls. You can also use Edit|Select All to select all controls in a dialog window.

To deselect a group of controls, choose the Selection tool and click the dialog window background.

See Also

Dialog Window Objects

Working with Controls

Copying and Moving Controls

To move a control to a different dialog box (or SpeedBar), select the control, choose Edit|Cut to move it to the Clipboard, then choose Edit|Paste to copy it into another dialog window in another notebook. (The target notebook and dialog box must be open and active.) Each time you choose Edit|Paste a new copy of the control is created.

Caution: Deselect any controls in the active dialog window before using Edit|Paste, so the selected control is not changed (see Using the Clipboard with Controls for details).

To copy a control to the Clipboard without removing it from the active dialog window, select it and choose Edit|Copy.

Copying Parent Controls

Copying a parent control to the Clipboard copies any controls attached to it as well; removing the parent control from the dialog window also removes the attached controls. See Grouping Controls for details.

See Also

Dialog Window Objects

Working with Controls

Moving and Sizing Controls

To resize or move a control,

1. Choose the Selection tool.
2. Select the control by clicking it.
3. To move a control, point to it and drag it. To resize a control, drag one of its handles.

If multiple controls are selected, dragging one control moves all the controls; dragging the handle of one control also resizes the other selected controls.

Selecting multiple controls (by holding down Shift while clicking them) and choosing Dialog|Align|Resize to Same resizes the selected controls to the same size as the first control selected.

See Also

Using the Dimension Options

Moving Overlapping Controls

Dialog Window Objects

Working with Controls

Using the Dimension Options

To specify a precise size and position for a control, right-click the control and choose the Properties command, then choose Dimension. The next table lists Dimension options:

Option	Description
X Pos	Distance in pixels between the left edge of the control and the left side of the dialog box.
Y Pos	Distance in pixels between the top edge of the control and the bottom edge of the dialog box's title bar.
Width	Width of the control, in pixels.
Height	Height of the control, in pixels.

See Also

Moving and Sizing Controls

Moving Overlapping Controls

Dialog Window Objects

Dimension Property

Working with Controls

Moving Overlapping Controls

The next table lists commands on the Dialog|Order menu that let you display a control that's beneath another control.

Command	Description
Bring Forward	Brings the selected control one step forward.
Send Backward	Sends the selected control one step backward.
Bring to Front	Places the selected control over any controls that covered it previously.
Send to Back	Places the selected control beneath any controls it covered previously.

See Also

Moving and Sizing Controls
Using the Dimension Options
Dialog Window Objects
Working with Controls

Aligning Controls

To align controls, right-click the dialog window background and choose the Properties command, then choose Grid Options. There are three settings that help align controls.

Setting	Description
Grid Size	Lets you specify the distance in pixels between grid points.
Show Grid	Makes the grid visible. Dots appearing in the background represent each grid point. (Users won't see this grid.)
Snap to Grid	Enables a hidden grid. Controls will align with the grid as you move them.

See Also

Aligning Multiple Controls

Dialog Window Objects

Working with Controls

Aligning Multiple Controls

You can use commands on the Dialog|Align menu to align or reorder multiple controls.

Command	Description
Left	Aligns controls along the left edge of the leftmost object selected.
Right	Aligns controls along the right edge of the rightmost object selected.
Horizontal Center	Centers controls between the top and bottom of the dialog box.
Top	Aligns controls along the top edge of the topmost object selected.
Bottom	Aligns controls along the bottom edge of the bottommost object selected.
Vertical Center	Centers controls between the left and right sides of the dialog box.
Horizontal Space	Horizontally spaces controls by the amount of pixels you specify (the vertical position of the controls remains the same). It also reorders the controls based on the order in which you selected them (see Repositioning Controls for details).
Vertical Space	Vertically spaces controls by the amount of pixels you specify (the horizontal position of the controls remain the same). It also reorders the controls based on the order in which you selected them (see Repositioning Controls for details).
Resize to Same	Sets all selected controls to the same size as the first control selected.

See Also

Dialog Window Objects

Working with Controls

Repositioning Controls

The commands Horizontal Space and Vertical Space let you evenly space and arrange controls in a dialog box.

The following example shows how to reposition controls using the Horizontal Space command.

1. Open a dialog box and use Shift+click to select controls to arrange. The first control you select appears at the left after repositioning, the next appears to the right of it, and so on.
2. Choose Dialog|Align|Horizontal Space.
3. Set their new spacing, in pixels. The default is 5.
4. Click OK.

Vertical Space behaves like Horizontal Space, but the first control you select moves to the top, the second control moves just below the first, and so on.

See Also

Dialog Window Objects

Working with Controls

Setting a Control's Value

Each control stores one setting; the user manipulates the control (by clicking a scroll arrow, typing text, and so on) to change that setting.

There are several ways to set a control's value:

- through initial settings you pass using a {DODIALOG} command
- by a user changing a setting from within the dialog box
- through the {SETOBJECTPROPERTY} command
- with rough link commands from any dialog control (see Working with Link Commands for details).

For example, the following macro command uses the Value property to enter Spanish into an edit field:

```
{SETOBJECTPROPERTY "Dialog1:EditField3.Value","Spanish"}
```

See the following topics for details on how to set a control's value:

Setting a Control's Initial Value

Changing the Order in Which You Pass Initial Values

Editing the Setting Order

See Also

Dialog Window Objects

Working with Controls

Setting a Control's Initial Value

The macro command {DODIALOG} displays the specified dialog box.

Here is an example of a {DODIALOG} command:

```
{DODIALOG "LoanData", Result, B1..B5}
```

This command displays the LoanData dialog box, places the results of how the user closed the dialog box in the cell named Result, and passes the values in cells B1 through B5 to the controls in the dialog box, in order.

If a control's Process Value property is set to No, it won't receive an initial value from the block specified in the {DODIALOG} command.

For example, if your dialog box has six controls, but you've set the Process Value property of the third control to No, your setting block should be only five cells long. The values in the first two cells will be passed to the first two controls. However, the value in the third cell will go to the fourth control, and the values in the fourth and fifth cells will go to the fifth and sixth controls, respectively.

Controls that don't have a Process Value property will receive no value from a {DODIALOG} command.

See Also

Setting a Control's Value

Changing the Order in Which You Pass Initial Values

Editing the Setting Order

Dialog Window Objects

Working with Controls

Changing the Order in Which You Pass Initial Values

By default, the order in which you created the controls is the order in which you must pass each control's value (through the {DODIALOG} command). In other words, the first cell in the specified block sends its value to the first control you created no matter where it is positioned within the dialog box.

For example, the last argument in the command {DODIALOG "LoanData", Result, B1..B5} specifies that cells B1 through B5 contain initial values for the controls in the dialog box. Cell B1 must contain the initial value for the first control you created, cell B2 must contain the value for the second one, and so on.

To change the order of the controls in the dialog box to match the order of the values in the setting block:

1. Click the dialog window background to deselect its controls.
2. Hold down the Shift key and, in the order in which you want to pass initial settings, click each control.
3. Choose Dialog|Order|Order Controls.

The first control you clicked will receive its initial value from the first cell of the block, the second control you click will receive its initial value from the second cell of the block, and so on, regardless of their arrangement within the dialog box.

See Also

Setting a Control's Value

Setting a Control's Initial Value

Editing the Setting Order

Dialog Window Objects

Working with Controls

Editing the Setting Order

If you have many controls in your dialog box, you can use a shortcut to re-order the controls to match the order of the values in the setting block. Use Dialog|Order|Order From to place related controls together in the setting order.

For example, suppose you have six controls in this order:

1 2 3 4 5 6

But the setting block in your notebook already has the values in this order:

1 2 5 4 3 6

The only controls that are out of order are 4 and 3; you want them to follow 2. You can re-order the controls to match this order:

1. Click the dialog window background to deselect its controls.
2. Hold down the Shift key, then click control 2, then control 3, then control 4.
3. Choose Dialog|Order|Order From.

See Also

Setting a Control's Value
Setting a Control's Initial Value
Changing the Order in Which You Pass Initial Values
Dialog Window Objects
Working with Controls

Changing a Control's Text

If a control has a Label Text property, which is set to the title appearing on the control, you can set the property by double-clicking the control, typing the new text (or editing the existing text) and pressing Enter.

See Also

Dialog Window Objects

Working with Controls

Using the Clipboard with Controls

When the Clipboard contains text from Quattro Pro or another Windows application, you can paste the text into these controls:

Control	Effect
Buttons	The text becomes the button title.
List boxes, Combo boxes, Pick Lists	The text becomes the list; each line becomes one item. The first item becomes the initial title.
Group boxes	Creates radio buttons in the group box; each line becomes the title of one radio button.
Edit fields	The text becomes its new value.

If the Clipboard contains a bitmap, you can select a bitmap button and choose Edit|Paste to paste the bitmap onto the button's face. This bitmap is saved with the dialog box.

You can also select a dialog control and choose Edit|Copy to copy control titles, lists, or bitmaps to the Clipboard to paste them into other areas of Quattro Pro and other Windows applications.

See Also

Dialog Window Objects
Working with Controls

Assigning a Key to a Control

An underlined letter in the text of a control acts as a shortcut key; it indicates that the user can press Alt and that letter to activate the control. To specify this underlined letter,

1. Double-click the control's text.
2. Move the insertion point to just before the letter to underline.
3. Type an ampersand (&). For example, B&utton sets the title to Button.
4. Press Enter.

You can also assign a shortcut letter to a control that has no title (such as an edit field or a list box):

1. Right-click the control and choose the Properties command.
2. Choose Name.
3. Move the insertion point to just before the letter to use. You can type a new name for the control if needed.
4. Type an ampersand (&). Then press Enter.

When a control doesn't have a title, you can place a shortcut key in the control's label. Just double-click the label and type an ampersand to the left of the letter you want to use as the shortcut. Press Enter to save the new label text. Use only one ampersand character in the label; if you add more than one, the first is recognized as a shortcut key designation, and the remaining ones will appear in the label as text.

You can add alternative shortcut keys to controls using link commands. See Link Events for details on the event key, which you can use to run a link command when a specific key is pressed.

See Also

Dialog Window Objects

Working with Controls

Setting Tab Order

A user can press Tab to cycle through each control in a dialog box. To disable this feature for a particular control, right-click the control, choose the Properties command, and set the Tab Stop property to No.

To specify the order in which Tab moves from control to control, hold down the Shift key and select each control in the order you want them to cycle. Then choose Dialog|Order|Order Tab Controls to save the new order.

You can pull specific controls out of the tab order and group them together using Dialog|Order|Order Tab From. For example, suppose the following numbers represented the existing tab order:

1 2 3 4 5 6

If you select items 2, 4 and 6, choosing Order Tab From would change the tab order as follows:

1 2 4 6 3 5

See Also

Dialog Window Objects

Working with Controls

Group Box Selection Order

When check box buttons are placed in a group box, the user can activate one of the check boxes and then use Tab or arrow keys to select the next check box (the user can then press Space to check the active check box button). To change the order in which the check boxes are selected,

1. Choose the Selection tool from the dialog window SpeedBar.
2. Click the dialog window's background to deselect any selected controls.
3. Hold down Shift and then click each check box, in the order in which you want the user to scroll through them.
4. Drag the check boxes until the pointer changes to a small hand. This saves the new order.

When radio buttons are placed in a group box, the user can activate one of the radio buttons and then use arrow keys to select and choose the next radio button. You can set this selection order in the same way as check boxes: Select the radio buttons in the order you want the user to scroll through them and then move them slightly to save the new order.

See Also

Dialog Window Objects

Working with Controls

Disabling and Hiding Controls

Dialog controls have a variety of properties that you can use to temporarily hide or disable them. You can use these properties in conjunction with link commands to hide unneeded controls or reveal special settings.

Properties to hide or disable dialog controls:

Property	Purpose
Grayed	Specifies whether the control can be used. When set to Yes, the control appears dimmer, to indicate it's unavailable, and is disabled.
Disabled	Specifies whether the control can be used, like Grayed, but doesn't dim the control when it's set to Yes.
Depend On	Specifies the states of Quattro Pro in which the control is enabled. (See Using the Depend On Property.)
Enabled	The opposite of Disabled (setting this property to No disables the control). It's used by link commands and doesn't appear in a control's Object Inspector.
Hidden	Setting this property to Yes hides the control when the dialog box displays. (The control still appears in the dialog window while you're editing.)
Show	The opposite of Hidden (setting this property to No hides the control). It's used by link commands and doesn't appear in the control Object Inspector.

See Also

Dialog Window Objects

Working with Controls

Using the Depend On Property

The most useful property for disabling controls is Depend On. Use it to specify the areas of Quattro Pro in which the control is disabled. For example, some controls are appropriate only when a graph window is active; to create a control that will be available only when a graph window is active,

1. Right-click the control and choose the Properties command, then choose Depend On.
2. Uncheck all check boxes except Graph to specify that this control is available only when a graph window is active.

Each check box button controls one area of Quattro Pro, as shown in the next table.

Quattro Pro areas:

Area	When its check box is checked
Desktop	Enables the control when no windows are open.
Notebook	Enables the control when a notebook window is active.
Graph	Enables the control when a graph window is active.
Dialog	Enables the control when a dialog window is active.
Input Line	Enables the control when the input line is active (when typing a new entry or editing a cell).
Graphs Page	Enables the control when a notebook window is active and the Graphs Page is selected.

If all options are checked, the dialog control is always available.

See Also

Dialog Window Objects

Working with Controls

Grouping Controls

Some dialog controls have an Attach Child property that makes controls dragged on top of them become attached. The control on top is then a child of the control beneath it, which is the parent. Grouping controls provides many advantages to the developer:

- Moving the parent control moves any attached elements.
- Copying the parent control copies any attached elements.
- Hiding the parent control hides any attached elements.
- Disabling the parent control disables any attached elements.
- Resizing the parent control resizes any attached elements proportionally.

Child controls must cover the parent control only; controls partially covering one another won't attach. Here's an example of grouping controls together:

1. Choose Tools|UI Builder to display a new dialog window.
2. Use the Rectangle tool to create a rectangle that's big enough to contain the OK button.
3. Move the OK button into the rectangle. The OK button is now attached to the rectangle.
4. Move the rectangle; the OK button stays with the rectangle.
5. Right-click the rectangle and choose the Properties command, then choose Disabled and set it to Yes.
6. Choose the Test button to test the dialog box's operation.
7. Click the OK button; nothing happens, since the rectangle (its parent control) is disabled.
8. Click the Cancel button to exit Test mode and return to the dialog window.

To prevent an object from becoming a parent control, right-click it and choose the Properties command, then set the Attach Child property to No. This doesn't affect controls already attached.

All controls in a dialog window are child controls of the dialog box (or SpeedBar), which is the parent control. To disable all controls in a dialog box, right-click the dialog box and choose the Properties command, then set Disabled to yes.

Nesting Grouped Controls

You can nest attached controls (for example, you can move a parent control on top of another control to attach them). The attached controls still stay associated with the original parent control, not that control's new parent.

See Also

Resizing Grouped Controls

Dialog Window Objects

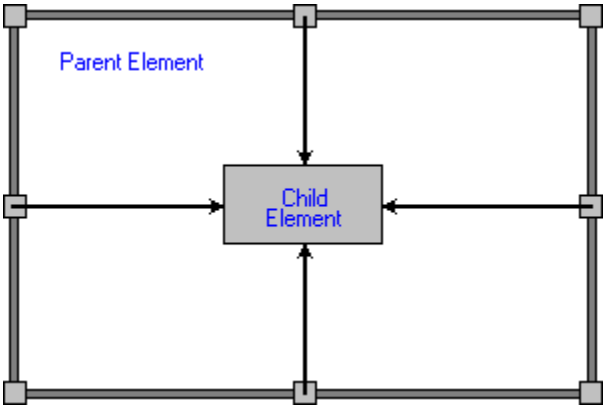
Working with Controls

Resizing Grouped Controls

Moving a parent control moves any child controls with it. (Moving child controls doesn't affect the parent.) You can use a child control's Position Adjust property to specify how it moves (or resizes) when the parent control containing it is resized.

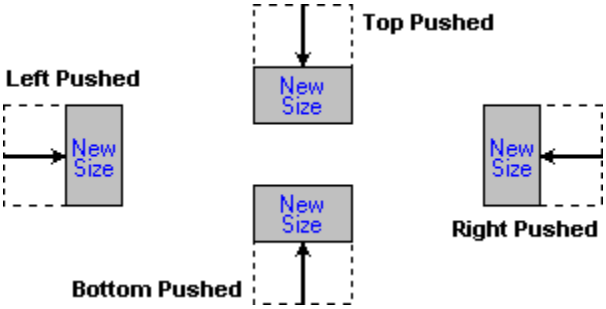
By default, a child control moves when a handle on the left or top side of its parent control is dragged. If you right-click a child control and choose the Properties command, then choose Position Adjust, and check Depend on parent, each edge of the child control is locked into place so that when the parent control is resized, the child control resizes proportionally.

When a handle of the parent control is dragged while Position Adjust is enabled, an edge of the child control is pushed or pulled, as shown by the arrows in the next figure.

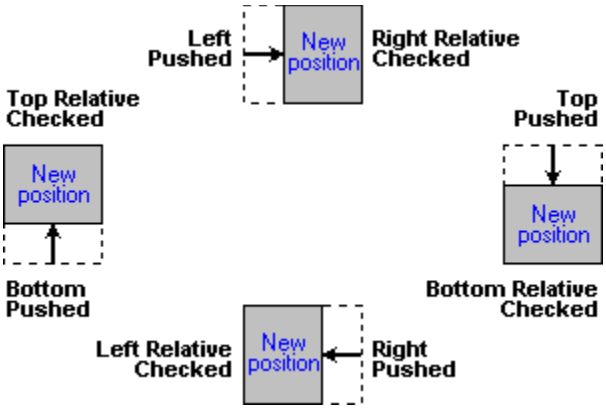


Dragging a corner handle affects two sides of the child control. For example, dragging the lower right handle would affect the bottom and right edges of the child control.

If the edge of a child control is pushed toward or pulled away from an edge of the child control that's locked, the child control is resized, as shown in the next figure. Arrows point to the edge being pushed.



You can use the check boxes in the Position Adjust dialog box to unlock edges of the child control. If an edge of the child control is pushed toward or pulled away from an unlocked edge, the child control moves instead of resizing, as shown in the next figure. Arrows point to the edge being pushed.



You can check Center Horizontally to center the control between the left and right sides of the parent control whenever the parent control is resized. Check Center Vertically to center the control between the top and bottom edges of the parent control.

Click Position Adjust Example for step-by-step directions.

See Also

Grouping Controls

Dialog Window Objects

Working with Controls

Position Adjust Example

This example shows how to use Position Adjust:

1. Choose Tools|UI Builder to display a new dialog window.
2. The OK and Cancel buttons are child controls of the dialog box. Right-click the OK button and choose the Properties command, then choose Position Adjust. Check Depend on Parent to enable automatic positioning. Choose OK to save the setting.
3. Repeat the previous step for the Cancel button.
4. Drag the lower right corner of the dialog window to shrink the dialog window. Notice that both buttons shrink proportionally.

To make the Cancel button move instead of resizing,

1. Right-click the Cancel button and choose the Properties command, then choose Position Adjust.
2. Check Top Relative and Left Relative to unlock the top and left edges of the Cancel button.
3. Choose OK to save the change and drag the lower right corner of the dialog window to shrink it.

Notice that the OK Button still shrinks, but the Cancel button moves. Now whenever the right or bottom edge of the dialog box is dragged, the Cancel button moves instead of shrinking.

See Also

Resizing Grouped Controls

Grouping Controls

Dialog Window Objects

Working with Controls

Connecting Controls to Cells or Other Controls

The Dialog|Connect command is a quick way to link a control to another control or to a notebook cell. This command exchanges settings by creating a link command in the selected object. Use Dialog|Links to modify "connected" link commands.

To connect a control:

1. Select the control you want to connect.
2. Choose Dialog|Connect, then select the cell (or control) you want to link to. The linked object's name or address appears in the Target edit field; the control's name appears in Source.
3. If you want to send changes in the control's value to the target cell or control immediately, check Dynamic Connection; otherwise, the target object will only be updated when the dialog box is closed.
4. Choose OK to save the connection and return to the dialog window.

This example connects a scroll bar to a notebook cell:

1. Select cell A1 of the notebook and enter 16. This will be the initial value for the scroll bar.
2. Choose Tools|UI Builder.
3. Create a vertical scroll bar.
4. Choose Dialog|Connect. Click cell A1, then choose OK.
5. To test the link, choose the Test button. The scroll box on the scroll bar will jump to a new position after receiving its initial value from cell A1.
6. Drag the scroll box; the value stored in cell A1 changes.
7. Choose OK to exit Test mode.

Note: To delete a "connected" link, choose Dialog|Links, select the correct link, then click Delete.

See Also

Working with Link Commands

Dialog Window Objects

Working with Controls

Providing Hints

When a user activates a control, a brief description of the control appears on the status line. To create this hint, right-click the control and choose the Properties command, then choose Help Line, type the hint, and choose OK.

See Also

Dialog Window Objects

Help Line Property

Working with Controls

Creating Object Help

Object Help identifies SpeedBar buttons and dialog box controls. To display Object Help for a control, you point to the control and press Ctrl as you right-click the mouse.

To create Object Help for a control, follow these steps:

1. Right-click the control and choose the Properties command.
2. Choose Object Help. The Object Help dialog box appears.
3. For Object Help Title, type the title of the control. For Object Help Text, type a description of how the control functions.
4. If you've created a Windows Help file for your custom application, and you want the Object Help window to display a Help button, type the context string of the relevant Help topic in the WinHelp Context field. The context string must be followed by the @ symbol and the Help file name. The name of the Help file cannot include a path; the file must be located in the Quattro Pro directory. For example, the WinHelp context for the Quattro Pro Help Contents screen is `HELP_TOPICS@QPW.HLP`.
5. Choose OK.
6. To test Object Help, choose the Test button, then point to the control and press Ctrl as you right-click the mouse.

Note: The combined total of all characters in the three Object Help fields may not exceed 194 characters.

For information about creating Windows Help files, see the Borland C++ *Tools and Utilities Guide* or the Microsoft Windows Software Development Kit (SDK).

See Also

Dialog Window Objects

Object Help Property

Working with Controls

Working with Link Commands

Link commands are statements you attach to a control, to indicate what should happen when the user changes or selects the control. A link command can do many things; it can send a value to a cell in the notebook; it can run a macro; it can close the dialog box; it can change the zoom factor in the notebook.

Link commands will usually specify at least three things: an event, an action, and an object. The event indicates what occurrence should trigger the link command; the action indicates what should happen; the object is what should be acted upon. One control can have many link commands.

The following topics explain link commands in detail.

The Object Link Dialog Box

Link Events

Assigning a Link Command to a Control

Link Command Examples

See Also

Dialog Window Objects

The Object Link Dialog Box

To add link commands to a control, select the control and choose Dialog|Links.

Buttons in the Object Link dialog box:

Button	Purpose
Add	creates a new link command below the selected link command.
Insert	creates a new link command above the selected link command.
Delete	removes the selected link command from the control.
Delete All	removes all link commands from the control.

Click Object Link Dialog Box Components for a graphic example of a control with a link command assigned to it. Each row of pick list buttons in the Object Link dialog box is a link command.

See Also

Working with Link Commands

Making Menu Commands Act

Dialog Window Objects

Link Events

A link command runs in response to some event that occurs in a dialog box (or SpeedBar). The first pick list button in a link command (the Event pick list button) shows what specific event you want to trigger the link command.

Note: A control can have more than one link command respond to the same event. The order in which the link commands appear in the Object Link dialog box is the order they'll run.

The next table lists each event and what it detects.

Event	Will trigger the link command when...
Init	The dialog box is about to appear. Use this to set initial settings.
Init Complete	Available only when adding a link command to the dialog box itself, this event occurs immediately after all controls have run all link commands that respond to Init.
OKExit	The user closes the dialog box. The link command runs before the dialog box closes.
CancelExit	The user cancels the dialog box. The link command runs before the dialog box closes.
Clicked	The user clicks the control.
Right_bdown	The user right-clicks the control.
Left_bdown	The user points to the control and is holding down the left mouse button. Releasing the button generates a Clicked event.
Doubleclick	The user double-clicks the control.
Valuechanged	The user changes the value of the control. This event occurs anytime the value is changed (by the user, by a link command, or by a macro command).
key:*keystroke*	The user presses the key *keystroke*. When you choose key from the event pick list, it flashes to indicate that you must press a key. Press the key you want to generate the event. The key displays after key: in the pick list button when it's set. For example, `key:Ctrl+F5` indicates that the link command is triggered by Ctrl+F5.
Activate	The user has chosen the control for manipulation. For example, clicking an edit field generates this event. Use Valuechanged to trap the final result.
Deactivate	The user has chosen another control, leaving this control inactive.
Enter	The user presses Enter in an edit field.
Lineup	The user increases the scroll bar's value by clicking a scroll arrow (available only on scroll bar controls).
Linedown	The user decreases the scroll bar's value by clicking a scroll arrow (available only on scroll bar controls).
Pageup	The user increases the scroll bar's value by clicking it between a scroll arrow and the scroll box (available only on scroll bar controls).
Pagedown	The user decreases the scroll bar's value by clicking it between a scroll arrow and the scroll box (available only on scroll bar controls).
Thumb	The user clicks or drags the scroll bar's scroll box (scroll bar controls only).
Editdynamic	The user is inserting or deleting characters in the combo box's edit field (combo box controls only).
Trigger	This event can't be generated by any user action; use it to set link commands that

	can only be run by other link commands. You can use the link command TRIGGER to generate this event.
Alarm	The time of day specified in the timer's Alarm Time property has been reached (time controls only).
Timer	The amount of time indicated in the timer's Timer Interval property has elapsed (time controls only).

See Also

Working with Link Commands

Dialog Window Objects

Assigning a Link Command to a Control

To add a link command to a control,

1. Select the control, then choose Dialog|Links.
2. Choose Add to create a new link command.
3. Point to the first pick list button (the Event pick list). Hold down the left mouse button, drag to the event that should initiate the link command, then release.
4. Point to the second button (the Action pick list). Hold down the mouse button, drag to the desired command, then release.
5. The remaining pick list buttons change, depending on which action you chose in step 4. Set the remaining buttons as needed.
6. Repeat steps 2 through 5 to create additional link commands.
7. Choose OK to save the link commands and return to the dialog window.

See Also

Working with Link Commands

Linking a Dialog Box to Help

Dialog Window Objects

Linking a Dialog Box to Help

You can link a dialog box or a Help button in a dialog box to a Help topic in a Windows Help file that you've created. When the dialog box is active, you can press F1 or click a Help button to display a Help topic.

To link a dialog box to Help, follow these steps:

1. Select the dialog box and choose Dialog|Links.
2. Choose Add and create the following link command:

   `On Init Set` *HelpContext@HelpFile* `To Help Context`

   In the edit field to the right of Set, type the WinHelp context string following by an at-sign (@) and the name of the Help file. The name of the Help file cannot include a path; the file must be located in the Quattro Pro directory. For example, the WinHelp context string for the Quattro Pro Help Contents screen is HELP_TOPICS@QPW.HLP.

   Note: When creating the link command, if you don't see Help on the object pick list, choose <Enter> and type `Help` in the edit field.
3. Choose OK to save the link command.
4. To test the link, choose the Test button, then press F1. The Help topic you specified appears.

Linking a Help button to Help

To link a Help button to Help, follow these steps:

1. Select the Help button and choose Dialog|Links.
2. Choose Add and create the following link command:

   `On Init Set` *HelpContext@HelpFile* `To Help Context`

   In the edit field to the right of Set, type the WinHelp context string following by an at-sign (@) and the name of the Help file.

   Note: When creating the link command, if you don't see Help on the object pick list, choose <Enter> and type `Help` in the edit field.
3. Choose Add again and create the following link command:

   `On Clicked Execute Help Context`
4. Choose OK to save the link command.
5. To test the link, choose the Test button, then click the Help button. The Help topic you specified appears.

For details on how to create Windows Help files, refer to the Borland C++ *Tools and Utilities Guide* or the Microsoft Windows Software Development Kit (SDK).

See Also

Working with Link Commands

Dialog Window Objects

Enhancing Graphs

The following topics describe how to add drawn objects, graphics, and text to graphs; import and export graphics in a variety of formats, and create slide show presentations from graphs.

Tips for Creating Lines and Shapes

Drawing Lines and Arrows

Drawing Polylines and Polygons

Drawing Freehand Shapes

Drawing Rectangles and Ellipses

Selecting Objects

Grouping and Ungrouping Objects

Arranging Object Layers

Aligning Objects

Moving and Resizing Objects

Changing Object Properties

Creating Text Boxes

Entering Text from the Clipboard

Placing Text Blocks on the Spreadsheet

Adding Bullets to Text

Changing Text Properties

Pre-setting Text and Box Properties

Importing Graphics

Cropping and Resizing Bitmaps

Modifying .CGM and .CLP Graphics

Exporting Graphics

Placing Text and Graphics on the Spreadsheet

Using Color Palettes

Creating Custom Colors

Creating Custom Color Palettes

Creating a Graph Button

Creating a Background Graph Button

Creating a Slide Show

Editing a Slide Show

Running a Slide Show

Slide Show Shortcuts

Slide Show Properties

See Also

Building Graphs

Customizing Graph Properties

Analytical Graphing
Printing Graphs

Tips for Creating Lines and Shapes

You can draw lines and shapes in any graph with the tools on the graph window SpeedBar. Here are a few rules and tips:

Pick the correct Aspect Ratio for a graph before you begin to draw; otherwise, you may have to resize, reposition, or even redraw elements to fit the final format.

To draw a shape, choose a tool on the SpeedBar. This tool stays selected until you choose a different tool. (Once you create an object, you can right-click it immediately and change its properties, but the tool will still be selected when you close the Object Inspector.)

To select fill color and background colors, fill pattern (if any), and border color and style of an object *before* you draw it, choose a color from the palette, then choose the drawing tool.

To change the color and pattern of an object *after* you create it, right-click the object and choose the Properties command to display its Object Inspector. The tool will still be selected when you close the Object Inspector.

You can also use the palette to change an object's attributes after you create it. Choose the Selection tool on the SpeedBar, select the object, then choose a color from the palette.

To change the palette displayed in the SpeedBar, select a new palette from the list box under the Cut, Copy, and Paste buttons. You can also create and save your own custom palettes (see Creating Custom Color Palettes).

Use the Draw palette (available from the palette list box) to add color to lines, polylines, freehand lines, and arrows, either before or after you draw them. Choices from this palette also make filled objects appear borderless because the fill color and border color are the same.

See Also

Drawing Lines and Arrows
Drawing Polylines and Polygons
Drawing Freehand Shapes
Drawing Rectangles and Ellipses
Enhancing Graphs
Customizing Graph Properties

Drawing Lines and Arrows

Use the Line tool to draw straight lines anywhere on a graph. Use the Arrow tool to draw a straight line with an arrow at the end. To draw a line or arrow, click the Line tool or the Arrow tool and drag from where you want the line to start to where you want it to end.

If you're using the Arrow tool, the arrowhead appears at the end of the line where you release the mouse button.

To customize a Line or Arrow, right-click it, choose the Properties command, and change its properties.

See Also

Tips for Creating Lines and Shapes

Drawing Polylines and Polygons

Drawing Freehand Shapes

Drawing Rectangles and Ellipses

Enhancing Graphs

Customizing Graph Properties

Drawing Polylines and Polygons

The Polyline tool draws lines that have more than one segment. The Polygon tool creates closed shapes with as many sides as you want. To draw a polyline or polygon,

1. Click the Polyline tool or the Polygon tool.
2. Place the pointer on the part of the graph where you want the line or segment to begin.
3. Drag to the point where you want the first segment or side to end and the second segment or side to begin, then release the mouse button.
4. Create additional segments or sides by dragging the mouse and releasing the mouse button at the end of each segment.
5. When you are ready to create the last side, double-click or right-click. Quattro Pro completes the shape for you. You can also close the polygon by clicking very close to the starting point.

To customize the Polyline or Polygon, right-click it, choose the Properties command, and change its properties.

See Also

Tips for Creating Lines and Shapes

Drawing Lines and Arrows

Drawing Freehand Shapes

Drawing Rectangles and Ellipses

Enhancing Graphs

Customizing Graph Properties

Drawing Freehand Shapes

The Freehand Polyline tool draws a line anywhere you drag the mouse. The Freehand Polygon tool creates a closed shape; you drag the mouse to draw the shape's border. To draw a freehand line or polygon,

1. Click the Freehand Polyline tool or the Freehand Polygon tool.
2. Position the mouse pointer where you want the figure to begin.
3. Hold down the mouse button and drag to draw the figure.
4. Release the mouse button to complete the polyline or polygon.

To customize the Freehand Line or Freehand Polygon, right-click it, choose the Properties command, and change its properties.

See Also

Tips for Creating Lines and Shapes

Drawing Lines and Arrows

Drawing Polylines and Polygons

Drawing Rectangles and Ellipses

Enhancing Graphs

Customizing Graph Properties

Drawing Rectangles and Ellipses

The Rectangle tool draws squares and rectangles. The Rounded Rectangle tool creates squares and rectangles with rounded edges. The Ellipse tool draws circles and ovals. To draw any of these shapes,

1. Click the tool you want to use.
2. Drag the mouse to create a figure of the shape and size you want, then release the mouse to complete the figure.

To customize the Ellipse, Rectangle, or Rounded Rectangle, right-click it, choose the Properties command, and change its properties.

See Also

Tips for Creating Lines and Shapes

Drawing Lines and Arrows

Drawing Polylines and Polygons

Drawing Freehand Shapes

Enhancing Graphs

Customizing Graph Properties

Selecting Objects

Before you can modify, move, or resize an object, you must select it.

 To select a single object, click the Selection tool, then click the object.

 To select an object group, click the Selection tool, then click any object in the group.

 To select every object in an area, click the selection tool, place the mouse pointer just outside one corner of the area, then drag diagonally until the selection box encloses all the objects.

 To select several objects individually, click the Selection tool, then hold down Shift while you click each object. To deselect an object, click it again while you still hold down Shift.

When an object is selected it has handles at its corners and sides.

See Also

Enhancing Graphs

Grouping and Ungrouping Objects

You group two or more objects when you want them to be treated as a single object in subsequent operations. For example, you might group all the objects in a complex drawing to ensure that no objects are left behind or shift position when you move them.

To group objects, select them and choose Draw|Group. To nest groups within other groups, select any combination of groups and individual objects, then choose Draw|Group. Once you group objects, you can move and resize them with the mouse.

To break a group into its original components, select it and choose Draw|Ungroup. If the group had other groups nested within it, these groups remain intact. You must use Draw|Ungroup again to ungroup them.

To change the properties of every object in a group in one operation, right-click an object in the group (NOT an area between objects) and choose the Properties command to display the group Object Inspector, then adjust property settings and choose OK. To change the fill color, background color, fill style, border color and border style of all solid objects in the group simultaneously, click the group, then click a color on the SpeedBar palette.

See Also
Enhancing Graphs
Customizing Graph Properties

Arranging Object Layers

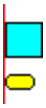
The objects you create can overlap on invisible layers. To change the order of these layers, select the object (or group of objects), then choose one of these commands from the Draw menu:

Bring Forward	Moves the selected objects forward one layer.
Send Backward	Moves the selected objects backward one layer.
Bring to Front	Places the selected objects on top of all other objects in the stack.
Send to Back	Places the selected objects behind all other objects in the stack.

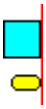
See Also
Enhancing Graphs

Aligning Objects

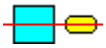
You can line up the center or edges of two or more objects using Draw|Align. This command has six options:



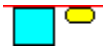
Left lines up the left sides of the selected figures.



Right lines up the right sides of the selected figures.

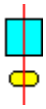


Horizontal Center lines up the centers of the objects as though they are beads on an invisible horizontal string.



Top lines up the top edges of the selected figures.

Bottom lines up the bottom edges of the selected figures.



Vertical Center lines up the centers of the objects as though they are beads on an invisible vertical string.

You can choose more than one of these commands in sequence. For example, to align the right sides and tops of a group of objects, choose Draw|Align|Right and then choose Draw|Align|Top.

When you select an object group as one of the objects to be aligned, Quattro Pro aligns the group as if it were a single object.

You can also use Snap to Grid to align objects (see Grid Options for details).

See Also
Enhancing Graphs

Moving and Resizing Objects

Once an object or group of objects is selected, you can resize or move it at will. An object is selected when it has handles.

To move the object, drag any area except its handles.

To resize only height or width, drag a side handle.

To change the height and width at the same time, drag a corner handle diagonally.

See Also
Enhancing Graphs

Changing Object Properties

The easiest and most efficient way to customize the appearance of any object is to right-click it and choose the Properties command to display its Object Inspector. (You don't have to select the object first.) Make changes to as many properties as you want, then choose OK to apply the changes all at once.

Drawn objects have many of the same properties as graph objects. For example, the rectangle Object Inspector lists properties that are also found in the graph pane and bar series Object Inspectors. Object Inspectors for drawn objects, like those for graphs, can be displayed only in the graph window.

When objects are grouped, you can right-click any object in the group (NOT the area between the objects) and choose the Properties command to display the group Object Inspector. This inspector lets you make changes that are applied to the entire group.

See Also

Customizing Graph Properties

Enhancing Graphs

Creating Text Boxes

The Text tool on the graph window SpeedBar lets you place boxed text anywhere on a graph. To set text and text box properties before you create a text box, right-click the Text tool on the SpeedBar and choose the Properties command to display its Object Inspector. Choose a text font and size, set the alignment (or change any of the other Text Box properties), then choose OK.

To create a text box,

1. Click the Text tool.
2. Drag out a text box to the size you want. When you release the mouse, a blinking insertion point signals that Quattro Pro is ready for you to enter text. (If you click the graph instead of dragging the mouse, a default-size text box appears.)
3. Enter text. As you type, Quattro Pro wraps the text to fit the width of this block, and the depth of the block increases automatically. (The Alignment property turns this wordwrap feature off and on.)
4. When you finish entering text, click the Selection tool, click another tool from the SpeedBar, or click outside the text box to deselect it.

Use standard keyboard procedures to edit the text in text boxes. Cut and Copy apply to the text when text is highlighted. When text is not highlighted, but the text box has handles around it, you cut or copy the text and the text box. To delete a text box, cut it, or move the cursor to the end of a line and press Delete.

To enhance the look of the text, change its properties (see Changing Text Properties for details).

See Also
Enhancing Graphs

Entering Text from the Clipboard

You can use the Windows Clipboard to paste text into a new or existing text box:

1. Copy text to the Clipboard (from a word processing program, for example).
2. Create a new text box with the Text tool, or click the Selection tool, then click at the point in an existing text box where you want to insert the text.
3. Click the Paste button on the SpeedBar.

See Also
Enhancing Graphs

Placing Text Blocks on the Spreadsheet

To place word-wrapped text on a spreadsheet page,

1. Create a floating text graph (see Creating a Text Graph).
2. Double-click the graph to display it in a graph window.
3. Change the graph's Aspect Ratio to Floating Graph.
4. Create a text box that is exactly the same size as the graph.
5. Enter text.
6. Change text properties, if you want.
7. Close the graph window to see the changes in the floating graph.

See Also

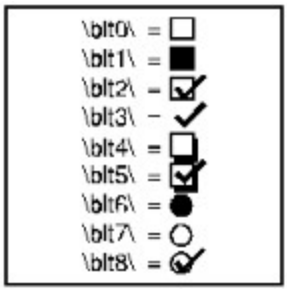
Enhancing Graphs

Changing Text Properties

Adding Bullets to Text

You can add special bullet characters to text boxes by entering a bullet code within backslash characters (\). The bullet code appears when you're editing the text, but the bullet itself appears when you're not. For example, if you enter \blt1\Sales Report in a text box, when you show the graph, a square bullet is displayed to the left of Sales Report.

These are the available bullets and the codes that create them:



See Also
Enhancing Graphs

Changing Text Properties

Text objects have two different Object Inspector menus: one for the text box and one for the text itself. To display the text Object Inspector, right-click the text and choose the Properties command. To display the text box Object Inspector, right-click any part of the background area that is not covered by text and choose the Properties command.

An easy way to determine which Object Inspector you'll see is to move (not drag) the mouse pointer over the text box. If you right-click when the pointer is an arrow, you can display text box properties. When the pointer is an I-beam, a right-click accesses text properties.

Text Properties:

Text Color	Sets a color for the text. When you select a pattern fill style, the text color is the color of the pattern. If you choose a wash fill style, the text color is distributed over the text background color.
Text Background (Bkg) Color	Provides the second color for pattern and wash fill styles. The text background color is also the shadow color.
Text Font	Determines the font and font size, and whether the text appears **bold**, *italic*, underlined, or with ~~strikeout~~.
Text Style	Includes options to fill text with a solid color, a two-color wash, or an imported bitmap. It also lets you add a drop shadow.

Text Box Properties:

Graph Button	Turns a text box into a "live" button in a slide show. When clicked, a graph button can run a macro or show a different graph (see Creating a Graph Button for more information).
Alignment	Makes the text left-justified, centered, or right-justified. It also turns wordwrap off and on, and sets tab stops at regular intervals. When wordwrap is on, text goes to the next line when it reaches the right margin, and the box expands vertically, if necessary, to fit all the text you type. When wordwrap is off, the box expands horizontally as you type; you must press Enter to start a new line.
Box Type	Determines the look of the box around the text. There are 12 choices, ranging from "no frame" to several 3-D styles.
Fill Color	Sets the color of the background in the text box. If the fill style is a pattern, the fill color is the color of the pattern, and the background color (Bkg color) is the color behind the pattern. When the fill style is a wash, the fill color is distributed over the background color.
Fill Style	Makes the text box transparent (None), or fills it with a solid color, a pattern, a wash, or an imported bitmap.
Border Color	The color of the frame around the text box.

See Also

Filling an Object with a Bitmap

Enhancing Graphs

Pre-Setting Text and Box Properties

To preset text and text box properties before you create text boxes,

1. Right-click the Text tool to display its Object Inspector, which lists all the text and text box properties.
2. Make any changes you want, then choose OK.

To create "unboxed" text, set Fill Style to None, and choose the blank (upper left) Box Type.

To preset just the Fill Color, Fill Style, and Border Color of a text box, click a color square on the palette before you choose the Text tool.

Note: The default text box Fill Color, Fill Style, and Border Color change whenever you click a palette color, even if you previously used theText tool Object Inspector to preset these properties.

See Also
Enhancing Graphs

Importing Graphics

You can bring clip art or drawings from other programs (including non-Windows programs) into the graph window with the Draw|Import command. Quattro Pro accepts files in any of these formats:

.BMP	(Bitmap) The format used in Windows paint programs, like Paintbrush, and the Windows Control Panel for wallpaper.
.CGM	(Computer Graphics Metafile) The most common vector-based clip art format.
.CLP	A graphic format used by Quattro Pro for DOS.
.EPS	(Encapsulated PostScript) The file format universally understood by PostScript printers and image setters. The file contains PostScript code for high-quality output plus a bitmap of the image for lower-resolution screen display. (If the .EPS file does not contain a bitmap image, you can't import it into Quattro Pro.)
CIS .GIF	(Graphics Interchange Format) A bitmap format originating on CompuServe Information Service (CIS) and widely used on other electronic bulletin boards.
.PCX	A bitmap format supported by many programs, including Paintbrush and Quattro Pro for DOS.
.PIC	The graph format used in the DOS versions of 1-2-3.
.TIF	(Tag Image File Format) A bitmap file format often used for scanned or gray scale images. Because .TIF files can be large, they are often compressed.

To import graphics,

1. Open the graph window where you want to import the drawing. (To import the drawing onto a new, blank graph, see Creating Text Graphs.)
2. Choose Draw|Import. You can also click the Import button on the graph SpeedBar. The Import dialog box appears.
3. Locate the graphics file you want.
4. Choose OK. The imported graphic appears in the graph window, ready to be moved or resized.

See Also

Cropping and Resizing Bitmaps

Modifying CGM and .CLP Graphics

Placing Text and Graphics on the Spreadsheet

Enhancing Graphs

Cropping and Resizing Bitmaps

When you import a bitmap graphic file (.BMP, .EPS, CIS .GIF, .PCX or .TIF format), Quattro Pro places the graphic in a rectangle. Right-click the bitmap and choose the Properties command to display the rectangle Object Inspector. When you select Fill Style, you'll see the name of the file you imported in the File Name edit field.

A bitmap fill style has two options, Crop to Fit and Shrink to Fit.

Shrink to Fit (the default)

Shrinks or enlarges the bitmap to fit within the object. To resize the object or imported bitmap, select it, then drag one of the handles. To resize only height, drag a top or bottom handle; to resize only width, drag a side handle. To change the height and width at the same time, drag a corner handle diagonally.

Crop to Fit

Displays only the part of the graphic that fits within the object. To crop a bitmap,

1. Right-click it and choose the Properties command to display the Object Inspector.
2. Choose Fill Style.
3. Select Crop to Fit and choose OK.
4. Select the bitmap to reveal the handles.
5. Drag one of the handles. As you decrease the size of the box, Quattro Pro eliminates parts of the bitmap. (The bitmap will not expand if you increase the size of the rectangle beyond its original dimensions.)

Note: The upper left corner of the bitmap is a reference point that cannot be changed or eliminated. If you drag a handle on the left or top side of the bitmap, Quattro Pro crops the right or bottom side of the bitmap, respectively.

You can use the Border Color and Border Style properties in the rectangle Object Inspector to give the bitmap a fancy border.

See Also

Filling an Object with a Bitmap

Placing Text and Graphics on the Spreadsheet

Enhancing Graphs

Modifying .CGM and .CLP Graphics

If your imported graphic file has a .CGM or .CLP file format, you can ungroup the graphic, then modify the properties of individual sections of the graphic. To do this,

1. Select the graphic.
2. Choose Draw|Ungroup. Handles appear around the individual objects that comprise the graphic.
3. Right-click any component and choose the Properties command to display and change properties such as the Fill Color and Border Style.
4. When you're finished, select all the components and choose Draw|Group to make the graphic a single object again.

See Also

Placing Text and Graphics on the Spreadsheet

Enhancing Graphs

Customizing Graph Properties

Exporting Graphics

You can export the entire contents of a graph window into other file formats. The exported file can be used in other programs or sent to a 35mm slide service bureau for processing. Quattro Pro exports to these file formats:

.BMP	(Bitmap) The format used in Windows paint programs, like Paintbrush, and the Windows Control Panel for wallpaper.
.CGM	(Computer Graphics Metafile) The most common vector-based clip art format.
.EPS	(Encapsulated PostScript) The file format universally understood by PostScript printers and image setters. The file contains PostScript code for high-quality output plus a bitmap of the image for lower-resolution screen display. (If the .EPS file does not contain a bitmap image, you can't import it into Quattro Pro.)
CIS .GIF	(Graphics Interchange Format) A bitmap format originating on CompuServe Information Service (CIS) and widely used on other electronic bulletin boards.
.PCX	A bitmap format supported by many programs, including Paintbrush and Quattro Pro for DOS.
.TIF	(Tag Image File Format) A bitmap file format often used for scanned on gray scale images. Because TIF files can be large, they are often compressed.

To export a graph and any drawn objects, imported graphics, or text boxes associated with the graph,

1. Make sure the graph you want to export is the active graph window, then choose Draw|Export. The Export dialog box appears.
2. Choose a destination and file name. Be sure to use the file-name extension that matches the type of file you want to create.
3. If you're exporting to a .TIF file, you can choose a compression method: PackBits or None (no compression). There are many types of TIF files; make sure the application to which you're exporting can accept the compression method you choose.
4. At the bottom of the dialog box is an option for Bitmap gray scale. Check this option if you want to convert the colors in the graph to levels of gray. This is useful you are going to import this file into a program that can only accept gray scale TIF files.
5. Choose OK. Quattro Pro creates the file and returns you to the graph window.

If you're exporting a graph to send to a 35mm slide bureau, be sure to set the graph's Aspect Ratio to 35mm Slide.

See Also
Enhancing Graphs

Placing Text and Graphics on the Spreadsheet

To place a picture or a block of word-wrapped text on the spreadsheet page, first create a floating text graph:

1. Select an empty cell on the spreadsheet page.
2. Click the Graph button on the SpeedBar. The mouse pointer changes to a small graph.
3. Drag the pointer over the notebook area where you want the floating graph to appear. Release the mouse button, and a blank graph appears.
4. Double-click the floating graph to bring it into a graph window. (You must add text and drawings in a graph window, where all the tools are located.)
5. Right-click the graph window title bar and set the Aspect Ratio to Floating Graph. (This ensures that the text or picture can be sized to fill the floating graph.)

To create a drawing, see Creating Lines and Shapes. To import a drawing, see Importing Graphics. After you add a drawing to the text graph, close the graph window to see the drawing "float" on the spreadsheet page.

To add a block of word-wrapped text to the spreadsheet:

1. Create a floating graph that covers the area where you want the text to appear.
2. Double-click the floating text graph to bring it into a graph window and set the aspect ratio to Floating Graph.
3. Create a text box that is exactly the same size as the graph (see Creating Text Blocks for details).
4. Enter text.
5. Use Object Inspector menus to customize the appearance of the text and text box (see Text Box for details).
6. Close the graph window to see the changes in the floating graph.

See Also
Enhancing Graphs

Using Color Palettes

The graph window SpeedBar displays a palette you can use to change the fill and border properties of graph elements and drawn objects. Each palette square has these properties:

Fill Color	Selects the interior color of the object.
Background (Bkg) Color	Provides the second color for objects that have a pattern or wash fill style.
Fill Style	Makes an object transparent (None), or fills it with a solid color, a pattern, a wash, or a bitmap graphic.
Border Color	Sets the color of the lines that form the boundary of the object.
Border Style	Sets the thickness of the lines that form the boundary of the object.

The large square to the right of the palette displays the active palette selection. Objects you create with a drawing tool display the Fill Color, Background Color, Fill Style, Border Color and Border Style shown in this square. To create an object with different properties, choose another palette color, then click a drawing tool and begin drawing.

To change the color of an existing object, select the object, then choose a color from the palette. Remember, the object acquires the Border Color and Border Style shown in your selection. If you want to change an object's Fill Color but not its Border Color and Border Style, or vice-versa, right-click it and choose the Properties command to display its Object Inspector, then make changes to the individual properties. This technique applies to all graph objects, not just drawn ones.

Quattro Pro has several predefined color palettes gathered by theme. For example, the colors in the Summer palette are mostly bright and sunny, those of Winter are dark, the Washes palette displays different colors distributed over a white background, and so on. There is also a Monochrome palette that offers 20 distinctive black-and-white pattern choices.

To select a color palette, pick it from the list box. The palette's selections appear on the SpeedBar. You can use colors from any number of palettes in the same graph.

See Also

Creating Custom Colors
Creating Custom Color Palettes
Enhancing Graphs
Customizing Graph Properties

Creating Custom Colors

Quattro Pro provides three different models for creating new colors: HSB, RGB, and CMY. Only one model displays at a time, but all three update when you change a color setting. The model to use is purely a matter of preference.

HSB	stands for Hue, Saturation, and Brightness. Hue is the basic color: red, yellow, orange, or green, for example. Saturation refers to the amount of white mixed with the hue. Brightness is the amount of light energy present. When brightness is zero, no light is present, and the result is black for every hue and saturation value. By adjusting the saturation and brightness scales in the HSB model, you can easily control the amount of gray in a color.
RGB	is named for its primary colors, Red, Green, and Blue. Using this model, you mix different amounts of red, green, and blue to form all other colors.
CMY	stands for Cyan, Magenta, and Yellow. You mix different amounts of these colors to form all other colors. Commercial printing houses often use this model to mix colors.

Once you display the Object Inspector and choose the property you want to change (Fill Color, Background Color, or Border Color, for example), use this procedure to adjust the color:

1. Select the color closest to the one you want.
2. Choose a color model.
3. Use the sliders on the color scales to adjust the color. The sample color reflects the changes as you make them.
4. When you're satisfied with the color, choose OK.

See Also

Using Color Palettes
Creating Custom Color Palettes
Enhancing Graphs

Creating Custom Color Palettes

When you can't find the color you want in any of the existing palettes, you can define your own. Once you define new colors you like, you can save them in your own palettes.

To define a new color,

1. Right-click a color square on the palette to display its Object Inspector. The Fill Color property is selected, and the scales for changing this color appear to the left.
2. Change the Fill Color, Bkg Color (which shows only when you have a pattern or wash fill style selected), the Fill Style, the Border Color and the Border Style, or any combination of these properties).
3. Choose OK.

To save the modified colors in a new palette,

1. Choose <Edit> in the palette list. The Edit Palettes dialog box appears.
2. Type a new palette name in the edit field, then click New.

The new palette is saved with Quattro Pro, not with the notebook, so you can use it anytime you're working in a graph window.

To permanently change the colors of a palette that has already been defined (including any of the ones that come with Quattro Pro),

1. Define the new color(s).
2. Choose <Edit> in the palette list. The Edit Palettes dialog box appears.
3. Choose a palette name from the list, and click Replace.

To delete a palette,

1. Choose <Edit> in the palette list.
2. Choose the palette name from the list in the Edit Palettes dialog box.
3. Click Delete.

See Also

Creating Custom Colors

Using Color Palettes

Enhancing Graphs

Customizing Graph Properties

Creating Graph Buttons

Graph buttons run macros when clicked. To turn a text box into a graph button,

1. Display a graph in a graph window.
2. Create a text box, and enter text (the box can be textless if you want).
3. Right-click the background of the text box (not the text) and choose the Properties command. The Text Box Object Inspector appears.

   Graph Button is the first choice on the property list, so its options appear automatically.

   Check the Execute Macro box if you want the graph button to run a macro when it's clicked. Enter a macro into the edit field that appears. You can either enter a named macro that you've already defined in the notebook, like {MYMACRO}, or enter macro statements of up to 80 characters, like {SlideShowRun.Goto_Slide "BUDGET"}.
4. Check Select Graph to pick the graph you want the graph button to display after it's finished running the macro. If you don't want it to display a different graph, uncheck this box. You can also choose a transition effect, speed, and display time at this point.
5. Check Overlay if you want to display this graph over the previous one in a slide show.

See Also

Creating a Background Graph Button

Enhancing Graphs

Creating a Background Graph Button

The Graph Button property in the Graph Setup and Background Object Inspector turns the graph background into a graph button. To create a background graph button,

1. Right-click the background of the graph and choose the Properties command, or choose Property| Graph Setup. The Graph Setup and Background Object Inspector appears.
2. Choose Graph Button.

   Check the Execute Macro box if you want to run a macro whenever the background is clicked. Enter a macro into the Macro edit field. You can either enter a named macro that you've already defined in the notebook, like {MY MACRO}, or enter a macro statement of up to 80 characters, like {Slide.Goto_Slide "BUDGET"}.
3. Check the Select Graph box, then choose the name of the graph you want the graph button to display when the background is clicked. (If you've entered a macro, this graph will appear when the macro is run.) If you want a clicked background to redisplay the current graph, pick the current graph name from the list.

See Also

Creating a Graph Button

Enhancing Graphs

Creating a Slide Show

You create slide shows in the Graphs page. The Graphs page has an icon for every graph you've made in the notebook, and it has tools in its SpeedBar for creating, editing, and running slide shows.

To create a slide show,

1. Click the Create Slide Show tool.
2. Enter a name for the slide show and choose OK. An icon for the new slide show appears on the Graphs page.
3. To add graphs to the slide show, drag one or more graph icons over the slide show icon. When the slide show icon becomes highlighted, release the mouse button. The graph becomes part of the slide show, and the graph icon snaps to its original position.

   If any graph icons were selected when you clicked the Create Slide Show tool, the graphs were automatically added to the slide show as well, in the order in which they were selected.
4. Continue adding graphs to the slide show as in the previous step. Graphs appear in the order in which you add them. You can add the same graph to the slide show more than once, if you need to.

See Also

Editing a Slide Show

Running a Slide Show

Slide Show Shortcuts

Slide Show Properties

Enhancing Graphs

Running a Slide Show

To begin a slide show, choose Graph|Slide Show. Choose a slide show name from the list, then choose OK.

To start a slide show from the Graphs page, select a slide show icon, then either choose Graph|Slide Show or click the Run Slide Show button on the SpeedBar.

When a slide show is running, the following actions have special meaning:

Action	Result
Left-click	Go to next slide
Right-click	Go to previous slide
Press Backspace	Go to previous slide
Press Esc	Cancel slide show
Click a graph button	(or type its first letter) Perform the actions defined in the graph button
Press any other key	Go to next slide

Note: Quattro Pro for Windows will run Quattro Pro for DOS slide shows, which are built on the spreadsheet. After you open the DOS file, highlight the range that contains slide show information, choose Graph|Slide Show, give the show a name, and choose OK to simultaneously create the slide show and run it. Sound files will be ignored. An icon appears on the Graphs page to represent the slide show. You can edit the slide show on the spreadsheet page, or use the procedure in Editing a Slide Show.

See Also

Creating a Graph Button

Creating a Background Graph Button

Creating a Slide Show

Editing a Slide Show

Slide Show Shortcuts

Slide Show Properties

Enhancing Graphs

Editing a Slide Show

Editing a slide show is easy because Quattro Pro displays the slides as though they were laid out on a light table. You can change the slide order, transition effect, display time, and transition speed assigned to each slide.

To edit the contents of a slide show, double-click its icon on the Graphs page or select it and click the Edit Slide Show button on the SpeedBar. The Light Table dialog box appears, which shows all the graphs in the slide show in miniature form, in the order in which they will appear.

In this dialog box, you rearrange slides and edit visual effects.


To move a slide to a new position, drag the miniature of the graph to a new location. When you're rearranging slides in this way, the pointer changes to a hand. When you release the mouse button, the graph moves to the new position. The light table scrolls to reveal more slides if you drag the pointer to the top or bottom edge.


To delete a graph from the Slide Table, select it and press Del.

For special effects information, see Adding Transition Effects.

See Also

Creating a Slide Show

Running a Slide Show

Slide Show Shortcuts

Slide Show Properties

Enhancing Graphs

Adding Transition Effects

By default, each slide immediately "cuts" to the next in a slide show. This is usually the best way to lead into the next graph because it is unobtrusive and quick. You can choose a different effect from the Effect list at the bottom of the Light Table dialog box. This list contains all the visual transition effects available. The effect you assign to a graph controls the way that graph makes its appearance on the screen. (The way the graph disappears from the screen depends on the transition effect of the next slide in the slide show.)

The names of the effects in the list box describe what the effect does. When the Overlay box is checked, the second image overlays the first instead of replacing it. The best way to choose a visual effect is to try as many as you can, then select the most appropriate one. Here are some tips on using visual effects:


Use wipes to simulate turning a page because the second slide appears to be underneath the first one. Use tilts to give the effect of one image pushing away another.


Avoid slower effects like the dissolve--2x2 pixels unless there's good reason to keep your viewers waiting (perhaps to increase suspense). Moderate dissolves and fade out/fade in can be restful, but fast dissolves (when run at full speed) can appear almost violent.


Use overlays to build slides from other ones. This is often effective when discussing a list of bulleted items. The first slide shows only the first bulleted item. The second slide then builds on the first by adding the second item. The subsequent slides continue building until all items are onscreen.


Take care to match the effects to the overall tone of your presentation. Showy effects like spirals, sides to center, center to sides, vertical stripes, or diamonds are best in small doses; they can look out of place in serious board rooms.

The speed of the transition effect is controlled by the Slow, Medium, and Fast buttons.

If you want the slide to remain on the screen for a set number of seconds (up to 3600) instead of waiting for the viewer to click or press a key, enter a number in the Display Time edit field. Entering 0 (the default) means the slide stays on the screen until the viewer clicks the mouse or presses a key.

See Also

Creating a Slide Show

Editing a Slide Show

Running a Slide Show

Enhancing Graphs

Slide Show Shortcuts

If you're in a hurry and don't want to bother with some of the fine points of putting together a slide show, there are several ways to accelerate the process:

To get a fast idea of what your graphs will look like full-screen, select the graph icon and choose Graph|View. Even better, select several icons and choose Graph|View to see them in sequence. Graph buttons continue to work when you display graphs in this way.

To quickly assign graphs to a slide show, select their icons--in the order you want them shown--before you create the slide show icon.

To swiftly set a transition effect, a transition speed, and a duration period for all the slides in the slide show, right-click the slide show icon, choose the Properties command, and change the Default Effect property. The options you select here also apply to every graph you subsequently add to the slide show. (To give graphs individual settings, double-click the slide show icon and use the Light Table dialog box.)

To create a new text graph while you're working in the Graphs page, click the New Graph button on the SpeedBar or choose Graph|New. The usual series dialog box appears with all series blank. Leave all the series blank and choose OK to begin editing a new text graph. (You can also create a numeric graph by clicking series buttons, then pointing to blocks in a spreadsheet page. See Assigning Series by Pointing for instructions.)

See Also

Creating a Slide Show

Editing a Slide Show

Running a Slide Show

Slide Show Properties

Enhancing Graphs

Slide Show Properties

The slide show has several properties of its own. Right-click the slide show icon and choose the Properties command to display them.

Name	Lets you rename the slide show. Choose this property, enter a new name, then choose OK. The new name appears under the slide show icon.
Default Effect	Sets a transition effect, a transition speed, and a duration period for all the slides in the slide show.
Show Pointer	Hides the pointer throughout the show, which is useful when your slide show contains no graph buttons. Choose this property, then choose No to hide the pointer, or Yes to display it.

See Also

Creating a Slide Show
Editing a Slide Show
Running a Slide Show
Slide Show Shortcuts
Enhancing Graphs

Basic Editing Tasks

These topics describe how to perform basic data editing tasks:

Selecting Cells and Blocks

Before you can enter data or perform an action in a notebook, you need to select the cell or block(s) you want to affect.

To select

 a cell, click the cell.

 a block, drag it by clicking a cell in one corner, holding down the left mouse button, moving to the opposite corner and releasing the mouse button. Or click one corner, hold down the Shift key, and click the opposite corner.

 a noncontiguous block (a selection consisting of more than one disconnected block of cells), hold down the Ctrl key while you drag blocks. See Using Noncontiguous Blocks.

 every cell in a given row or column, click the corresponding row number or column letter in the border.

 all cells in the active notebook page, click the Select-All box at the intersection of the row and column borders.

 a cell on another page, click the page tab, then click the cell you want.

 a 3-D block (a block with the same coordinates in multiple pages), first select the block in the first page in the block, then hold down the Shift key while clicking the tab of the last page for the block. A black line appears under the tabs of the selected pages. See Using 3-D Blocks.

You can also use the keyboard to select in a notebook (see Point Mode Keys and Selecting Blocks with GoTo for more information).

When you select a block, the active cell within the selected block contains the selector, while the remainder of the selected block is highlighted.

See Also

Editing Entries

Basic Editing Tasks

Editing Entries

There are two ways to change the data in a cell:

Select the cell and type a new entry. When you press Enter or click the check mark in the input line, the new entry replaces the old.

Select the cell and edit the existing entry.

When you edit an entry, you can insert or delete characters without retyping the whole entry. Begin by selecting the cell you want to change. Then click anywhere in the input line or press F2 to enter Edit mode. The EDIT indicator appears on the status line, and the contents of the active cell appear in the input line with an insertion point at the place where you clicked, or at the end if you pressed F2.

You can move the insertion point to any part of the entry by clicking in the input line or by using the arrow keys. Backspace deletes characters to the left of the insertion point, and Del deletes characters to the right. You can also drag in the input line to select characters.

After you've made a selection in the input line, you can press Del to erase the selection, or start typing or pointing to replace the selection. See Using Blocks in Formulas for more information on pointing. You can also use the Clipboard commands on your selection (see Copying and Moving with the Windows Clipboard for details).

All of the common editing keys work just as you'd expect them to work in a word processor. For details, see Edit Mode Keys.

When you first enter Edit mode, you're also in Insert mode . Any characters you type are inserted to the left of the insertion point. To write over existing characters, press Ins to switch to Overtype mode. (OVR appears on the status line.) Press Ins again to return to Insert mode.

Quattro Pro doesn't store your changes in the cell until you click the check mark or press Enter. Until then, you can click the X button or press Esc twice to return the entry to the way it was before.

See Also

Erasing Entries

Basic Editing Tasks

Erasing Entries

There are several ways to erase data. You can

 erase the contents of a block with Edit|Clear Contents or by pressing Del.

 erase and move the contents of a block to the Clipboard with Edit|Cut (see Copying and Moving with the Windows Clipboard for more information).

 remove the data and the cells containing it by deleting entire or partial rows, columns, or pages with Block|Delete (see Deleting for more information).

To erase block contents, select the block, then choose Edit|Clear Contents or press Del. The block's properties (alignment, numeric format, font, and so on) remain unchanged.

To erase block contents and remove property settings, select the block, then choose Edit|Clear. For more information on block properties or the Normal style, see Changing Block and Page Properties. If you accidentally erase data, use Edit|Undo to restore it.

See Also

Undoing Mistakes

Editing Entries

Basic Editing Tasks

Copying and Moving Data

There are three ways to copy and move blocks in Quattro Pro:

- Use the Drag and Drop feature to drag a block of cells anywhere on the page.
- Use the Clipboard commands. Use Edit|Copy and Paste in combination to copy blocks, or use Edit|Cut and Paste in sequence to move blocks.
- Use Block|Copy or Move, commands that are familiar to users of DOS spreadsheets.

You can also move entire pages (see Moving Pages for more information).

Choosing a method

The advantage of Drag and Drop copying or moving is its ease and visual nature (see Dragging and Dropping for details).

The advantages of the Clipboard commands are (see Copying and Moving with the Windows Clipboard for details):

- You can copy or move text within the input line or to other text objects.
- You can paste the same block to many different destinations without reselecting the source.
- You can copy data to other Windows applications.

The advantages of the Block menu commands are (see Using Block Commands for Copying and Moving for details):

- You can specify blocks to be copied or moved by name.
- If you're already familiar with these commands, or if you have macros that use them, they are more convenient.
- Block menu commands work a bit faster than the Clipboard commands.
- The Block|Copy command provides a unique, time-saving feature called Model Copy. Use it to copy blocks that contain absolute references to cells within the copied block.

See Also

Copying Formulas

Basic Editing Tasks

Dragging and Dropping

For a quick and simple way to copy or move a block of cells, use Drag and Drop. To move a block,

1. Select the block.
2. Click anywhere in the block, hold down the mouse button, and wait a moment. The mouse pointer changes to a hand, and a colored outline appears.
4. Drag the block to where you want to move it, using the colored outline as a guide. Be careful; data under the colored outline will be overwritten.
5. Release the mouse button to "drop" the block into place.

To copy a block, follow the steps above, except hold down the Ctrl key after selecting the block in step 1.

Drag and Drop doesn't work on noncontiguous blocks, and you can't make multiple copies. Depending on which one of these things you want to do, use the Clipboard commands, or Block|Copy or Move.

You can use the Delay Time property in the application Object Inspector to adjust the length of time before the hand icon appears.

See Also

Copying and Moving with the Windows Clipboard

Copying Formulas

Using BlockCommands for Copying and Moving

Moving Pages

Moving Formulas or Referenced Cells

Basic Editing Tasks

Copying and Moving with the Windows Clipboard

Edit|Copy and Edit|Cut place copies of the current selection (the source) to the Clipboard in preparation for being pasted elsewhere (the destination). While Copy leaves the source intact, Cut erases it.

The Clipboard commands in Quattro Pro work as in other Windows applications. In Quattro Pro, however, they are available as buttons from the SpeedBar:

1. Select the source you want to copy or move.
2. Click Copy (to copy data) or Cut (to move data).
3. Select the destination. The destination can be a cell on this or another page, a cell in another notebook, or characters in the input line or in a text box. If you choose a cell, the entire source is pasted, using that cell for the upper left position of the pasted copy.

Caution: If the destination already contains data, you'll overwrite it when you choose Paste. Be careful that you allow enough space for the complete source to be pasted.

3. Click the Paste button to paste the data into the destination.

Pasting Multiple Copies

If the source is a single cell, column, or row, you can make multiple copies of it by selecting a block as the destination.

You can also make multiple copies of data you've cut. The first time you click the Paste button (or choose Edit|Paste), the source is simply moved. Clicking or choosing Paste the second time copies the source throughout the destination area.

Copying Noncontiguous Blocks

If the source is a set of noncontiguous blocks, imagine the noncontiguous blocks contained by one large block. Choose the new position for the upper left corner of that large block for the destination.

See Also

Dragging and Dropping

Copying Formulas

Using Block Commands for Copying and Moving

Moving Pages

Basic Editing Tasks

Copying Formulas

When you copy data, whether you use Drag and Drop, Block|Copy, or Clipboard commands, Quattro Pro adjusts formulas to correspond to their new positions. This is helpful in most situations, but there are times when you should avoid adjusting references.

Relative addresses

Unless you specify otherwise, cell references in a formula are relative. Quattro Pro finds each cell reference by its position in relation to the cell containing the formula, not by its address.

For example, if you enter +A1+A2 in cell A3, Quattro Pro interprets it as "Add the values in the two cells directly above this one." When you copy this formula to cell B4, Quattro Pro adjusts the formula to add the values in cells B2 and B3 (the two cells directly above).

Absolute addresses

To prevent a cell reference from adjusting when you copy a formula, make the reference absolute by entering dollar signs ($) before the column and the row coordinates. An absolute cell reference always refers to the original cell address, regardless of where you copy the formula. You can also make a page reference absolute if you're copying from one page to another.

For example, if you enter +$A$1+$A$2 in cell A3, Quattro Pro reads this as "Add the values in the cells A1 and A2." When you copy this formula to cell B4, Quattro Pro still refers to cells A1 and A2.

You can specify all or part of a cell address as absolute. Just insert a dollar sign ($) before the coordinate you want to remain fixed. For example,

$A$1 makes both coordinates of address A1 absolute.

$A1 locks the address into column A, but lets the row coordinate adjust.

A$1 locks the address into row 1, but lets the column coordinate adjust.

$A:A$1 locks the address into page A and row 1, but lets the column coordinate adjust.

When copying between pages, you can make page references absolute also. Include a dollar sign before the page name. If you omit the page reference in a formula, it will adjust to refer to the current page, wherever it is copied.

The Abs key (F4) simplifies inserting dollar signs in an address. Press F4 when a cell address is highlighted in the input line.

The next table shows the effect of each repeated press of F4.

Press F4	Cell reference	Block name
starting reference	B6	SALES
1x	$A:$B$6	$SALES
2x	$A:B$6	SALES
3x	$A:$B6	
4x	$A:B6	
5x	$B$6	
6x	B$6	

7x $B6

To specify a named block as absolute, precede its name with a dollar sign. To specify a block as partially absolute, use its coordinates, not its name.

See Also

Dragging and Dropping

Copying and Moving with the Windows Clipboard

Using BlockCommands for Copying and Moving

Moving Pages

Basic Editing Tasks

Using Block Commands for Copying and Moving

Use Block|Copy or Block|Move to copy or move a cell or a block. With Block|Copy, you can also copy noncontiguous blocks. To copy or move data or objects other than cell blocks, use the Clipboard commands.

Copying

To copy a block of cells,

1. Choose Block|Copy.
2. The current selection appears in the From box, which contains the coordinates of the source block (what you want to copy). If this entry is correct, skip to step 4.
3. To change the From entry, double-click in the dialog box, then point to the cells you want to copy. Instead of pointing, you can type new coordinates, or press F3 to choose from a list of block names (see Naming Blocks for more information). The source block can be the following types of references:


Cell or block references; either named or not (such as B4..B4 or C5..D10 or INCOME)


Noncontiguous blocks (such as B4..B4,C5..D10 with Block|Copy)


Blocks in other pages in the same notebook (such as C:B4..B6)


Blocks in other notebook files (such as [C:\BUDGETS\SALES]B:D10..D15)

When you're finished pointing, click the Maximize button in the title bar of the reduced dialog box (see Pointing from Dialog Boxes for details).

4. Select or double-click the To entry. Then to make a single copy, point to (or type the address of) the upper left cell of the destination block (where you want to place the copy).

Caution: If the destination already contains data, you'll overwrite it when you choose OK.

5. If appropriate, check the Model Copy option (see Using the Model Copy Option for details). Then choose OK.

Moving

Block|Move moves the contents of a block from one location to another. It overwrites existing data in the new location. To move a block,

1. Choose Block|Move.
2. Enter the source block in the same way as for Block|Copy except that you can't move a noncontiguous block. Specify the upper left cell of the destination block.

Caution: If the destination already contains data, you'll overwrite it when you choose OK.

3. Choose OK.

The data is pasted along with its block properties.

See Also

Dragging and Dropping

Copying and Moving with the Windows Clipboard

Copying Formulas

Moving Pages

Basic Editing Tasks

Using the Model Copy Option

As explained in Copying Formulas, relative and absolute addresses let you control how formulas adjust when you copy them. The Model Copy option allows another variation on relative and absolute addresses that's useful when you're copying a block that contains absolute references to cells within the copied block. Model Copy can save painstaking and error-prone minutes of editing formulas.

The example below shows a small model containing a formula to figure the monthly payment for a 30-year loan at different interest rates. The reference to the loan amount had to be absolute so that when the formula was copied, it continued to refer to cell B1. Click Model Copy Examples for details.

You might want to calculate monthly payments for a different loan amount. To do so, copy the block, change the loan amount entry, and expect to see the new monthly payment values. But as shown in the example, without using Model Copy, the absolute reference inappropriately refers to row 1.

One solution is to edit each formula individually to refer to B6 instead of B1. An easier solution is to check Model Copy before copying; then even absolute references adjust to the new location of the referenced cell. Although the absolute reference adjusts, it remains absolute to make future copies absolute. In the previous example, should you want to make more copies of the formula for interest rates below 8% for a $150,000 loan, the reference to cell B6 is still absolute.

Several options help you use Model Copy more precisely. For details, see Model Copy Option Settings

See Also

Copying and Moving Data

Copying and Moving with the Windows Clipboard

Basic Editing Tasks

Model Copy Option Settings

These options are available when you check Model Copy in the Block Copy dialog box. You can check as many as you want. When checked,


Formula Cells copies all formula cells in the source block.


Value Cells copies all source block cells containing values instead of formulas.


Properties copies all block properties applied to each source block cell.


Objects copies all SpeedButtons, floating graphs, and other objects in the source block.


Row/Column Sizes makes the destination block rows and columns the same sizes as those in the source block.

See Also

Using the Model Copy Option

Basic Editing Tasks

Moving Pages

You can move pages within a notebook to reorder them. With the mouse, you can directly manipulate the pages, similarly to the Drag and Drop feature used on cells:

1. Click the tab of the page you want to move and drag down toward the bottom of the screen until the mouse pointer changes to a hand.
2. Drag right or left along the row of other tabs, moving the highlighted tab that appears. Release the mouse button when the highlighted tab is just before the tab where you want to place the moved page.

The moved page is dropped into place.

If you move a page into or out of grouped page while Group mode is on (the blue line appears under the tabs), the page moves alone, and the group expands or shrinks accordingly.

If you drag a page that is part of a selected 3-D block (the black line appears under the tabs), all pages in the 3-D block move together.

You can also move pages with Block|Move Pages. From the dialog box, enter the page name to be moved and the page name before which you want the page placed. Then choose OK.

See Also

Dragging and Dropping

Copying and Moving with the Windows Clipboard

Copying Formulas

Grouping Notebook Pages

Using 3-D Blocks

Using BlockCommands for Copying and Moving

Basic Editing Tasks

Basic Search and Replace Procedure

To find or replace entries,

1. Select the block or blocks you want to search.
2. Choose Edit|Search and Replace.
3. Choose Find and enter the find string.
4. Choose Replace and enter the replacement string.
5. Specify any Look In or Options settings you want.
6. To begin the search, choose Next or Previous.
7. If a string is found, choose Next, Previous, Replace, Replace All, Reset, or Cancel. See Search and Replace Options for more details.

See Also

Searching for Records

Basic Editing Tasks

Search and Replace Options

Edit|Search and Replace lets you find or alter multiple cell entries with a single command. It searches for the specified value or string and finds each match, which you can choose to replace with a new value.

The dialog box offers these options:

Block(s) is the block or blocks to search. To search the entire page, this edit field should be blank.

Find is the group of characters to be found in labels, values, or formulas. These characters are called the find string.

Replace is the group of characters to substitute for characters found. This is called the replacement string.

The first two Look In options determine how Quattro Pro treats formulas during its search. The last option lets you set conditional searches through all types of entries.

Formula looks at the formulas as they appear in the input line.

Value looks at the results of the formulas, as they appear in the cells.

Condition treats the find string as a conditional expression. For example, a find string of B13>500 with Condition chosen looks for the first value that is greater than 500, starting with cell B13. To search starting from the active cell (which is usually the upper left cell), substitute a question mark for the cell address, for example, ? > 500. See Types of Operators for details on the logical operators that can be used in conditional find strings.

Note: If you're using a conditional expression, you can only find values; they aren't replaced when found.

Columns First sets the search path. By default, this option is unchecked for searching across rows starting with row 1. When checked, searching occurs down columns starting with column 1.

Match Whole determines whether the find string must be all or part of a cell entry. By default, this option is unchecked for searching for partial as well as whole entries. For example, if the find string is `cat`, `catamaran` and `scatter` are found as well as `cat`.

Note: If you're searching for a label with Match Whole checked, you must either include the label prefix or set Look In to Value.

Case Sensitive controls whether entries must exactly match the find string. By default, this option is unchecked for searching for strings regardless of capitalization. For example, if the find string is `HARPER`, `Harper`, `harper`, and `HaRpEr` are found as well as `HARPER`.


The Next button begins or resumes a forward search without replacing found entries.


The Previous button begins or resumes a backward search without replacing found entries.


The Replace button lets you decide on an individual basis whether to replace each string found.


The Replace All button replaces all found strings without stopping.


The Reset button removes any entries in the dialog box and reinstates the defaults.


The Close cancels the find operation without making a change.

You can also use Data|Query to search a database.

See Also

Basic Search and Replace Procedure

Searching for Records

Basic Editing Tasks

Inserting

You can insert blank columns, rows, or pages anywhere in the notebook. You can also insert partial rows, columns, or pages. Wherever you insert, existing data is pushed down, to the right, or to the back of the notebook to make room.

You can also insert from another file onto a new page.

Note: If inserting spaces pushes a named block or cell reference beyond the limit of a page (beyond column IV or row 8192), or beyond page IV, the reference becomes ERR.

Inserting Rows or Columns

Inserting Pages

Inserting Blocks

See Also

Inserting a File into a Notebook

Basic Editing Tasks

Inserting Rows or Columns

To insert an entire row, click the row border just below where you want the row inserted and click the Insert button in the SpeedBar. To insert an entire column, click the column border just below where you want the column inserted and click the Insert button.

To insert multiple rows or columns, select as many row or column borders as you want to insert.

You can also insert a row or column by choosing Block|Insert|Rows or Columns. As with the Insert and Delete buttons, select the row below or the column to the right of where you want to insert before choosing the command.

If you insert a column or row within the boundaries of a named block or a block referenced by a formula, Quattro Pro expands block references to include the new column or row.

See Also

Inserting Pages

Inserting Blocks

Basic Editing Tasks

Inserting Pages

To insert a blank page,

1. Click the tab of the page you want to follow the new page.
2. Click the Insert button.
3. Choose Pages and choose OK.

To insert more than one page, select the corresponding number of pages in step 1. (To select multiple pages, click the first page tab, then hold down Shift while you click the last page tab you want. A black line appears under the tabs.) For example, to insert two pages before page B, select pages B and C before following steps 2 and 3; the data in page B moves to page D.

You can also insert pages before the active page by choosing Block|Insert|Pages. In the dialog box:

1. Enter a block in the Block edit field. To insert only one page, enter the page to insert before, and any cell on the page, such as B:B12. To insert multiple pages, enter a 3-D block starting with the page to insert before and spanning the same number of pages you want to insert. For example, to insert three pages before page B, enter B:A3..D:A3 (it doesn't matter which cell you reference).
2. Choose Entire and choose OK.

See Also

Inserting Rows or Columns

Inserting Blocks

Basic Editing Tasks

Inserting Blocks

Instead of inserting an entire row, column, or page, you can insert partial ones (a block). Data in adjacent cells shifts to accomodate the inserted cells. When you insert a partial page, data shifts to the next page.

To insert a block,

1. Select a block in the same location where you want to insert space; make sure the upper left corner of the block you select contains the first cell entry you want shifted right, down, or back. The block you select should be the same size as the amount of space you want to insert.
2. Click the Insert button.
3. From the dialog box, choose Partial.
4. Choose Rows if you want the selected entries to shift down out of the way. Choose Columns if you want them to shift to the right. Choose Pages if you want them to shift to the next page.
5. Choose OK.

You can also insert a block by choosing Block|Insert|Rows, Columns, or Pages. A dialog box appears. In the Block edit field, enter the coordinates of the block to be shifted down (for rows), to the right (for columns), or to the next page (for pages). Then choose Partial and choose OK.

See Also

Inserting Rows or Columns

Inserting Pages

Basic Editing Tasks

Deleting

You can delete entire or partial columns, rows, or pages anywhere in the notebook. Deleting space is different from erasing (or blanking) data with Edit|Clear, Clear Contents, or the Del key. Deleting with the Delete button in the SpeedBar.or Block|Delete makes remaining rows, columns, or pages move to take up the deleted space.

Caution: If you delete space that is within the boundaries of a named block or a block referenced by a formula, Quattro Pro adjusts all references to the block. However, if one of the deleted cells is a coordinate cell that defines a block, the block becomes invalid and any formulas or names referencing the block show ERR. Any formulas that reference a cell within a deleted column, row, or page also appear as ERR.

Deleting Rows or Columns
Deleting Pages
Deleting Blocks

See Also

Copying Formulas
Basic Editing Tasks

Deleting Rows or Columns

To delete an entire row, click the row border and click the Delete button. To delete an entire column, click the column border and click the Delete button.

To delete multiple rows or columns, select as many row or column borders as you want to delete.

You can also delete a selected row or column by choosing Block|Delete|Rows or Columns.

See Also

Deleting Pages

Deleting Blocks

Deleting

Deleting Pages

To delete a page,

1. Click the tab of the page to be deleted.
2. Click the Delete button.
3. Choose Pages and choose OK.

You can also delete a page by clicking its Select-All box (to select all cells) and clicking the Delete button or by clicking its tab and choosing Block|Delete|Pages.

To delete more than one page, select the corresponding number of pages in step 1. (To select multiple pages, click the first page tab, then hold down Shift while you click the last page tab you want. A black line appears under the tabs.) For example, to delete pages B and C, select them before following steps 2 and 3. The data in page D moves to page B.

See Also

Deleting Rows or Columns

Deleting Blocks

Deleting

Deleting Blocks

Instead of entire rows, columns, or pages, you can delete partial ones--also known as blocks. Data in surrounding cells shift to take up the deleted space, as shown in the next figure.

When you delete a partial page, data moves forward from the next page.

To delete a block,

1. Select the block you want deleted.
2. Click the Delete button.
3. Choose Partial.
4. Choose Rows to shift entries up into the selected block, choose Column to shift entries left, or choose Pages to shift from the next page.
5. Choose OK.

You can also delete a selected block by choosing Block|Delete|Rows, Columns, or Pages. A dialog box appears; choose Partial and choose OK.

See Also

Deleting Rows or Columns

Deleting Pages

Deleting

Using Named Blocks

Instead of referring to a single cell or a block by its coordinates, such as B10 or C15..F21, you can assign a specific name. This has several advantages:

- Names are easier to remember than coordinates.
- If you move the contents of a named block, Quattro Pro still associates the name with the same data, regardless of the new block coordinates.
- Referencing block names in a formula makes the formula easier to understand. For example, +PRICE - COST is more intuitive than B15 - D8.
- Using block names increases accuracy. If you mistype a block name, Quattro Pro alerts you to the error. If you mistype block coordinates, you operate on the wrong block.
- When you link to another notebook, you won't need to open the other notebook to find the block coordinates.

Consider the following formula, which calculates a monthly mortgage payment:

```
@PMT(B3,B4/B6,B5*B6)
```

Here is the same formula using named blocks:

```
@PMT(PRICE,INTEREST/MONTHS,TERM*MONTHS)
```

The following topics explain how to create block names and how block names operate with formulas.

Naming Blocks

Using Block Names in Formulas

See Also

Copying Formulas

Basic Editing Tasks

Naming Blocks

Keep the following guidelines in mind when naming blocks:

Block names can be up to 15 characters long.

Use any keyboard characters (A to Z, 0 to 9, punctuation marks, and some special characters such as %, or ~). Accented characters such as Â and Ä are also allowed if they're available on your keyboard.

Avoid using operator characters (+, -, *, /, ^, =, <, >, #, or &), $, open and close parentheses, and spaces.

Uppercase and lowercase letters are equivalent; in other words, INCOME is the same as income. Block names always appear in uppercase letters in formulas.

Don't use numbers alone (such as 327) or valid cell addresses (such as G1) as block names.

Block names can define overlapping areas. For example, the following group of block names is acceptable:

HOTEL	B3..B7
TRANS	C3..C7
MEALS	D3..D7
TOTAL	B3..D7

To assign a name to a block of cells,

1. Select the block you want to name.
2. Choose Block|Names|Create.
3. Type a name for the block in the Name box. Enter a name not included on the list of existing names at the left, and choose OK.

To change the block assigned to a name, choose Block|Names|Create and choose the name from the list. Then enter new coordinates in the Block(s) edit field.

To change the name of a block, delete the existing name first, then assign the new name to the block.

Caution: If you use Block|Move to alter the contents of the upper left or lower right cells of a named block, the name becomes invalid. Formulas referencing the named block display ERR to indicate the formulas no longer reference the data they did before. Deleting any of the two coordinates of a block makes any reference to that block display ERR, also. See Moving Formulas or Referenced Cells for details.

See Also

Deleting Block Names

Copying Formulas

Using Block Names in Formulas

Basic Editing Tasks

Using Block Names in Formulas

To reference a named block, type the block's name or choose it from a choice list.

To display a choice list while writing or editing a formula, press F3. To enter a name onto the input line, choose the name from the list and choose OK.

To display the list while entering or editing a formula, the insertion point must be to the right of an operator or open parenthesis. For example, in the formula

```
+C7*@SUM(B6..D19)
```

you can display a list of block names whenever you can enter Point mode (see Pointing to Blocks for details).

To expand the list of names to show their coordinates, press the Expand key (+). Press the Contract key (-) to remove the coordinates. Press F3 again to expand the names list to full screen or shrink it back down.

Note: The Choices key, F3, has different purposes in other situations. In Value or Edit mode, Alt+F3 displays a list of @functions and Shift+F3 displays a menu of macro categories. These two key combinations are equivalent to clicking the @Functions button and the Macros button in the Edit mode SpeedBar.

If you edit a formula that already contains a reference to a named block, it appears in the input line as block coordinates. After you press Enter to finish editing, the block coordinates switch back to the block name reference in the formula.

For information on an automated way to name blocks (Block|Names|Labels), delete block names (Block|Names|Delete and Reset), and view a table of block names (Block|Names|Make Table), see Maintaining Block Names.

See Also

Copying Formulas

Naming Blocks

Basic Editing Tasks

Using the GoTo Command

Edit|GoTo is useful for quickly moving to named blocks (to identify or modify them) or for moving quickly to distant parts of the notebook, or for selecting a noncontiguous block in disparate parts of a notebook. The selector moves to whatever location you choose with the command.

To move the selector to any location quickly,

1. Choose Edit|GoTo.
2. Choose the target block name from the list, or type the target cell address in the text box. If the cell is on another page, add the page prefix preceded by a colon. For instance, to move to cell Z36 on page D, type D:Z36. If the cell is in another notebook, add the notebook prefix. For instance, to move to the corresponding cell in the Budget notebook, type

   ```
   [Budget]D:Z36
   ```

3. Choose OK.

The indicated block is selected.

You can also point out blocks and trace errors with GoTo. For instructions, see Extending Blocks with GoToand Tracing Errors with GoTo.

See Also

Mouse Techniques

Basic Editing Tasks

Extending Blocks with GoTo

To select a block with Edit|Goto or the Goto key, move to the upper left corner of the block, then choose Edit|Goto or press F5 and specify the lower right corner. Press Shift, then press Enter or click OK. The selector moves to that corner and selects the block defined by both corners.

See Also

Using the GoTo Command

Basic Editing Tasks

Tracing Errors with GoTo

To audit, or trace the source of, errors with the GoTo command or key, select a cell containing NA or ERR then press F5 or choose Edit|Goto. The original source of the error appears in the Reference edit field. Click OK or press Enter to select that cell and display its contents in the input line.

See Also

Displaying Error Sources in Cells

Using the GoTo Command

Basic Editing Tasks

Using the Spell Checker

To use the Spell Checker, available in the Workgroup edition of Quattro Pro, choose Tools|Spell Check. The Spell Checker SpeedBar appears.

To continue,

1. Select the block or graph to check and click Start.
2. Suspected misspellings appear in the Misspelled field. Suggested corrections, if any, appear in the Suggestion field. Click the arrow beside the Suggestion field to list all suggestions.
3. Decide how to handle each suspect word, then click an action button:


Skip leaves the current suspect word unchanged.


Skip All leaves the current suspect word unchanged and ignores all other occurrences of it in the selected block or graph.


Add leaves the current suspect word unchanged and adds it to the dictionary, so it won't be suspected again.


Change changes the suspect word to the current suggestion. If none of the suggested words is correct, type the correct spelling in the Suggestion field then click Change.


Change All changes all occurrences of the suspect word to the current suggestion, whether selected from the list or typed.

You can choose from several spell checker options. For details, see Spell Checker Options.

See Also
Basic Editing Tasks

Spell Checker Options

To verify and set Spell Checker options in the Workgroup edition of Quattro Pro, click the Options button. The Options dialog box appears.

You can:


Choose the language to check in.


Choose a custom dictionary; click Choose New Dictionary.


Remove the current custom dictionary; click Remove Current Dict.


Check any or all spelling options. When checked,

- Ignore Repeated Words skips double words like and and.
- Ignore Capitalization Errors skips words with capital letters in the wrong places: dOg.
- Ignore Acronym Errors skips acronyms and abbreviations that appear in the dictionary with different punctuation. For example, when this option is checked, ATM and A.T.M. are considered the same.
- Ignore Words with Numbers skips labels like M9020.
- Ignore UPPERCASE Words skips any word entered in all uppercase letters.

Settings are saved in your Windows directory in a text file called BORSPELL.INI.

See Also
Using the Spell Checker

Entering Data

These topics describe how to enter data into Quattro Pro notebooks:

Basic Data Entry

When you start Quattro Pro, a blank notebook appears, ready for data entry. To enter data in a cell, click the cell (or use the arrow keys to move the selector to the cell) and type the entry. The characters you type appear on the input line below the SpeedBar. Use Backspace to erase mistakes as you type.

When you click the check mark in the input line or press Enter or one of the arrow keys, Quattro Pro writes the data into the cell. If the entry is a formula, the result of the formula appears in the cell. See Types of Data for a description of what you can enter into a cell.

Quattro Pro displays an error message if it discovers a problem with an entry, such as an invalid character. After you click OK to clear the error message, you enter Edit mode, with the insertion point positioned at the problem character in the input line. Quattro Pro won't accept the entry until you correct the problem. To cancel the entry, click the X button or press Esc.

You can also calculate values before you enter them. Type the formula on the input line, but press F9 instead of Enter. For example, if you type 8*9 on the input line, then press F9, Quattro Pro replaces the formula with the result, 72. Press Enter or click the checkmark button to write the result into the cell.

Finally, you can drill an entry into several pages at once using Group mode. For information, see Grouping Notebook Pages.

See Also

Editing Entries

Entering Labels

Entering Numbers

Entering Formulas

Entering Dates and Times

Entering Data

Types of Data

There are two types of data in a notebook page: labels and values.


A label is a text entry such as "Total."


A value is a number, a date, a formula or a formula's result. Formulas usually refer to numbers in other cells to calculate a value, such as the difference between the values in two cells or the total of values in a column. You can also use @functions and numeric values in formulas.

Quattro Pro determines the data type by the first character you type. It replaces the READY indicator on the status line with LABEL for text or VALUE for numbers, dates, or formulas. After you enter the data, the READY indicator reappears.

See Also

Basic Data Entry

Entering Labels

Entering Numbers

Entering Formulas

Entering Dates and Times

Entering Data

Entering Labels

A label is a text entry. It can begin with any letter or punctuation mark except the following:

/ forward slash

+ plus symbol

- minus symbol, or hyphen

$ dollar sign

(open or left parenthesis

@ At symbol

pound or number symbol

. period

If you want to start a label with a number or with any of the above special characters, see the Aligning Labels subtopic below; other label topics follow.

Aligning Labels

Block Centering.

Entering Numbers in Labels

Entering Wide Labels

Repeating Characters in Labels

You can also force cells to accept only labels with the Data Entry Input property in the block Object Inspector (see Changing Block and Page Properties for more information).

Note: In addition to the characters on your keyboard, you can enter any ANSI character. This includes international characters such as Ä and Â, and special symbols such as ¶.

See Also

Basic Data Entry

Types of Data

Entering Data

Aligning Labels

When you enter a label, it is positioned within the cell according to the default label alignment, which is usually left-aligned. To align a label differently from the default alignment, precede the label with a label-prefix character.

Label-prefix characters don't appear in the cell itself, but they do appear on the input line when you select the cell.

Label-prefix character	Alignment
' (apostrophe)	Left-aligned
^ (caret)	Centered
" (quotation mark)	Right-aligned

To begin a label with one of the label-prefix characters, precede it with another label-prefix character indicating the alignment you want. For example, to enter the label

```
"Lefty" Grove
```

and left-align it, enter the label as follows:

```
'"Lefty" Grove
```

To change alignment for several labels at once, use the Alignment buttons in the SpeedBar or the Alignment property in the active block Object Inspector. To change the default label alignment setting for a given page to right-aligned or centered, use the page Alignment Property (see Setting Label Alignment for more information).

You can align text across several cells. For instructions, see Block Centering.

See Also

Aligning Cell Entries
Entering Numbers in Labels
Entering Wide Labels
Repeating Characters in Labels
Entering Data

Block Centering

It's easy to center text over a block of cells, for example, a table:

1. Type the text to be centered in the leftmost cell of the block to center across. If you're centering text in row 12 across columns C, D, and E, type the text to be centered in C12.
2. Highlight the block to center within, starting with the cell containing the text: C12..E12, in this example.
3. Click the Block Center button in the notebook SpeedBar.

The text moves to the center of the selected block. To display the centered text in the input line, click the cell where the text was originally entered. A caret (^) now appears before the text.

Another Method

Block centering is a block property, so you can follow these steps instead of clicking the Block Center button:

1. Select the block to center across as described above.
2. Right-click and choose Block Properties to display the active block Object Inspector.
3. Choose Alignment and check Center Across Block.

Preformatting For Centered Text

You can format blocks for centered text before entering the text. Highlight the block to center text across and click the Block Center button. Then, enter the text in the leftmost cell of the block. Type a caret (^) first, followed by the text.

Shading Formatted Blocks

If you've used block centering, shading applied to the leftmost cell appears in the rest of the block. For example, if cell C12 in the previous figure is shaded red, D12 and E12 also are red.

See Also

Aligning Labels

Aligning Cell Entries

Entering Data

Multiple Entries with Block Centering

If you enter text in a second cell within a center-formatted block, the first entry is centered over formatted cells to the left of the next entry. The position of the next entry depends on the alignment character typed before it, as shown in the next figure:

If ', the new text is left-aligned in its entry cell.

If ", the new text is right-aligned in its entry cell.

If ^, the new text is centered over the remaining blank cells in the formatted block.

See Also

Block Centering

Aligning Labels

Aligning Cell Entries

Entering Data

Entering Numbers in Labels

As long as a label begins with a letter or punctuation mark, you can include numbers with no special prefix. To enter a label that begins with a number, such as 145 Howard Ave., precede it with a label-prefix character. Otherwise, the entry is considered invalid (Quattro Pro initially identifies it as a value, but values cannot contain text).

Some numeric entries cause problems if you don't enter them as labels. For example, the phone number 555-1233 appears as -678 in the cell because Quattro Pro interprets it as a formula; it subtracts 1233 from 555.

If there are groups of cells where you want to enter phone numbers or zip codes, you can predefine those cells as labels with the block Data Entry Input property (see Limiting Data Types for details). You can then omit the label-prefix character when you enter phone numbers or zip codes.

See Also

Aligning Labels

Entering Wide Labels

Repeating Characters in Labels

Entering Data

Entering Wide Labels

When a label exceeds the width of the cell, the text spills into blank cells to the right. When the label runs into a cell that contains an entry, Quattro Pro truncates the displayed appearance of the label, but still stores all of it.

To see all of a truncated entry, select the cell. The entire label appears in the input line. Be sure to select the leftmost cell (the one that actually contains the entry). To fully display a truncated entry in its own cell, widen the column containing it (see Resizing Rows and Columns for details), or erase the cells to the right of it.

See Also

Aligning Labels

Entering Numbers in Labels

Repeating Characters in Labels

Entering Data

Repeating Characters in Labels

You can repeat one or more characters to fill a cell by preceding the character(s) with a backslash (\). For example, to fill a cell with hyphens, type

```
\-
```

To repeat a series of characters in a pattern, enter the first set of characters to be repeated after the backslash, such as \abc.

If you want a label to begin with a backslash, but you don't want to repeat the characters after it, precede the backslash with a label-prefix character.

See Also

Aligning Labels

Entering Numbers in Labels

Entering Wide Labels

Entering Data

Entering Numbers

When you begin typing a number, the word VALUE appears on the status line. After you complete the entry, the indicator changes to READY. By default, numbers appear right-aligned in the cell. To change alignment, use the Block Alignment property (see Aligning Cell Entries for details).

Keep these rules in mind when entering numbers:


A number entry can contain only numerals (0 to 9), a leading negative (-) sign, a leading positive (+) sign, a single decimal point, and an ending percent sign (%).


Don't use parentheses to indicate a negative number. Use a minus sign (-) instead. However, if you change the numeric format to Currency or Comma, negative numbers appear in parentheses (see Setting Numeric Format for information on numeric formats).


Don't include commas when entering numbers. You can display them by changing the numeric format.


Don't include spaces in the entry.


Don't substitute a lowercase l ("el") for 1 (one) or an uppercase O ("oh") for 0 (zero).

Note: If you want to surround negative numbers with parentheses to add commas, or to otherwise change the appearance of numbers, you can change the block Numeric Format property (see Setting Numeric Format for details).

You can use scientific notation (for example, 2.35E+8) to enter a number. If the calculated number fits in the cell (and the numeric format makes it possible), it appears in full. You can also automatically enter a sequence of numbers into a block (see Entering a Sequence of Numbers for more information).

Length

Like a label, a number can be longer than a cell is wide. If a number is wider than a cell, however, it doesn't spill into adjacent cells as labels do. Instead, it appears either in scientific notation or as a row of asterisks (*********), depending on the cell's numeric format and width. The full number appears when you widen the column (see Resizing Rows and Columns for details).

See Also

Entering Dates and Times

Entering Formulas

Entering Data

Entering Dates and Times

Another type of value entry is a date or a time, as long as it's entered in one of the legal date or time formats. To enter a date or time into a cell, hold down the Ctrl and Shift keys and press D, then enter the date or time in one of the following formats:

 DD-MMM-YY (04-Mar-92)

 DD-MMM (04-Mar, assumes current year)

 MMM-YY (Mar-92, assumes first day of the month)

 the Long International date format--the current application International property setting, ordinarily MM/DD/YY

 the Short International date format--the shortened version of the current application International property setting, ordinarily MM/DD

 HH:MM:SS AM/PM (01:42:30 PM)

 HH:MM AM/PM (01:42 PM)

 the Long International time format--the current application International property setting, ordinarily HH:MM:SS

 the Short International time format--the shortened version of the current application International property setting, ordinarily HH:MM

The four International date and time formats correspond to the International property settings in the application Object Inspector (see Setting International Options for details). If you change those settings, you must then conform to the new settings when you enter dates or times in one of the international formats.

Once you enter a date or time, you can change to another date or time format with the block Numeric Format property (see Setting Numeric Format for details).

If you forget to press Ctrl+Shift+D before entering the date or time, Quattro Pro interprets the date or time as a formula. Press F2 to edit the cell, then press Ctrl+Shift+D followed by Enter.

To force a block to accept only date or time entries, use the block Data Entry Input property (see Limiting Data Types for details).

Date Calculations

Dates and times are represented on the input line by serial numbers. For dates, the serial number is an integer (usually a five-digit number beginning with 3). The serial number for times is a decimal fraction between 0.000 and 0.99999.

Serial numbers let you use date or time values in formulas. For example, subtracting 10/1/92 from 10/8/92 results in 7. For more information on date and time serial numbers, see Date and Time

@Functions for more details.

See Also
Entering Numbers
Entering Formulas
Entering Data

Entering Formulas

Another way to enter a value is with a formula. Quattro Pro formulas are like algebraic formulas. They combine values and operators to calculate a single value. See the following subtopics for specific information on formulas:

Formula Basics

Using Values in Formulas

Using Operators in Formulas

Types of Operators

Using @Functions in Formulas

Using the @Functions Button

Using Blocks in Formulas

Pointing to Blocks

Pointing from Dialog Boxes

Using Noncontiguous Blocks

Using 3-D Blocks

Types of Formulas

Creating Totals

Note: To begin a formula with a letter, such as for a cell address or block name, precede it with a plus sign so the entry is considered a value, not a label.

See Also

@Functions

Using Arrays in Formulas

Entering Data

Formula Basics

You can use numbers, cell coordinates, or block names for the values in formulas. For example,

```
+A1 - 3
```

subtracts three from the value in cell A1.

The result of a formula appears in the cell. The formula appears on the input line when you select the cell and on the status line when you're editing the entry.

You can include spaces between operators and values, but Quattro Pro deletes them when you click the check mark button or press Enter to complete the formula. A formula must begin with one of the following characters:

```
0 1 2 3 4 5 6 7 8 9 . + - ( @ # $ =
```

By default, Quattro Pro calculates formulas when you enter them, and recalculates them each time you change the data involved. If a notebook contains many complex formulas, you may want to delay recalculation (see Setting Recalc Options for details) to save time.

See Also

Types of Formulas

@Functions

Entering Formulas

Using Values in Formulas

Values in a formula can be any of the following:

- Numbers (for example, 948, -84, 43.23).
- Coordinates of other cells or blocks (for example, B12, G29..G31, B:A3..D6, or [NOTEBK2]A:A1).
- Block names which are names you give to a cell or block (for example, EXPENSES). Among other things, block names make rereading formulas easier. See Maintaining Block Names for more information.
- @Functions (for example, @SUM(B1..B24)). Quattro Pro @functions are a set of standard formulas used to simplify complex calculations.
- Text surrounded by double quotation marks (for example, "PROFIT" or "Dear Mr.").

When you use a cell address, Quattro Pro refers to the values in the cells. For example, the formula +B6+C1 adds the values in those two cells and displays the result.

See Also

Using Operators in Formulas
Using @Functions in Formulas
Using Blocks in Formulas
Creating Totals
Types of Formulas
Entering Formulas

Using Operators in Formulas

Formulas use operators, such as **+**, **-**, *****, and **/**, to act on two or more values. Often formulas contain several operators, as in

```
+C5 - D12 + F24 * 0.123
```

The result of a formula depends on the order in which the arithmetic operations are performed. Quattro Pro assigns each operator a precedence and performs the operations in order of precedence. Multiplication has higher precedence than addition, therefore

```
5 + 1 * 3 = 8, not 18
```

Operations with equal precedence are performed from left to right.

The next table lists the operators and the precedence assigned to each. Operators with the highest precedence (7) are performed first.

Quattro Pro Operator Precedence

Operator	Description = Precedence
&	String combination = 1
#AND#, #OR#	Logical AND, logical OR = 1
#NOT#	Logical NOT = 2
=, <>	Equal, not equal = 3
<, >	Less than, greater than = 3
<,=	Less than or equal = 3
>,=	Greater than or equal = 3
-, +	Subtraction, addition = 4
*, /	Multiplication, division = 5
-, +	Negative, positive = 6
^	To the power of (exponentiation) = 7

To quickly remember the order of precedence of the common arithmetic operators, use the phrase "My Dear Aunt Sally" (multiplication, division, addition, and subtraction).

You can override operator precedence by including parentheses in formulas. Enclose in parentheses the portion of a formula you want calculated first. You can nest parentheses inside other parentheses; Quattro Pro calculates the innermost part first. For example,

```
4 * 2 + 3 = 11
4 * (2 + 3) = 20
(4*2) + (3 + 5) * 4 = 40
((4*2) + (3 + 5)) * 4 = 64
```

See Also

Input-Line Parenthesis Matching

Types of Operators

Using Values in Formulas

Using @Functions in Formulas

Using Blocks in Formulas

Creating Totals

Types of Formulas
Entering Formulas

Types of Operators

Arithmetic operators, such as **+** and ***** used in `+B3*1.3`, perform addition, subtraction, multiplication, division, and exponentiation. These are the most commonly used operators:

- + * / ^

The **&** (concatenation) operator is a text operator. You can use it to append text strings, whether each string is typed into the formula or pulled from a cell reference. For example, to concatenate a label from a cell to a string,

1. Type a plus sign (+) to begin the formula.

   Enter the first string by pointing to the cell containing it (the cell must contain a label).
2. Type an ampersand (&).
3. Enter the second string by surrounding it with quote marks. If you want a space between strings, be sure to type it within the quote marks.
4. Click the check mark or press Enter to complete the formula.

Logical operators determine whether expressions are true or false. For example, +A1<10 determines whether the value in A1 is less than 10. If the statement is true, the cell displays 1; if false, it displays 0.

These are logical operators:

`<  >  <=  >=  <>  =  #NOT#  #AND#  #OR#`

See Also

Using @Functions in Formulas

Types of Formulas

Entering Formulas

Using @Functions in Formulas

Quattro Pro provides many built-in functions, called @functions because they always begin with an @ sign. @Functions are special commands that perform particular calculations and return results. You enter them into spreadsheet cells, either alone or within formulas.

Quattro Pro has more than 100 @functions. Some of the most commonly used @functions are mathematical @functions like @SUM and @AVG. @SUM totals the contents of the blocks you reference, and @AVG counts the number of values in the referenced block and returns the average of the values.

@Functions have these parts:

- the name of the @function (such as @SUM, or @AVG)
- the arguments (the values, blocks, or text strings to be operated on)
- the commas that separate multiple arguments
- the parentheses around the arguments

All of these parts and the order in which they are used are called the syntax. Each @function has its own particular syntax.

Note: If you change the Punctuation setting of the application International property, other characters besides commas can be used to separate arguments (see Controlling Punctuation for details).

To see how @functions work, here are some examples:

- @SUM(A1..A4,B1) is equivalent to +A1+A2+A3+A4+B1
- @AVG(A5..A8) finds the average of the values in A5 through A8
- @ROUND(A9,2) rounds the value in A9 to two decimal places
- @ROUND(@AVG(A5..A8),3) rounds the average of the values in A5 through A8 to three decimal places

See Also

Input-Line Parenthesis Matching

Using the @Functions Button

@Function Help

@Functions

Types of Formulas

Entering Formulas

Using the @Functions Button

Quattro Pro provides the @Functions button to help you enter @functions. The @Functions button appears in the SpeedBar whenever you're entering or editing a cell entry.

To enter an @function into a cell,

1. Select the cell.
2. Type + or click in the input line to begin entering the formula.
3. Click the @Functions button or press Alt+F3.
4. Choose a category and double-click the @function you want from the list.
5. The @function name and a left parenthesis appear in the input line. The complete syntax of the @function appears in the status line. The first argument appears in uppercase.
6. Enter the value, block references, or text string as the first argument.
7. If the syntax of the @function you're entering requires more than one argument, type a comma. The next argument in the syntax in the status line turns to uppercase.
8. After you enter all remaining arguments, type a close parenthesis, and press Enter.

You can also enter @functions into cells by typing them; keep an eye on the status line to follow the correct syntax.

See Also

Using @Functions in Formulas

@Functions

Types of Formulas

Entering Formulas

Using Blocks in Formulas

A block is any rectangular group of cells (a block can also be a single cell). Many formulas operate on blocks containing more than one cell. For example, to total the values in a column, type `@SUM(B1..B7)`.

To specify the coordinates of a block, type the address of the top left cell, followed by one or two periods (..) and the address of the bottom right cell. For example, the block C3..D6 refers to cells C3, D3, C4, D4, C5, D5, C6, and D6. You can enter the addresses of any two cells in opposite corners in any order. Quattro Pro rewrites the block coordinates to list the top left cell followed by the bottom right cell.

When specifying a block, you can enter just one dot (period) instead of two; for example, F4..F11 and F4.F11 indicate the same block.

Note: However, you must enter two periods in block coordinates if you specified a period as the argument separator in the Punctuation setting of the application International property (see Controlling Punctuation for details).

To refer to a block in another page in the same notebook, type the page name and a colon before the block reference (for example, B:B4 refers to cell B4 in page B). For information on referring to a block that spans multiple pages, see Using 3-D Blocks.

Note: If you rename a page, you need to use the new name on its tab in the block reference instead of the page letter. Also, if you group pages, you can refer to them by their group name instead of by their individual page names.

To refer to a cell in a different notebook, type the file name in brackets and the page name before the block reference (for example, [SALES]C:B5 refers to cell B5 of page C in the SALES notebook file. See Creating Links for details).

You can also specify a block by entering the name you've assigned a block (see Block|Names for details).

See Also

Pointing to Blocks

Pointing from Dialog Boxes

Using Noncontiguous Blocks

Setting Display Options

Entering Formulas

Pointing to Blocks

The easiest way to enter single cell or block references into formulas is to point to them. If you click the cell, or move the selector to the cell you want to include, you enter Point mode.

For example, to enter the formula +B3*(B4-B5) into cell B6 by pointing,

1. Select cell B6.
2. Type + to begin the formula.
3. Click cell B3 or press Up until the selector is in B3. (The indicator changes to POINT.)
4. Type an asterisk (*) to indicate multiplication. Quattro Pro enters an asterisk after +B3 and returns the selector to B6.
5. Type an open parenthesis, (.
6. Select cell B4 and type a minus (-).
7. Select cell B5 and type a close parenthesis,).
8. Press Enter. Quattro Pro writes the formula into B6.

To enter Point mode, the insertion point must be after one of these characters:

 an arithmetic operator [+ - / * ^]

 a text operator [&]

 the last character of a logical operator [< > = #]

 an open parenthesis [(]

 a comma [,]

 a semicolon[;] (or whatever character you've chosen as an argument separator through the Punctuation setting of the application International property; see Controlling Punctuation for details)

You can also enter Point mode if a cell or block reference is selected in the input line. Otherwise, if you click another cell or press an arrow key, Quattro Pro enters the formula into the cell and returns to Ready mode.

Instead of only pointing to cells, you can point to blocks by dragging them.You can also point to blocks using the keyboard.

See Also

Using Blocks in Formulas

Pointing from Dialog Boxes

Entering Formulas

Pointing from Dialog Boxes

Point mode is useful from dialog box edit fields that require cell references. In the case of commands requiring cell references, you can make the dialog box temporarily collapsed to make pointing easier.

When you first choose a command, the active cells are highlighted in the primary edit field. When you click and drag to point to a block, you enter Point mode, and the dialog box is reduced to its title bar until you release the mouse button. The coordinates of the block you point to appear in the edit field.

You can also enter Point mode by pressing Up, Down, or F2, or by double-clicking the contents of the edit field. In these cases, the dialog box remains collapsed until you exit Point mode or until you expand it with the Minimize/Maximize button.

You can exit Point mode by pressing F2 (which enters the block in the edit field), or by pressing Esc (which restores the previous entry in the edit field).

See Also
Using Blocks in Formulas
Setting Display Options

Using Noncontiguous Blocks

Noncontiguous blocks are separate blocks that you operate on at the same time.You can write formulas that refer to many noncontiguous blocks at once, or you can act on noncontiguous blocks with commands.

To point to noncontiguous blocks, select the first block, hold down the Ctrl key, and select additional blocks. When every block you want to refer to or act on is highlighted, complete the formula or command.

To type references to noncontiguous blocks into a formula, separate each block with a comma, as shown in this example:

```
A2..A5,B7,D5..E12
```

Most commands can work on noncontiguous blocks.

See Also

Using Blocks in Formulas

Setting Display Options

Using 3-D Blocks

A 3-D block is the same two-dimensional block selected on a series of consecutive notebook pages. For example, the block A2..B5 on pages A through D is a 3-D block.

To point to a 3-D block, first select the "2-D" block on the first page of the series. Then hold down the Shift key while you click the tab for the last page in the series. A black line appears under the tabs. Complete the formula that refers to this block, or choose the command to act on it. This block remains selected only until you select elsewhere in the notebook.

To type a reference to a 3-D block, include the page references first, followed by the block coordinates. For example,

```
A..D:A2..B5
```

refers to a block on pages A, B, C, and D.

If the same pages were grouped under the name Receipts (see Grouping Notebook Pages for details), the same 3-D reference would look like this:

```
Receipts:A2..B5
```

Another way to type a reference to a 3-D block is to visualize it as a cube. Type the page and block coordinates of the cube's front upper left corner, followed by two periods, then the back lower right corner. For instance, you can refer to the same block as in the previous example by typing

```
A:A2..D:B5.
```

Either 3-D syntax is correct, but by default, all references you point to or type are converted to the first method. If you prefer the second method, you can switch to it with the application Display property.

See Also

Using Blocks in Formulas

Setting Display Options

Types of Formulas

Quattro Pro works with three different kinds of formulas:

Arithmetic formulas, such as `+B3+421.3`, calculate numeric values. These are the most commonly used formulas.

Text formulas are any formulas that have a textual result. For example, `+C4&" Review"` enters the text in cell C4 and adds a space and the word Review. Text formulas include those created with string @functions or an @IF function that results in a text string.

Logical formulas are true/false statements concerning values in other cells; for example, `+C3<10`. Quattro Pro checks the spreadsheet to see if the statement is true or not. If true, it enters a 1 in the cell; if false, it enters a 0.

Quattro Pro uses relative addressing to keep track of cells referenced in formulas.

See Also
@Functions and Formulas
Entering Formulas

Creating Totals

Quattro Pro includes SpeedSum, a way to quickly total columns and rows of numbers. The SpeedSum button on the SpeedBar uses @SUM to add the values of the specified block. You can use it to total a single row or column, multiple rows or columns, all the rows and columns in a block, or 3-D blocks.

To use SpeedSum,

1. Select a block that includes the data to total plus a blank cell (or cells) beneath or to the right to contain the results.
2. Click the SpeedSum button.

The entries added depend on what you select before clicking SpeedSum:


To total several cells in a column, select the data plus one blank cell below. For example, to total the values in the column A1..A3, select A1..A4. The total appears in cell A4.


To total several cells across a row, select the cells plus one blank cell to the right.


To total many columns, select the data and a blank row below.


To total many rows, select the data and a blank column to the right.


To total all rows and columns and create a grand total at the same time, select the block of data with a blank column to the right and a blank row below. Click SpeedSum Example for details.


To total all cells on multiple pages onto a blank page, select the 3-D block of data and the same block on an extra blank page.


To create totals outside all rows and columns on multiple pages, select a 3-D block with a blank row and column around the data.

You can also use SpeedSum without selecting the data you're totaling. This is useful for totaling large blocks of data that are time-consuming to select. Just select the blank cells where you want the totals placed (the blank cells must be adjacent to the data), and click SpeedSum. Quattro Pro guesses which cells you want totaled by the extent of the data above (if you selected cells in a row), to the left (if you selected cells in a column), or on previous pages (if you selected a block). After clicking SpeedSum, check each @SUM formula to make sure it actually refers to the appropriate cells.

Note: SpeedSum adds values only. Any label entries in the selected block are treated as zero values.

See Also
Entering Formulas

Adding Comments to Entries

You can attach hidden explanatory comments to value entries. Just type a semicolon (;) after the entry and type your comment. The semicolon and the characters after it are stored with the cell, but they don't appear in the cell. You can see the comment on the input line when you select the cell.

Hidden comments don't appear in a printed notebook unless you check Cell Formulas in the Options dialog box of File|Print.

To hide an entire entry, use the Hidden setting of the Numeric Format property in the block Object Inspector (see Setting Numeric Format for details).

See Also

Editing Entries

Entering Numbers

Entering Formulas

Entering Dates and Times

Entering Data

Entering a Sequence of Numbers

You can use the SpeedFill button to fill a block according to one or more seed values. Or, use Block| Fill to specify rules for filling the selected block. You can use numbers, dates, times, or even formulas.

To fill a block with sequential values using Block|Fill,

1. Select the block or blocks to fill. If you select a noncontiguous block, Quattro Pro fills it in the order selected.
2. Choose Block|Fill.
3. Enter a Start value for the first cell. You can enter a number, a formula, a date, or a time. If you enter a date or time, see instructions at the bottom of this screen.
4. Enter a Step value. This is the constant value to add (or multiply or use as an exponent) with the Start value or the previous value. If you want the values to decrease, enter a negative number and a Stop value lower than the Start value. The Step value can also be a formula.
5. The default stop value is 8191. Enter a different value if you want a higher or lower limit for the fill values.
6. Choose column or row order. The initial setting is Column, so the fill sequence moves down columns, starting with the leftmost column. Choose Row to fill across rows instead, starting with the top row.
7. Choose the type of series you'd like to govern the fill. For details on using date series, see Filling with Dates or Times.
8. If the Start value is a number or a formula, you can choose Linear (addition), Growth (multiplication), or Power (exponentiation). Linear adds the Step value to the previous value (defined at first to be the Start value). Growth multiplies the Step value by the previous value. Power uses the Step value as the exponent of the previous value.
9. Choose OK.

See Also
Entering Data

Filling with Dates or Times

To fill a block with dates or times, choose Block|Fill and enter the beginning date or time in the Start field. Use Ctrl+Shift+D as you would in a cell to enter a date or time. The date or time format you use is repeated in subsequent cells.

For the Stop value, enter a number larger than 50000, or if you want to stop at a particular date, enter it using Ctrl+Shift+D.

For the Series option, the fill operation works linearly (Step values are added), but you can add cells in units of years, months, weeks, weekdays (thereby skipping weekends), or in days, hours, minutes, or seconds.

For the Step value, enter the number of years, months, and so on to add. If you're filling with dates instead of times, only the integer part of the Start, Step, and Stop values are used; the fractional part is ignored.

For example, with a Start value of 6/20/92, a Step value of 2, a Stop value of 50000, and Series set to Day, the second cell in the series contains 6/22/92.

If the Start value is a number that isn't formatted as a date (for example, if you enter 33764 instead of 09-Jun-92), cells are filled with different date or time-related information depending on the Series setting. If you choose Month, only month names are entered. If you choose Year, 4-digit year values appear. Choosing Week, Weekday, or Day fills with days incremented by the appropriate interval, and choosing Hour, Minute, or Second displays times with the corresponding level of detail.

See Also

Entering a Sequence of Numbers
Using SpeedFill
Entering Dates and Times
Entering Data

Using SpeedFill

Based on the entries in one or more cells, you can quickly fill a block with entries that continue a sequence. You can fill a block with numbers, a combination of letters and numbers, or with specific types of labels, such as days of the week or months of the year.

To use SpeedFill,

1. Select the cell to act as a "seed" value for the sequence plus the blank cells to fill with entries. For details, see SpeedFill Seed Values.
2. Click the SpeedFill button.

You can extend a sequence in any direction, using these rules:

 You can have one seed value, or many.

 The blank cells to be filled must be contiguous with the seed value(s), extending down the column or across the row to the right.

 You can fill a 3-D or noncontiguous block, but each page or subblock must have its own seed value(s) for its own filled sequence.

 You can extend in two directions at once by selecting a block with seed values next to the upper left cell.

You can create a sequence of labels to enter with SpeedFill. For details, see Creating SpeedFill Lists.

See Also

Entering a Sequence of Numbers

Entering Dates and Times

Entering Formulas

Entering Data

SpeedFill Seed Values

These are the types of entries you can use as SpeedFill seed values, along with the resulting sequence:

Seed values(s)	Continued sequence
1st	2nd, 3rd, 4th, 5th...
Qtr 1	Qtr 2, Qtr 3, Qtr 4, Qtr 1...
1st Quarter	2nd Quarter, 3rd Quarter...
Jan	Feb, Mar...
January	February, March...
Mon	Tue, Wed, Thu...
Monday	Tuesday, Wednesday...
Week 1	Week 2, Week 3...
P1, P2, Total	P3, P4, Total, P5, P6, Total...
Jan 89, Feb 89	Mar 89, Apr 89...
100 Days	101 Days, 102 Days, 103 Days...
100 Days, 200 Days	300 Days, 400 Days, 500 Days...
1, 3, 5	7, 9, 11, 13...
1, 3, 6	8.333333, 10.83333, 13.33333...
04/05/92	04/06/92, 04/07/92...

The last example shows the seed value formatted as a date (it was entered using Ctrl+Shift+D). To create your own sequences for SpeedFill, see Creating SpeedFill Lists.

See Also

Using SpeedFill

Entering Dates and Times

Entering Formulas

Entering Data

Creating SpeedFill Lists

If you often type the same list of labels into a column or row, you can create a custom list to use with SpeedFill. To create a custom SpeedFill list:

1. Use a text editor or word processor to open this text file: C:\WINDOWS\QPW.INI.
2. Go to the bottom of the file, leave a blank line, then enter this line at the left margin: `[SpeedFill]`
3. On the next line, enter the SpeedFill list in this format:

   Name1=Item1,Item2,...,Last Item

   You may enter as many items in the list as you want. Items can contain blank spaces, but don't leave spaces between items.
4. If you have another list, enter it on the next line in the same format with a different name:

   Name2=Item1,Item2,...,Last Item

   For example, here are two SpeedFill lists:

   ```
   [SpeedFill]
   Department=Engineering,Quality Control,Accounting,Legal,Marketing
   Tasks=Analysis,Design,Testing,Marketing
   ```
5. Save the file as C:\WINDOWS\QPW.INI. Be sure to save it as unformatted text, its original file type.

To enter the list, enter one of the items in a list as a SpeedFill seed value and follow the instructions in Using SpeedFill. If you highlight more cells than there are items in a list, the list repeats to fill out space. If a seed value appears in more than one custom list, SpeedFill uses the first list that contains it.

If you include an item in a custom list, such as the name of a month, that Quattro Pro uses in a standard SpeedFill list, your list appears instead of the standard list. SpeedFill Seed Values discusses standard SpeedFill lists and seed values.

Using Files

These topics describe how to work with files in Quattro Pro:

File-Handling Options

Whether you're saving or loading a file, commands that operate on files require the name of a file to work on. When you choose one of these commands, you'll see a dialog box with the options described below.

When you first open the dialog box, it displays files in the startup directory. The startup directory is initially the directory from which you start Quattro Pro, but you can change it with the Directory option in the application Startup property.

If you save or load a file in a directory other than the startup directory, this last-used directory becomes the default directory. Then the next time you choose a command to save or load a file, the default directory initially appears in the dialog box. The last-used directory is always the default directory displayed until you restart Quattro Pro, when the startup directory again appears.

If the file name you want is already displayed, double-click it. To display file names in other drives or directories, use these controls:

Directories	Displays the startup directory below the directory or drive containing it and above its subdirectories. To change to a different directory, double-click it.
Drives	Lets you switch to another drive. Click the arrow at the right of the box and double-click the drive name you want to switch to.
File Types	Controls the file names that are listed above it. Initially, the dialog box is set to display all files with an extension beginning with .W, but you can choose another popular file type by clicking the arrow and choosing a type from the list. To choose a file type not in the Files Type listed, replace the current entry in the File name input box at the top of the dialog box.
File Name	Where you choose the file to be saved or loaded, either by typing it in the edit field or clicking it from the list below. You can also quickly choose files with the down arrow button next to the File Name edit field; it displays a list of the files you've used most recently with the current command. You can also control the types of files displayed using DOS wildcards, as explained later.
Protection Password	Lets you enter a password to protect your notebook from unauthorized use. This option is only available in the File|Save As dialog box. If you haven't set a password level in the notebook Object Inspector, entering a password here sets the level to High. For more information on password levels, see Assigning a Password to a File.

You can use DOS wildcard characters (* and ?) in any part of the File Name edit field to limit the files displayed. For instance, *.PRN lists all files with the extension .PRN, and BUDGET9?.WB1 lists all notebooks that begin with BUDGET9 followed by one character. An asterisk causes Quattro Pro to look for any number of characters in its position; a question mark stands for any single character in its position.

Note: If you're working on a network and the file you're trying to open or retrieve has already been opened by another user, you'll view a Read-only prompt. You can choose to open the file with read-only rights (able to save it under a different name) or cancel the operation.

See Also

File Menu

Using Files

Loading Files

Saving a notebook writes it to a file. You can redisplay it anytime by loading the file into a window. You can also load files created with other notebook programs; Quattro Pro translates them automatically.

These File menu commands open a window or load a file into a window:

New creates a new, blank notebook window, overlaying existing windows (see Creating a New File for details).

Open creates a new window and loads the file you specify into it, overlaying existing windows (see Opening a File for details).

Retrieve loads a file into the active window, replacing any data currently in the window (see Retrieving a File into a Window for details).

Another window-related command, File|Erase, clears any data and properties from the active notebook window, letting you begin a new file.

Note: Although Quattro Pro has other types of windows (graph windows and dialog box windows), notebook windows are the ones loaded, created, and saved with the File commands.

See Also

Closing Files

Saving Files

Using Files

Creating a New File

To create a new notebook window without putting away the active notebook, use File|New. Quattro Pro opens a blank window, overlaying existing windows. It names the window NOTEBK#.WB1, where # is the number of windows you've opened since starting Quattro Pro. You can change the name when you save the file.

You can have multiple windows open at a time, with different notebooks in each. Each new window overlays other open windows. You can reselect an open window by clicking any visible part of it, or by selecting it from the list at the bottom of the Window menu.

To open a new window and load a new file into it, use the File|Open command. To close a window, use the Windows Control menu or File Close. To close all open windows, choose File|Close All.

See Also

Using Windows

Using Files

Opening a File

To work with another existing notebook without putting away the active notebook window, use File|Open:

1. Choose File|Open.
2. Choose the file name you want.
3. If the file has a password, a dialog box appears with space for entering it. Type the password and choose OK.

If the password is incorrect, Quattro Pro displays an error message and cancels the Open command.

Automatically Opening a File

When you start Quattro Pro, a blank notebook appears. However, if you've specified an autoload file, this file is opened automatically. This feature lets you immediately display the notebook you use most often.

An autoload file is a notebook specified with the Autoload File option of the Startup property in the application Object Inspector (see Opening a File at Startup for more information). Initially, this default is QUATTRO.WB1, and any notebook with that name in the startup directory is opened automatically if no other is specified. You can change this default to any file you like.

If you use a template (a notebook you created with the characteristics you use most often), you can display it automatically instead of a blank notebook. For more information, see Saving Notebook Templates.

Opening from the Command Line

You can start Windows and Quattro Pro and open a notebook all at the same time.

To load a specific notebook, type WIN C:\QPW\QPW (depending on the directory in which you installed Quattro Pro), followed by a space and the name of the file you want to open. There is no need to type the notebook's file-name extension unless it differs from the default extension (which is .WB1), or if you're translating a file from another program. For example, to load the BUDGET.WB1 file from the current directory, type the following from the DOS prompt:

```
WIN C:\QPW\QPW BUDGET
```

To load a file from a directory other than the current directory, include the path name with the file name. For example,

```
WIN C:\QPW\QPW C:\COMPANY\BUDGET
```

If Quattro Pro can't find the directory or the file you specify, it opens a new notebook by that name.

You can also load a file and Quattro Pro at the same time using Program Manager's Run command. For example, in the Run dialog box, type

```
C:\QPW\QPW BUDGET
```

See Also

Using Windows

File-Handling Options

Using Workspaces

Translating Files

Using Files

Retrieving a File into a Window

To load a notebook into the active window, use File|Retrieve. This removes any data from the active window, then fills it with the specified notebook.

To retrieve a notebook into the active window,

1. Choose File|Retrieve.
2. If the active window contains unsaved data, a dialog box appears asking if you want to lose your changes. Choose No to go back and save changes, or Yes to continue.
3. Choose the file name you want, as described in File-Handling Options.
4. If the file has a password, a dialog box appears with space for entering it. Type the password and choose OK.

When a notebook is retrieved, it replaces existing data in the window. Also, the window takes on the size and position it had when you last saved the notebook. For example, if the window was in the left half of the desktop when you last saved the notebook, the window in which you retrieve the notebook moves to the left half of the desktop.

To retrieve a file created with a different program, include the file-name extension when you retrieve it (see Translating Files for details).

See Also

Using Quattro Pro Windows

Using Files

Saving Files

There are three commands that save notebooks to disk:

Save saves the notebook to the name under which you last saved it. If the notebook hasn't yet been saved, choosing Save opens the same dialog box used by Save As.

Save As saves the notebook under a new name you specify.

Save All saves all open notebooks.

If you're using Save with a notebook for the first time, or if you use Save As, the Save File dialog box appears:

1. Choose File|Save As.
2. Enter the file name you want, as described in File-Handling Options.
3. If the file name you enter already exists, Quattro Pro warns you and gives you these options:


Replace overwrites the existing file.


Backup copies the file that exists on disk, giving it a .BAK file-name extension, and creates a new file with the original's name. To load the backup file later, include the .BAK extension with that name. This .BAK file contains your data as you previously saved it.


Cancel stops the command and returns you to the notebook.

Quattro Pro saves the notebook, storing the data along with all its properties. Quattro Pro also saves any named blocks, graphs, slide shows, and dialog boxes you've created. You can also protect the file with a password (see Assigning a Password to a File for details), or save the file as a template (see Saving Notebook Templates for details).

After you name a notebook, you can store changes to it with File|Save. Quattro Pro assumes you want to save the file under the same name.

File|Save All saves all open notebooks. For each file, you have the same Replace, Backup, or Cancel options as with the Save As command.

To save a notebook for use with another program, such as Paradox, dBASE, Reflex, or 1-2-3, include the program's data file extension when you save the notebook (see Translating Files for more information).

Like all legal DOS file names, Quattro Pro file names can be up to eight characters long and can consist of both letters and numbers. You can enter file names using either uppercase or lowercase letters. You can't use spaces in a file name. However, you can use the underscore character to simulate spaces; for example, 92_SALES.

Caution: If you assign a name longer than eight characters, Quattro Pro shortens it to the first eight characters.

Unless you're saving a file for use with another program, it's best if you don't include an extension with the file name. Quattro Pro then adds the default extension, initially .WB1. (To change the default extension, see Setting the File Extension for more information.)

See Also

Using Quattro Pro Windows

Using Files

Assigning a Password to a File

A notebook property lets you set security limits so only password holders can view formulas, show hidden notebooks, or open an entire notebook.

To apply password protection,

1. Right-click the notebook title bar to display the active notebook Object Inspector. Then choose Password Level.
2. Check Low, Medium, or High and click OK. The box at the bottom of the Object Inspector pane summarizes each option. See Protection Options for details.
3. The Enter Password dialog box appears. Type a password and click OK. You're prompted to type the password again for verification.
4. When you click OK, Quattro Pro compares both entries and only accepts the password if both are identical.

   Passwords are case-sensitive. For example, if the second password entry has a lowercase letter where the first entry had a capital, Quattro Pro considers these different passwords.

Caution! Be sure to write down the password and keep it in a safe place. If you lose it, you're locked out of part or all of the protected notebook.

Once you save and close a notebook with Low or Medium password protection, Password Level is no longer available in the notebook Object Inspector. Then, only password holders can change or remove this protection. For access information, see Using a Password at Startup.

See Also

Using Files

Protection Options

The Password Protection Level pane offers these Password Protection Level options:

None	Requires no password to open and use the notebook.
Low	Requires a password to view formulas in the notebook.
Medium	Requires a password to show the notebook and view formulas in it.
High	Requires a password to open the notebook. This option is suitable for most purposes.

See Also

Assigning a Password to a File

Low Protection

If Low is checked, anyone can open the notebook, but only password holders can view and edit formulas (see Using a Password at Startup for details). Other users see values instead of formulas. Selected formula cells show asterisks (*****) in the input line.

As usual, entering data into formula cells overwrites the original formulas. If you plan to assign Low or Medium protection to a notebook, it's wise to protect formula cells first or shade them so they're easily distinguishable from data cells.

See Also
Protection Options

Medium Protection

If Password Protection Level is set to Medium, the notebook is always hidden once it is saved and closed. Then, only password holders can show it again. All the features of Low protection apply as well. See Using a Password at Startup for information on showing hidden notebooks with Medium protection.

See Also
Protection Options

High Protection

With Password Protection Level set to High, only password holders can open the notebook. You're prompted for the password following any command that attempts to load the protected file. Remember, you must enter the assigned password in the same combination of lowercase and capital letters as the original entry. You can also open the notebook with a startup command (see Using a Password at Startup for details).

Once the notebook is open, you can hide and show it or edit its formulas without restriction. You can also change the password protection level.

See Also
Protection Options

Using a Password At Startup

If you need to edit formulas or password settings in a notebook with Low or Medium password protection, exit Quattro Pro, then start it again with a special startup command. Choose File|Run in the Program Manager, then enter this in the Command Line edit field and click OK:

path\QPW *filename* /S*password*

For example, if Quattro Pro is in C:\QPW5, you could enter this command line to open MYFILE.WB1 in subdirectory DATA with password sesame:

```
C:\QPW5\QPW C:\QPW5\DATA\MYFILE.WB1 /Ssesame
```

If MYFILE.WB1 had Medium password protection, you could choose Window|Show to display it again.

See Also
Assigning a Password to a File

Saving Notebook Templates

After you've worked with Quattro Pro for a while, you may find that you repeatedly use certain settings, formats, or structures in your notebooks. In these cases, you can reduce the time required to set up standard notebooks by creating notebook templates.

A notebook template is like a structural skeleton you can use as a basis for other notebooks. If you've already set up one notebook with the formatting properties and customized structure you prefer, just erase any nonstandard data you've entered (with Edit|Clear Contents) and save the file under a different name (with File|Save As). When you load the template, much of your initial work will be done for you. Just add new data and save the file under yet another name.

If you want to use the template most of the time, you can load the template every time you start Quattro Pro with the Autoload File option of the application Startup property; see Opening a File at Startup for details.

See Also

Saving Files

Clear Contents

File-Handling Options

Translating Files

Password Protecting a File

Saving Workspaces (Window and File Setups)

Closing Files

When you close a file, you remove the notebook, all its associated graph and dialog windows, and all its views (created with Window|New View) from the desktop. To close files,

1. Choose File|Close (for one file) or File|Close All (for all open files).
2. If a notebook has unsaved changes, a dialog box appears asking you if you want to save your changes. To save the file, choose Yes; choosing No closes the file without saving it.
3. Quattro Pro closes the window, revealing any windows underneath.

To close all windows and exit Quattro Pro at the same time, choose File|Exit.

To close a particular window (without closing related views), choose Close from the window's Control menu.

See Also

File|Close

File|Close All

File|Exit

Saving Files

Using Files

Using Workspaces

The arrangement of windows onscreen is called a workspace. It includes the position and size of all notebook windows and the names of the files contained in each window. The positions of graph and dialog windows are not saved as part of a workspace.

To save a workspace,

1. Choose File|Workspace|Save.
2. Enter the file name you want, as described in File-Handling Options. Don't include a file-name extension; Quattro Pro includes the .WBS extension for workspace files.

Caution: Saving a workspace doesn't save the contents of the files within it; use File|Save, Save As, or Save All for this. Also, if you use File|Save As, you must use File|Workspace|Save afterward so that Quattro Pro can load the correct file the next time you try to retrieve the workspace.

To retrieve a workspace,

1. Choose File|Workspace|Restore.
2. Choose the file you want from the list, or type in a name and choose OK.

Quattro Pro overlays any existing windows with the windows stored in the workspace file, then retrieves the appropriate file for each.

Quattro Pro always retrieves the latest saved version of files when you retrieve a workspace. If you leave the workspace and save a file included in the workspace, Quattro Pro retrieves the updated version of the file the next time you choose File|Workspace|Restore.

See Also

Saving Files

Using Files

Browsing in the Consolidator

Use this dialog box to choose a block in another file to use in a consolidation.

If the file name you want is already displayed, double-click it. To display file names in other drives or directories, use these controls:

Directories	Displays the startup directory below the directory or drive containing it and above its subdirectories. To change to a different directory, double-click it.
Drives	Lets you switch to another drive. Click the arrow at the right of the box and double-click the drive name you want to switch to.
File Types	Controls the file names that are listed above it. Initially, the dialog box is set to display all files with an extension beginning with .W, but you can choose another popular file type by clicking the arrow and choosing a type from the list. To choose a file type not in the Files Type listed, replace the current entry in the File name input box at the top of the dialog box.
File Name	The file containing the source block you want to use in the consolidation. You can control the types of files displayed using DOS wildcards, as explained later.
Block	The source block for the consolidation.

You can use DOS wildcard characters (* and ?) in any part of the File Name edit field to limit the files displayed. An asterisk causes Quattro Pro to look for any number of characters in its position; a question mark stands for any single character in its position.

Note: If you're working on a network and the file you're trying to open or retrieve has already been opened by another user, you'll view a Read-only prompt. You can choose to open the file with read-only rights (able to save it under a different name) or cancel the operation.

See Also
Using the Consolidator

Global Properties

Global properties include the properties that affect the entire Quattro Pro application and properties associated with each notebook.

Changing Global Properties

Application Properties

Setting Display Options

Setting International Options

Setting Startup Options (and Enabling Undo)

Setting Macro and Menu Options

Selecting Secondary SpeedBars

Setting Drag and Drop Delay Time

Active Notebook Properties

Setting Recalc Options

Setting the Zoom Factor

Changing the Notebook Color Palette

Hiding Parts of the Notebook Display

Making a Notebook a Macro Library

Assigning a Password to a File

Creating System Notebooks

Changing Global Properties

Global properties include the properties that affect the entire Quattro Pro application and properties associated with each notebook.

Application properties (those of Quattro Pro itself) include various display, international, startup, and macro options. You can also specify your own customized SpeedBar.

Use notebook properties to set recalculation options, reduce or enlarge notebook display, change the palette of default colors, control the display of parts of a notebook, or make a notebook a macro library.

You can change the Quattro Pro application or notebook properties by right-clicking or by choosing commands in the Property menu.

To change Quattro Pro application properties, do one of the following:


Right-click the application title bar.


Choose Property|Application.

To change notebook properties, first select the notebook by choosing it from the Window menu or by clicking any visible part of the window. Then do one of the following:


Right-click the notebook title bar.


Choose Property|Active Notebook.

Setting Display Options

The Display property sets clock display options, hides portions of the Quattro Pro window, and switches between page range syntax schemes.

To change a Display property setting, right-click the Quattro Pro title bar (or choose Property| Application) then choose Display. Change the settings you prefer and choose OK.

Setting Clock Display

Hiding Parts of the Window

Switching the Page Range Syntax

Setting Clock Display

The Clock option displays the date and time on the status line in your choice of formats.

To display the date and time, right-click the Quattro Pro title bar, choose Display, then choose Standard or International.

Standard displays the date and time in standard format (DD-MMM-YY and HH:MM AM/PM); International displays date and time in the Long International formats specified with the International property.

If the time or date displayed is incorrect, you must update your computer's internal clock. You can do this with the Windows Control Panel.

See Also
International Property

Hiding Parts of the Window

You can hide the SpeedBar, the input line, or the status line to allow more space in the Quattro Pro window.

To hide one of these elements, right-click the Quattro Pro title bar, choose Display, then uncheck Show SpeedBar, Show Input Line, or Show Status Line. After you choose OK, the parts you chose disappear.

See Also
Notebook Display Property

Switching Page Range Syntax

Quattro Pro provides two ways to refer to a 3-D block in a formula. The default method uses a unique page notation syntax for cell references The page references are expressed first, followed by a colon and the block coordinates. For instance,

A..C:B4..D9

refers to the block B4..D9 on pages A, B, and C.

An alternate syntax is available that refers to each corner of the block with the page reference included. For instance,

A:B4..C:D9

refers to the same block in the alternate syntax.

The default syntax makes group references more concise. For example, a reference to 1stQtr:A1..A10 in the default syntax would switch to January:A1..March:A10 in the alternate syntax.

You can always type a 3-D reference in either syntax, but Quattro Pro displays the reference according to the setting in the 3-D syntax option.

To switch to the alternate syntax, right-click the Quattro Pro title bar, choose Display, then choose the A..B:A1..B2 option. After you click to select a new cell, all existing 3-D references in formulas in open notebooks switch to the new syntax.

See Also
Grouping Notebook Pages

Setting International Options

The International property controls the appearance of currency, punctuation, dates, times, and international characters. It also lets you choose international sort orders. These options affect the appearance of values, how the date and time on the status line appear, how you enter arguments in @functions, and how certain international characters appear.

The initial defaults are standard for the United States. If you are using Quattro Pro in another country, or are doing business with another country, you can change these settings to suit your requirements.

To change an International property setting, right-click the Quattro Pro title bar (or choose Property| Application) and choose International. Change the settings you prefer and choose OK.

Choosing a Currency Symbol

Controlling Punctuation

Setting Date Formats

Setting Time Formats

Changing the Sort Order

Converting LICS Characters

Choosing a Currency Symbol

When you set the block Numeric Format to Currency, Quattro Pro initially displays currency values as specified in the International Currency Format in the Windows Control Panel. You can change to a different monetary symbol with the International property's Currency option. You can position the symbol after or before the currency value.

You can also specify whether negative numbers will be indicated by preceding them with a minus sign or by enclosing them in parentheses.

To change the currency or negative symbol,

1. Right-click the Quattro Pro title bar and choose International. Then choose the Currency option.
2. Choose QuattroPro/Windows.
3. Replace the contents of the Currency Symbol edit field with any character or character combination, including special ANSI characters. To enter an ANSI character not on your keyboard, hold down Alt and use the numeric keypad to type 0 and then the ANSI number for that character. For example, to enter a pound sign (£), press Alt and type 0163.
4. Choose Prefix to display the symbol before the value (as in $100), or choose Suffix to display the symbol after the value (as in 500F).
5. For the Negative Values option, choose Signed to precede negative values with a minus sign, or choose Parentheses to surround them with parentheses.

Controlling Punctuation

Punctuation settings specify the characters used to

- separate thousands in numbers in numeric display (usually a comma in North America)
- designate a decimal separator in numbers (usually a period in North America)
- separate arguments in @function statements and macro commands (usually a comma or semicolon)

By default, Quattro Pro uses commas as thousands separators, a period as a decimal separator, and a comma as an argument separator.

You can change to another combination of settings, or to Windows Default, which uses the International Number Format settings in the Windows Control Panel.

To change the setting, right-click the Quattro Pro title bar, choose International, and choose Punctuation. Then choose one of the alternate punctuation settings listed.

These options show the punctuation marks used to mark thousands and the decimal place, followed by the punctuation mark used to separate arguments in @functions and macros (a1,a2). The last four options specify that a blank space separates thousands in numbers.

Note: Regardless of the punctuation setting, semicolons are always accepted as argument separators.

Setting Date Formats

The International property's Date Format option doesn't directly determine how dates appear. Rather, it determines the international date formats given as options for date appearance. To set the actual appearance of dates, use the block Numeric Format property.

By default, Quattro Pro displays dates in long form as MM/DD/YY (06/18/92, for example) and in short form as MM/DD (06/18, for example).

You can change to another long and short form combination, or to Windows Default, which uses the International Date Format settings in the Windows Control Panel.

To change the setting, right-click the Quattro Pro title bar and choose International. Then choose Date Format, and choose one of the alternate formats.

Each of these options is shown in long and short versions. The short form is shown in parentheses. The format you choose here appears (in both long and short form) as a choice in the block Numeric Format property. Quattro Pro uses the long version of the format as the international clock setting (see Setting Clock Display for more information).

See Also
Block Numeric Format

Setting Time Formats

The International property's Time Format option determines the international time formats given as options for time display. To set the actual appearance of times, use the block Numeric Format property.

By default, Quattro Pro displays times in long form as HH:MM:SS (09:32:30, for example) and in short form as HH:MM (09:32, for example).

You can change to another long and short form combination, or you can choose Windows Default, which uses the International Time Format settings in the Windows Control Panel.

To change the setting, right-click the Quattro Pro title bar and choose International. Then choose Time Format, and choose one of the alternate formats.

Each of these options is shown in its long and short versions. The format you choose here appears (in both long and short form) as a choice in the block Numeric Format property. Quattro Pro uses the long version of the format as the international clock setting (see Setting Clock Display for more information). All international time settings use 24-hour formats (00:00 to 23:59).

See Also
Block Numeric Format

Changing the Sort Order

If you use data written in languages other than American English, you can tailor Quattro Pro's sort order to match that language.

By default, the Language setting is Windows Default, which makes Quattro Pro use sort rules as specified by the International Language setting (or, in the case of Windows 3.0, the International Country setting) in the Windows Control Panel.

To change sort order to that of a different language:

1. Right-click the Quattro Pro title bar, and choose International.
2. Then choose Language.
3. Choose Quattro Pro/Windows and choose one of the languages from the list.

Note: Quattro Pro may sort international characters in a slightly different order than products such as Paradox, dBASE, and Database Desktop do. Windows supports a richer character set than DOS products, so the results of sort operations in these products may be slightly different.

See Also
Sorting Data

Converting LICS Characters

Quattro Pro can translate LICS characters used in 1-2-3 .WK1 spreadsheets.

To convert these characters into standard ANSI characters, right-click the Quattro Pro title bar, and choose International. Then check LICS Conversion.

When you save the file, Quattro Pro writes these characters back to the LICS equivalents.

LICS characters are identical to the standard ANSI set except in the range 128 through 255, which is usually used for international and graphics characters. For more on LICS characters, see the 1-2-3 documentation.

See Also
Translating Files

Setting Startup Options

The Startup property includes information that is used every time you start Quattro Pro.

To change a Startup property setting, right-click the Quattro Pro title bar (or choose Property| Application), then choose Startup. Change the settings you prefer and choose OK.

Setting the Startup Directory

Opening a File at Startup

Running a Macro Automatically

Setting the File Extension

Setting the Computer's Beep

Enabling the Undo Command

Changing Key Compatibility

Setting Cell Selector Behavior

See Also
File Handling Options

Setting the Startup Directory

The startup directory is the drive and directory initially displayed when you use any File command at the beginning of a Quattro Pro session. You should make this the directory where you keep most of your notebook files.

The first time you start Quattro Pro, the drive and directory from which you started Quattro Pro indicate the startup directory.

To specify a new startup directory, right-click the Quattro Pro title bar, and choose Startup. Then in the Directory edit field enter the path, including the drive letter, of the new startup directory.

The startup directory is the one initially displayed in file-handling dialog boxes only until you save or load a file in another directory. Then, the last-used directory becomes the default directory for the duration of the Quattro Pro session.

See Also
File Handling Options

Opening a File at Startup

Each time you start Quattro Pro it looks for a file named in the Autoload File option, and loads it if it exists. By default, the name of this autoload file is QUATTRO.WB1.

To make another file open automatically instead, right-click the Quattro Pro title bar, and choose Startup. Then enter the new file name for the Autoload File option. If the file isn't in the startup directory specified in the Directory option (just above Autoload File), include the full path name of the file.

You can also automatically open a notebook file if you start Quattro Pro with the Run command from the Windows Program Manager. (See Opening a File for details.)

See Also
Running a Macro Automatically

Running a Macro Automatically

Every time you load a notebook, Quattro Pro looks in it for a macro named _NBSTARTMACRO and runs it, if found. If that macro isn't found, Quattro Pro looks for the name entered in the Startup Macro edit field in the application Object Inspector. If it exists, that macro runs immediately. By default, \0 is the startup macro.

To enter a different name in the Startup Macro edit field, right-click the Quattro Pro title bar and choose Startup. Then enter the new name for the Startup Macro option.

To make the startup macro run every time you start Quattro Pro, include it in the notebook named in the Autoload File option.

You can also create macros that run when you close a notebook or start Quattro Pro. For details, see Startup and Exit Macros.

See Also
Opening a File at Startup

Setting the File Extension

If you don't specify a file-name extension when saving or accessing files, .WB1 is used.

To specify a different default extension, right-click the Quattro Pro title bar, and choose Startup. Then enter one to three letters for the File Extension option.

See Also
File Handling Options

Setting the Computer's Beep

Normally, when you make an error (for example, if you misspell an @function name when entering a formula), Quattro Pro beeps.

If you don't want to hear the beep, right-click the Quattro Pro title bar, and choose Startup. Then uncheck the Beep option of the Startup property.

Enabling the Undo Command

When you want to reverse the effects of the last operation you perform, you can choose Edit|Undo. Most operations are reversed immediately.

You can always undo the following actions whether Undo is enabled or not:

- Entering data into a cell.
- Changing any properties except Block or Page property settings.
- Moving and resizing floating objects (floating graphs, SpeedButtons, floating OLE objects, floating bitmaps, or floating pictures).
- Changes to the Move, Align, Resize, ResizeToSame, and Create controls in graphic elements or dialog boxes you create.
- Changes to a graph type or graph title.
- Importing or exporting bitmaps into either a floating object or a graphic element.

To enable Undo for all other types of undoable actions, right-click the Quattro Pro title bar, and choose Startup. Then choose Undo Enabled.

See Also
Undoing Mistakes

Changing Key Compatibility

By default, certain keys work in ways that take the most advantage of Windows capabilities. For example:

The Up and Down keys move the insertion point between lines in a multiline entry (instead of completing the entry and moving the selector to the adjacent cell).

The Ctrl+Left and Ctrl+Right keys move the insertion point to the beginning of the previous or next word, respectively (instead of moving the insertion point five characters to the left or right).

In selected blocks, Enter moves the selector to the next row and Tab moves it to the next column; Shift+Enter moves up a row, while Shift+Tab moves back one column.

With one cell selected, Tab and Shift+Tab work like the Left and Right keys.

After you display the list of @functions (with the @Functions button), macros (with the Macros button), or block names (with F3), you can type a letter to highlight the first item in the list beginning with that letter (instead of having that item entered immediately in the input line). For example, in the @functions list, typing F highlights @FALSE, and typing V highlights @VALUE.

By changing the Compatible Keys setting, you can edit entries in the input line using keys described in Editing Entries and Edit Mode Keys. These keys are compatible with Quattro Pro for DOS. To produce this compatibility, right-click the Quattro Pro title bar and choose Startup. Then check Compatible Keys.

See Also

Setting Cell Selector Behavior

Setting Cell Selector Behavior

Normally, the selector doesn't move when you enter data from the input line into a single cell. If you're entering data in one column at a time, you can set Quattro Pro so the selector moves down a row each time you press Enter. To set Quattro Pro for columnar data entry, right-click the Quattro Pro title bar and choose Startup. Then, check Move Cell Selector on Enter Key. Before you can change this setting, Compatible Keys (the setting above it) must be unchecked.

See Also
Changing Key Compatibility

Setting Macro and Menu Options

Macro property settings let you control what the screen displays while you run macros. They also make alternate menu systems available with the slash key.

To change a Macro property setting, right-click the Quattro Pro title bar (or choose Property| Application) then choose Macro. Change the settings you prefer and choose OK.

Macro Suppress-Redraw

The Macro Suppress-Redraw settings control which parts of the screen redraw while macros are running. The default setting, Both, suppresses redrawing of both notebook windows and panels (menus and dialog boxes), including the status line and the input line, until the macro is finished. This speeds up macros, since Quattro Pro doesn't have to stop and draw each of these items used in the macro.

To change screen redrawing, right-click the Quattro Pro title bar and choose Macro. In the Macro Suppress-Redraw dialog box, choose one of these options:

Panel suppresses redrawing of menus, dialog boxes, the status line, and the input line.

Window suppresses redrawing of windows only.

None redraws everything during macros.

Using Alternate Menu Systems

Instead of using commands from the menu bar, you can run Quattro Pro for Windows using alternate menu systems. The Slash Menu setting controls which menu system displays when you press the slash key (/) from Ready mode.

To use one of the alternate menu systems, right-click the Quattro Pro title bar, choose Macro, and choose the menu system name from Slash Menu options.

After you click OK, you can press the slash key to open menus in the chosen menu system. Then choose the commands that appear. You can also run Quattro Pro for DOS macros that use keystrokes instead of menu equivalents.

You can also replace the menu bar with your own custom menu system. See Application Building for information.

See Also

Quattro Pro for DOS Macros

DOS Spreadsheet Users

Macros

Selecting Secondary SpeedBars

As explained in SpeedBar Designer and Application Building, you can create your own custom secondary SpeedBars for use in the notebook window.

To use your own SpeedBar, display it with the SpeedBar Control menu. You can display as many secondary SpeedBars, standard and custom, as you have room for.

To remove a custom SpeedBar, display the SpeedBar Control Menu again and choose Remove, or click the red X at the right end of the SpeedBar.

To hide the default SpeedBar, uncheck the Show SpeedBar option in the Display property of the application Object Inspector.

To display the current arrangement of SpeedBars as the default, right-click the application title bar to display the application Object Inspector. Choose SpeedBars and check Save On Exit.

The next time you start Quattro Pro, the same SpeedBars will appear at the top of the notebook window.

Setting Drag and Drop Delay Time

Quattro Pro's Drag and Drop feature lets you move and copy blocks with the mouse. You select a block, wait for a brief period of time, then drag the block to move it or press Ctrl, then drag to copy it. To change the waiting period,

1. Right-click the Quattro Pro application title bar to display the Application Object Inspector.
2. Choose Delay Time.
3. Enter a delay time, in milliseconds, into the Delay Time edit field. The default time is 500 ms.
4. Click OK to use the new setting.

See Also
Dragging and Dropping

Setting Recalculation Options

Recalculation refers to how Quattro Pro updates formula results when you change values those formulas depend on. If you need to change the way recalculation works, right-click the notebook title bar. Recalc Settings is already selected.

Setting Recalculation Mode

Setting Recalculation Order

Limiting Recalculation Iterations

Compiling Formulas to save memory and boost recalculation speed for certain types of notebooks.

Displaying Error Sources in Cells to display the source of ERR and NA values in cells.

Setting Recalculation Mode

Recalculation time is minimized by Quattro Pro's method of intelligent recalculation, which recalculates only those formulas that need to be recalculated.

Quattro Pro's default recalculation mode is Background, which recalculates formulas between keystrokes. It doesn't interrupt your work and recalculates automatically. Quattro Pro always finishes recalculation before saving, extracting, or printing the notebook.

To choose one of the other two modes of recalculation, right-click the notebook title bar and choose Recalc Settings. Then choose one of these options:


Automatic, which also recalculates formulas automatically, but pauses until recalculation is finished. Use Automatic if you have complex formulas that Background calculation doesn't handle fast enough.


Manual, which recalculates the formulas you enter or edit, but doesn't recalculate the entire notebook until you press F9. For example, if you enter +B3-C16 in a cell, Quattro Pro displays the resulting value, but the value isn't updated if the values in B3 or C16 change. If a formula needs to be recalculated, CALC appears on the status line to remind you that the displayed results of formulas may not be accurate. Press F9 to recalculate all formulas.

Either Background or Automatic is the recommended mode; both ensure that your data is always accurate. Manual mode is appropriate as a temporary break from recalculation while you adjust crucial cells in large notebooks that take a long time to recalculate.

In Manual mode, floating graphs don't adjust automatically; press F9 to replot after supporting values change. To recalculate a single cell in Manual mode, select the cell, press F2, then press Enter. (This works only if the formula isn't dependent on other formulas.)

In Background mode, when you want to pause your work in the notebook to finish recalculation in the background, press F9.

See Also

Setting Recalculation Order

Limiting Recalculation Iterations

Setting Recalculation Order

By default, Quattro Pro recalculates data in Natural order. This means that cells are calculated in dependency order. Formulas without dependencies are calculated first, followed by formulas that depend on them.

To change recalculation order, right-click the notebook title bar and choose Recalc Settings. Then choose one of these options:


Column-wise starts recalculation in cell A1 of the first page and proceeds down column A, ignoring formulas in other columns. When A is finished, recalculation continues in B1, and on down column B. This continues to the end of the page (column IV), and on to the last page in the notebook.


Row-wise also starts recalculation in cell A1, but proceeds by rows, starting at 1 and continuing through row 8192, through to the last page.

Note: If you specify Column-wise or Row-wise order of recalculation, you should set the number of iterations to at least two (see Limiting Recalculation Iterations). Otherwise, the notebook may be inaccurate.

Column-wise and Row-wise recalculation are provided for special purposes, such as for a series of formulas that create circular references to solve a problem or attain a goal. If you're not trying to solve problems such as these, use Natural recalculation order.

See Also
Setting Recalculation Mode

Limiting Recalculation Iterations

Elaborate formulas, such as those involving complicated engineering problems, require multiple evaluations to attain an acceptable degree of accuracy. These formulas are deliberately constructed to contain circular references, references that eventually refer back to the original formula. For such applications, you can set the number of iterations, or cycles of recalculation, to be performed when the notebook is recalculated.

To change number of iterations, right-click the notebook title bar and choose Recalc Settings. Then enter any number up to 255 for the # of Iterations option.

If the notebook contains no circular references and recalculation order is set to Natural, the iteration count is ignored.

Note: Before constructing elaborate formulas requiring limited iterations, see if Tools|Optimizer or Solve For can find solutions to your problems.

See Also

Goal-seeking with Solve For

Finding Solutions with Optimizer

Setting Recalculation Mode

Setting Recalculation Order

Creating Formulas

Compiling Formulas

If you work with large notebooks containing many numerical formulas, using compiled formulas can often boost recalculation speed. For maximum performance gains, you need a system with a numeric coprocessor (387 or 486) and an extra megabyte of RAM available for calculations.

To use compiled formulas, right-click the notebook title bar to display the active notebook Object Inspector. Choose Recalc Settings and check Compile Formulas, near the bottom of the window.

With this setting checked, recalculations can be slower initially while formulas are compiled; they can be up to 100 to 200 percent faster afterwards.

If you're uncertain about using compiled formulas, choose Help|Experts|Performance or click the Experts button, then click Performance Expert. The Performance Expert analyzes your hardware and how you use spreadsheet notebooks, then recommends a choice.

See Also

Experts

Displaying Error Sources in Cells

A new option in the active notebook Object Inspector highlights the source of error for each cell containing NA or ERR. To use it, right-click the notebook title bar and check Audit Errors, near the bottom of the Recalc Settings window. Click OK or press Enter to return to the notebook. If necessary, use the Fit button to adjust the width of columns containing asterisks (******).

Now, each problem cell displays the page and cell reference where the problem started. The notebook name is included if the cell is in another notebook.

See Also
Tracing Errors with GoTo

Setting the Zoom Factor

You can reduce the displayed size of all notebook cells, or you can increase their size. The Zoom Factor property lets you pull back to see a whole printed page, or focus in on the detail of a few cells.

To change the size of the notebook, right-click the notebook title bar, and choose Zoom Factor. The default setting is 100%. Percentages less than 100% reduce the displayed data to show more columns and rows onscreen; percentages greater than 100% enlarge the display.

Note: The Zoom Factor setting doesn't affect printed output; use the Scaling option in the File|Page Setup dialog box to scale printed documents (see File|Page Setup for details).

Changing the Notebook Color Palette

Many Quattro Pro controls use a color palette for changing colors. For example, you can change colors with the block Text Color property and the page Line Drawing property. The contents of each color square on those palettes is determined by the notebook palette.

To change the notebook palette contents,

1. Right-click the notebook title bar and choose Palette.
2. Choose the color square you want to change and choose Edit Color (or just double-click the color square).
3. Click the color square in the top part of the dialog box closest to the color you want to create, or use any one of the color models to create the exact color you want. For information on color models, see Creating Custom Color Palettes.
4. Choose OK to install the new color into the notebook palette. Edit any other color squares, and choose OK when you're finished

The new color palette appears when you select other color controls in Quattro Pro.

Caution: Changing any color square in the notebook palette also changes any previous uses of that color square throughout the notebook. For example, if you set the Shading property for a cell with the color square that's initially set to red (the leftmost square in the second row of the palette) and later change the corresponding color square in the notebook palette to green, the shaded cell switches to green.

To return the notebook palette to the default Quattro Pro settings, right-click the notebook title bar, and choose Palette. Then choose Reset Defaults.

Hiding Parts of the Notebook Display

To allow more space in a notebook window, or to create a special presentation notebook, you may want to temporarily hide the scroll bars or the page tabs.

To hide scroll bars or tabs, right-click the notebook title bar and choose Display. Then uncheck Vertical Scroll Bar, Horizontal Scroll Bar, or Page Tabs. After you choose OK, the scroll bars and/or page tabs disappear.

To restore them, right-click the notebook title bar, choose Display and check the items you want redisplayed.

See Also
Hiding Parts of the Wiindow

Making a Notebook a Macro Library

A macro library is convenient place to store frequently used macros. When you set a notebook's Macro Library property to Yes, Quattro Pro searches it when you try to run a macro that isn't in the active notebook.

To make a notebook a macro library, right-click its title bar and choose Macro Library. Then choose Yes and choose OK.

See Also
Using Macro Libraries

Creating System Notebooks

A system notebook, when hidden, stays open even when users choose File|Close All. Application developers can use system notebooks to hold application macros and ensure their availability while Quattro Pro is running.

To designate a notebook as a system notebook,

1. Right-click the notebook title bar to display the active notebook Object Inspector.
2. Choose System, then set System Notebook to Yes.
3. You'll probably want to designate the notebook as a macro library, too. If so, choose Macro Library in the notebook Object Inspector and set the property to Yes.
4. System notebooks are hidden from user view and access. Choose one of these methods to hide the notebook:


You can open the notebook in your application, then use Window|Hide to hide it. To use this method, click OK to close the active notebook Object Inspector and apply the new property settings. Then, follow the instructions in Hiding and Showing Windows to hide it in your application.


You can use the active notebook Password Level property (see Assigning a Password to a File for details). If you apply the Medium protection level, the notebook will be hidden each time it is opened without a password. Choose the appropriate password protection level, then click OK to close the active notebook Object Inspector and apply the new property settings.

If you're using a system notebook for macros, you'll probably want to apply the Macro Library property too.

See Also

Using Macro Libraries

Window|Hide

Active Notebook

Building Graphs

There are two basic methods for creating a graph: use the Graph Tool on the notebook window SpeedBar to create a floating graph directly on the spreadsheet, or use the Graph|New command to create a graph in its own window. If you're new to Quattro Pro, click Quattro Pro Graphs.

Click a topic below to learn more about graphs.

Creating Graphs

Creating a Floating Graph

Creating a Graph in a Window

Creating a Text Graph

Choosing Data Series

Rearranging Data Series

Assigning Series by Pointing

Adding a Series

Graphing Grouped or Linked Data

Deleting a Series

Adjusting Legend and X-Axis Series

Assigning a Graph Type

Choosing a Graph Type

- 2-D Graphs
- 3-D Graphs
- Rotated Graphs
- Combination Graphs
- Multiple Graphs
- Text Graphs

Changing the Graph Type

Working with Graphs

Adding Titles to the Graph

Adding Labels to Bars and Data Points

Saving and Deleting Graphs

Renaming Graphs

Selecting Graphs for Editing

Copying Graphs Within a Notebook

Copying Graphs Between Notebooks

Inserting an Existing Graph on a Page

Moving and Resizing Floating Graphs

See Also

Customizing Graph Properties

Enhancing Graphs

Printing Graphs
Analytical Graphing

Graph Basics

Using graphs to analyze your data is faster and easier than examining the data cell-by-cell. A graph presents a set of data as a picture. It may uncover a trouble spot, display a trend, or illustrate a correlation between categories of data in your notebook.

A single graph can appear in three places in Quattro Pro: as a floating graph on a spreadsheet page, in a separate graph window, and as an icon on the Graphs page.

Create a floating graph directly on a spreadsheet page when you want to view or print your graph on the same page as your data.. To enhance the floating graph, double-click it to display the graph in a graph window.

Create a graph in a graph window when you want the graph to appear in a slide show or on a printed page, but not on a spreadsheet.The graph window SpeedBar has tools you use to add text annotations, create drawings, or import graphic elements such as pictures and logos. You can also customize all parts of the graph using Object Inspector menus.

Each time you create a graph, an icon appears on the Graphs page to represent the new graph. The Graphs page is always the last page of the notebook. Here you can see every graph you've created, rename graphs, print a series of graphs, or create an onscreen slide show with special effects.

The graphs you create always exist on the Graphs Page and in a Graph Window; floating graph display is optional. To go to the Graphs page, click the SpeedTab button at the bottom of the notebook window. For information about displaying a graph in a graph window, see Selecting Graphs for Editing.

Click a topic below to learn about building graphs.

Creating a Floating Graph.

Creating a Graph in a Window

Creating a Text Graph

The Graph Window

All graph editing and annotation takes place within a graph window. There are four ways to display a graph in a graph window:


Create the graph in a window. You can then edit the graph immediately after you create it.


Double-click a floating graph.


Choose Graph|Edit and select the graph name from the list that appears.


Double-click a graph icon on the Graphs page.

A graph window has the same menus as the notebook window, with one addition, the Draw Menu. This menu lists the commands needed to arrange graphics, such as Align, Group, Ungroup, Bring Forward, and Send Backward. The Draw menu also has Draw|Import and Draw|Export commands for exchanging graphics files with other applications.

The graph window SpeedBar includes a set of quick-access tools and buttons:

Cut/Copy/Paste buttons	transfer objects to and from the Windows Clipboard.
Import button	provides quick access to importing a file.
Palette List Box	displays the name of the current palette and lets you choose or create a new one.
Selection Tool	switches to selection mode after you finish using a drawing tool.
Drawing tools	create shapes and text boxes.
Palette	includes 20 color/pattern choices and a larger box showing the current selections for fill color, background color, fill style, and border color. You can modify these choices, select a different palette, or create your own palettes.

To make drawing easier, maximize the graph window.

To learn more about the graph window, click one of these topics:

Graph Window Screen Areas	Illustration and callouts of graph window areas
Graph Window Properties	Setting aspect ratio and grid options

See Also

The Graphs Page

Creating a Graph in a Window

Enhancing Graphs

The Graphs Page

The Graphs page displays an icon for every graph and slide show in the notebook. It also displays icons for any custom dialog boxes you build. On the Graphs page you can

cut and copy graph icons (to cut or copy graphs)

double-click a graph icon to edit a graph (or double-click any icon to edit its object)

rename a graph (see Renaming Graphs)

print a single graph, a group of graphs, or a slide show (see Printing Graphs)

use tools on the SpeedBar to create graphs, slide shows, dialog boxes, and SpeedBars

The Graphs page SpeedBar has its own set of buttons:

Create Slide Show	creates new slide shows (see Creating Slide Shows).
New Graph	creates a graph in a window (equivalent to selecting Graph|New--see Creating a Graph in a Window).
New Dialog	builds a new dialog box (equivalent to Tools|UI Builder--see Creating Dialog Boxes).
New SpeedBar	creates a new custom SpeedBar (equivalent to Dialog|New SpeedBar--see Basic SpeedBar Procedure).
Edit Slide Show	displays the selected slide show on the Light Table, where you can change the sequence of slides and edit transition effects and timing parameters for individual slides (see Editing a Slide Show).
Run Slide Show	runs the selected slide show (equivalent to Graph|Slide Show--see Running a Slide Show).

Click Graphs Page to learn about the parts of the Graphs page.

The Graphs page is always the last page of the notebook. To quickly display it, click the SpeedTab button at the bottom of the notebook window. (This button looks like an arrow.) When you've finished you can click it again to return to the page you left.

Creating a Floating Graph

To create a floating graph on a spreadsheet page,

1. Select the block of data you want to plot. If surrounding cells contain explanatory labels you want to use as axis or legend labels, include them in the block selection.
2. Click the Graph tool in the SpeedBar, then drag over the area of the spreadsheet page where you want to place the graph. (If you click the spreadsheet page--instead of dragging the mouse--Quattro Pro creates a floating graph of a default size.)
3. Release the mouse button. A floating bar graph appears.


Use Graph|Type to change the graph type.


Use Graph|Series to add, delete, or rearrange the data series.


Use Graph|Title to add a main title, subtitle, and axis titles.


Double-click the floating graph to display it in a graph window. In the graph window, you can right-click any part of the graph and change its properties.


Right-click the floating graph and choose the Properties command to display floating graph properties, which include the color and style of the graph border.


To enhance the graph with drawings and text annotations, use the tools on the graph window SpeedBar.


See Renaming Graphs to change the graph name. Although Quattro Pro automatically assigns a default name (Graph 1, for example), it's a good idea to give the graph a descriptive name to make it easier to find on the Graphs page and in dialog box lists.

To create a floating text graph, see Creating a Text Graph for details.

See Also

How Series Are Plotted
Rearranging Data Series
Choosing a Graph Type
Changing the Graph Type
Customizing Graphs
Enhancing Graphs
Adding Titles to the Graph

Creating a Graph in a Window

Use the Graph|New command to create a graph directly in a window when you don't want the graph to appear on a spreadsheet page. A graph created in a window can be printed separately from the spreadsheet page, displayed in a slide show, and inserted on the spreadsheet as a floating graph later, if you want. You also get the opportunity to change or rearrange data series before you build the graph.

Graph|New is available from a spreadsheet page, from the Graphs page, and when a graph window is active. In addition, the Graph tool on the Graphs page SpeedBar gives you quick access to the Graph|New command.

1. Choose Graph|New. (If the Graphs page is active, click the Create Graphs button on the SpeedBar, instead.) The Graph New dialog box appears.
2. Enter a name for the graph in the Graph Name edit field. If you don't type a new name, Quattro Pro assigns the default name shown in the edit field.
3. Assign data blocks to as many series as you want. You can select a single cell, a contiguous block, or a noncontiguous block.


Click a series button, point to the block on the spreadsheet that you want to assign to the series, then press Enter. Or type the block coordinates in the series edit field.


To create a legend series: Click the Legend button, point to a block of labels on the spreadsheet, then press Enter.


To add labels to the x-axis, define an x-axis series: Click the X-Axis button, then point to a block of labels and press Enter.

4. Choose OK. A graph window opens and displays a bar graph (the default graph type).

Creating a Fast Graph

You can also let Quattro Pro define the series for you. This procedure is similar to the Fast Graph command in Quattro Pro for DOS. To create a fast graph,

1. Select the cell containing the data you want to plot, including any explanatory labels.
2. Choose Graph|New. A dialog box appears.
3. Enter a name for the graph. (If you don't type a new name, Quattro Pro assigns the default name shown in the dialog box.) It's a good idea to give each graph a descriptive name, to make it easier to find a particular graph on the Graphs page and in dialog box lists.
4. Check the series assignments to be sure the data is plotted the way you want it. (If it isn't, see Rearranging Data Series for information.)
5. Choose OK. Quattro Pro opens a graph window that displays a bar graph (the default graph type).

See How Series Are Plotted to understand how a block of data is divided into series.

Once you create the graph, you can


Use Graph|Type to change the graph type (see Changing the Graph Type for details).


Use Graph|Title to add a graph title, subtitle, and axis titles to the graph (see Adding Titles to the Graph for details).


Use Graph|Series to change the way series are plotted (see Rearranging Data Series for details).


Change the properties of any part of the graph using Object Inspectors (see Customizing Graph Properties for details).


Use the tools on the graph window SpeedBar to enhance the graph with drawings and text annotations (see Enhancing Graphs for details).

See Also
Choosing a Graph Type

Creating a Text Graph

Text graphs do not plot data; instead they give you a clean slate on which to draw graphics and add text boxes. A text graph can serve as a title screen to a slide show, as an area to display a bulleted list of text, as a vehicle for original art, or as a temporary scratchpad where you create a drawing before cutting and pasting it to a different graph. You also can import a graphic image from another application and insert it as a floating text graph on the spreadsheet page.

Creating a text graph is easy because Quattro Pro automatically sets the default graph type to "text" when no data is selected. Like other types of graphs, text graphs can exist as floating graphs on the notebook page, or they can appear solely in a window. No matter which kind of text graph you create, you must add text and drawings in a graph window, where all the tools are located.

To create a floating text graph,

1. Select an empty cell on the spreadsheet page.
2. Click the graph button on the SpeedBar. The mouse pointer changes to a small graph.
3. Drag the pointer over the notebook area where you want the graph frame to appear. Release the mouse button, and a blank graph appears (a textless text graph).

Once you create the empty floating graph, double-click it to bring the text graph into a graph window.

To create a text graph in a window,

1. Click an empty cell on the notebook page.
2. Choose Graph|New.
3. Enter a name for the graph. If you don't type in a new name, Quattro Pro assigns the default name shown in the dialog box. The data range of the empty cell also is filled in automatically by Quattro Pro.
4. Choose OK. A blank text graph appears in a graph window.

To create a text graph on the Graphs page,

1. Click the graph tool on the SpeedBar, or Choose Graph|New. The Graph|New dialog box appears.
2. Enter a name for the graph. (If you don't type in a new name, Quattro Pro assigns the default name shown in the dialog box.)
3. Choose OK (don't enter anything for the series data).

A new icon appears on the Graphs Page to represent the graph. Double-click it to display the text graph in a graph window.

See Also

Creating Lines and Shapes

Creating Text Blocks

Importing Graphics

Exporting Graphics

Turning Graphs into Slide Shows

How Series Are Plotted

The way Quattro Pro plots your data depends on the dimensions and contents of the block you select before you create the graph.

When a block contains more rows than columns, or when there are an equal number of rows and columns in the selected block,

Quattro Pro plots each column as a single series.

If the first series contains labels, the labels are placed along the x-axis. This column of labels is called the x-axis series.

If the first row contains labels, these labels are used in the graph legend. This row is called the legend series.

When a block contains more columns than rows,

Quattro Pro plots each row as a single series.

If the first row (the first series) contains labels, it automatically becomes the x-axis series.

If the first column contains labels, these labels become the legend series.

Caution: Blank columns and rows in the block of data can create a series of zero values that may throw off the graph. Delete them (or select the data as a non-contiguous block) before you create the graph.

If you select a noncontiguous block, Quattro Pro uses the dimensions of the first sub-block you select to determine whether to plot rows or columns. See Using Noncontiguous Blocks for a description of noncontiguous blocks, and how to select them.

If you select a 3-D block (a block from a spreadsheet page that is grouped to other pages), Quattro Pro starts the block with the row or column from the first page, then appends the corresponding blocks from succeeding pages to form the series. For example, if you select the block A:B3..C:D11, the first series is A:B3..C:B11, the second series is A:C3..C:C11, and the third series is A:D3..C:D11. For more information about 3-D blocks, see Using 3-D Blocks.

Pie graphs, doughnut graphs, column graphs, and 3-D area graphs plot negative numbers as positive values.

See Also

Rearranging Data Series

Assigning Series by Pointing

Rearranging Data Series

Selecting a block of data is a good way to begin a graph when your data is set up as Quattro Pro expects it. If the data doesn't graph correctly (for example, if you want to plot columns of data and Quattro Pro divides your data into rows) use the Graph|Series command to rearrange the data in the graph without changing the data on the spreadsheet page. The Graph|Series dialog box has these features:

Series buttons appear on the left side of the dialog box. These buttons are labeled with the series name or number (X-axis, Legend, 1st, 2nd, and so forth), and they activate point mode.

Edit fields, located to the right of the buttons, display the block coordinates of the series.

Add and **Delete buttons** let you insert a series after the selected series, or remove the selected series.

Reverse series plots the last series first, then moves backward through the series order to plot the first series last. This feature is useful in 3-D bar and unstacked area graphs, for example, where a large first series plotted at the front of the graph might obscure other series.

Row/column swap plots columns as series when the series you've selected are rows, and plots rows as series when the series you've selected are columns. Row/column swap also puts x-axis series labels in the legend, and places legend series labels along the x-axis.

To assign a data block to a series:

1. Select the graph, then choose Graph|Series.
2. Click the series button or double-click the edit field for the series. The graph and the dialog box move behind the spreadsheet. The current block coordinates for the series are highlighted on the spreadsheet.
3. Drag the mouse to select the block. To select a noncontiguous block, hold down the Ctrl key as you select the parts of the block.
4. To return to the dialog box, click the maximize arrow in the dialog box title bar (which appears over the SpeedBar), or press Enter.
5. Change as many series assignments as you want, then choose OK.

See Also

Adding a Series

Graphing Grouped or Linked Data

Deleting a Series

Assigning Series By Pointing

In the Graph|New and Graph|Series dialog boxes, you can point to a block to assign it to a data series, using the following procedure:

1. Click the series button or double-click the edit field for the series. The graph and the dialog box move behind the spreadsheet. The current block coordinates for the series are highlighted on the spreadsheet.
2. Drag the mouse to select the block. To select a noncontiguous block, hold down the Ctrl key as you select the parts of the block.
3. To return to the dialog box, click the maximize arrow in the dialog box title bar (which appears over the SpeedBar), or press Enter.
4. Change as many series assignments as you want, then choose OK.

See Also

Using Noncontiguous Blocks

Using 3-D Blocks

Adding a Series

To add a new, blank series to a new or existing graph,

1. Choose Graph|New or Graph|Series.
2. Select the series you want to precede the new series.
3. Choose Add. The new series is added after the selected series. If no series is selected (or if the x-axis series or legend series is selected), the series you add becomes the last one in the graph.

Once you create the series, you can assign a block by pointing, or by typing block coordinates into the series edit field.

Adding a new first series

The Graph|New and Graph|Series commands give you complete control over what is plotted in the graph. You can plot rows, columns, and noncontiguous blocks in a single graph (or in a single series). But since a new series is always added after the selected series, you have to do a little rearranging to add a new first series:

1. Select the current first series.
2. Choose Add. A new, blank series is added after the first series.
3. Enter the block coordinates currently assigned to the first series into the edit field for this new second series. (The first and second series should now have the same data block coordinates.)
4. Choose new data for the first series, either by pointing to a block, or by deleting the old block coordinates and typing in the new.

Adding series through the Clipboard

In the graph window, you can also add one or more series directly to a graph by cutting and pasting data through the Clipboard.

1. On the spreadsheet page, select the block you want to assign to the series.
2. Choose Copy.
3. Use Graph|Edit to display the graph in the graph window.
4. Select the insertion point for the series:

To insert the new series after an existing series, select the bar, line, or area that represents the series.

To add the block as the first series in the graph, select the graph pane (the area bounded by the x-axis and y-axis).

To add the block as the x-axis series, select the x-axis. (This method is especially useful for adding a numeric x-axis series to an XY graph.)

5. Choose Paste.

The new series are always plotted to match the existing row/column arrangement of the graph. For example, if you copy a row of four cells to the Clipboard, then paste it into a bar graph where columns are plotted as series, you'll see four new bars plotted at the first x-axis division, because you've just added four new series, of one value each, to the graph.

See Also

Adjusting the Legend and X-Axis Series

Assigning Series by Pointing

Graphing Grouped or Linked Data

Adjusting the Legend and X-Axis Series

When you add or delete a data series, the legend series (if one is defined) does not change automatically. If you insert a new series into the graph, or delete a series from the graph, you also have to insert or delete an entry in the appropriate position in the legend series. The easiest way to do this is to delete the old legend series and define a new one:

1. Choose Graph|Series.
2. Click the Legend button to activate point mode.
3. While holding down the Ctrl key, point to the legend label for each series, in order.
4. Choose OK.

You can also enter legend label text (or label cell coordinates) for individual series through the series Object Inspector (see Overriding a Legend Label for details).

Note: You don't have to change the x-axis series when you add another series to the graph. Additional series just add another bar, line marker, or area boundary point at each x-axis division. If you see new x-axis divisions without labels after you add a series, it means the new series has a greater number of data points than the other series in the graph. Select the series again, and assign a block that has the correct number of values.

Graphing Grouped or Linked Data

When you assign a 3-D block to a series, Quattro Pro starts the block with the row or column from the first page, then appends the corresponding blocks from succeeding pages to form the series. For example, if you select the block A:B3..C:B5, the series has 9 values, in this order: A:B3, A:B4, A:B5, B:B3, B:B4, B:B5, C:B3, C:B4 and C:B5. See Using 3-D Blocks for details.

When a graph appears in one notebook, and some or all of the data for the graph is located in another notebook, the graph is *linked* to the data. To assign data located in a different notebook to a series, either type the block coordinates in the form `[Notebookname]Pagename:Block Reference`, or assign blocks by pointing. All data assigned to an individual series must come from the same notebook.

This is one method for managing windows as you point to blocks in different notebooks:

1. Open all the notebooks that contain data you need for the graph.
2. Close all notebooks you don't need for the graph. (This makes it easier to see the data you do need.)
3. Click the notebook where you want to place the graph, to make its window the active window.
4. Choose Graph|New.
5. Choose a series button, then point to a block of data and press Enter to define the series. To switch notebooks, choose a notebook from the Window menu, or use Ctrl + F6 to cycle through the notebooks.
6. Choose OK when you've finished assigning blocks to series.

Each time you open a notebook, Quattro Pro checks whether all notebooks that provide data for its graphs are open. If they are not, a dialog box with three options, Open Supporting, Update References, and None, appears. See Loading Supporting Data for an explanation of these options.

Deleting a Series

To delete a series from a graph,

1. Choose Graph|Series.
2. Click the edit field to select the series.
3. Choose Delete, then choose OK.

In the graph window (but not on a floating graph) you can also select a bar, area, or line, directly on the graph and click the Cut button to delete the series.

See Also
Adjusting the Legend and X-Axis Series

Choosing a Graph Type

Quattro Pro builds many different graph types automatically. The type of graph you choose usually depends on the analysis you want to perform. Think about what you want to illustrate. Total sales dollars for each division of a company? The number of products produced each month of a year? The ratio of personnel expense to other classes of expense? How hours worked affect productivity?

Certain graphs are best suited for plotting certain types of data:

Bar graphs	compare values of different items at specific points in time--to contrast monthly commissions for each sales representative, for example.
Line graphs	show the progression of values over time--to track sales, for example.
Area graphs	show the relationship of each value to the total over time--how each sales representative contributed to total sales over a 12 month period, for example.
Stacked bar graphs	show the relationship of each value to the total--how total sales are divided between regions, for example. 100% stacked bar graphs show the percentage each series contributes to the total.
Comparison graphs	have lines connecting the boundaries between series. This makes it easier to compare series values or proportions from one bar to the next.
Pie or column graphs	compare individual values to other values and to the whole--how yearly expenses break down into categories, for example. Use them to focus on the individual values in a single series.
Doughnut graphs	like pie and column graphs, plot a single series with each value plotted as a percentage of the whole. You can add text or graphics to the interior "hole" with graph window drawing tools.
XY graphs	plot values in one series against those in another--to show the relationship between salary and length of employment, for example.
High-low graphs	illustrate the difference between corresponding values in two series. Though most often used in tracking daily stock prices, high-low graphs can be used whenever you want to compare the difference between pairs of values.
Surface graphs	plot rows and columns as intersecting lines on a surface that is suspended in a 3-D frame. Surface graphs are useful for plotting functions such as f(x) and f(x,y), and parametric curves (x(t), yt(t)).
Radar graphs	show x-axis values as imaginary lines radiating from a common center, like spokes of a wheel, with y-axis values plotted on each "spoke." They are useful for highlighting trends, depending on the shapes drawn by the plot lines, or simplifying series comparisons.
Text graphs	show drawings and text instead of values. They are useful for slide shows and view graph presentations.

Each of the basic graphs can be drawn in various ways. Quattro Pro divides these variations, or graph types, into the following categories:

2-D Graphs

3-D Graphs

Rotated Graphs

Combination Graphs

Multiple Graphs, which plot each series as a separate sub-graph, are found under Combination (Comb) in the Graph|Type dialog box, and in the graph setup and background Object Inspector.

Text Graphs

2-D Graphs

2-D graphs (except pie, doughnut, and column graphs) position data relative to two scales: the x-axis and the y-axis. The x-axis is the horizontal line at the bottom of a graph. It shows progression of values and often represents time (days, weeks, quarters, and so on). The vertical line is the y-axis. Y-axis scale values, which serve as reference points for the placement of bars, lines, and markers on the graph, are determined by the data plotted in the graph.

XY graphs (often called scatter diagrams) plot values against two numeric scales. The x-axis scale is determined by the values in the x-axis series. (Unlike in most other graph types, where the x-axis series contains labels, the x-axis series in an XY graph contains data.) The y-axis scale is determined by the values in all the other series.

2-D pie, doughnut, and column graphs do not have axes at all. Pie and doughnut graphs illustrate the values of a single series as slices in a circular whole. Column graphs also illustrate only a single series, and show values as sections of a column.

Quattro Pro offers the following 2-D graph types. To learn more about a graph type, click one of the following topics.

3-D Graphs

3-D graphs (except pie, doughnut, and column graphs) plot multiple series against a background of two walls and a base. You can thicken the walls, or hide the base and walls, through the 3-D Options property in the graph setup and background Object Inspector (see Changing 3-D Options for details). The first series is plotted at the front of the graph, from left to right. Successive series are drawn, in order, behind the first. Legend labels appear along the z-axis in most 3-D graphs.

You can adjust the viewpoint and the amount of perspective applied to these graphs with the 3-D View property in the Graph Setup and Background Object Inspector. 3-D View also lets you control the height and depth of the graph. See Adjusting 3-D View for more information.

3-D pie, doughnut, and column graphs are drawn in perspective, and like their 2-D counterparts, they represent a single series. There are no axes or walls in these graphs, and 3-D view options are not available.

Quattro Pro offers these different 3-D graph types. To learn more about a graph type, click one of the following topics:

Bar Graphs

Line Graphs have a 3-D version, the Ribbon graph

Area Graphs

Pie and Column Graphs

Doughnut Graphs

Surface Graphs

Rotated Graphs

Rotated graphs are plotted with the axes reversed. The x-axis is the vertical axis, and the y-axis is the horizontal axis. Quattro Pro offers the following rotated graph types. To learn more about a graph type, click one of the following topics:

Bar Graphs

Line Graphs include a Rotated Line graph

Area Graphs include a Rotated Area graph

Combination Graphs

Combination graphs mix bars, lines, and areas in the same graph, to highlight a series or to provide a contrast between different series in the graph. Quattro Pro builds three combination graphs automatically:

Line-Bar graphs	plot the first series as a line, and plot all other series as bars.
Area-Bar graphs	plot the first series as an area, and plot all other series as bars.
High Low-Bar	graphs plot corresponding data points in the first and second series as I-beams, then plot all other series as bars. The bars are plotted against the secondary y-axis, and the scale for this axis is adjusted to plot the bars on the lower one-fourth of the graph (see High-Low Graphs for more information).

You can create your own custom combination graphs by overriding the graph type for any series in 2-D bar, rotated bar, 2-D line, rotated line, and variance graphs (see Changing Graph Series Options for details).

See Also
Multiple Graphs

Multiple Graphs

Multiple graphs (found under Comb in the Graph|Type dialog box) plot each series as a separate graph. You can graph as many series as you like--each part of the graph scales down in size to fit the entire graph on the screen or page.

Although it looks like several different graphs, a multiple graph is a single graph. When you change a property (the label font, for example) on one part of a multiple graph, all the other parts change accordingly. If you define a legend series, the legend label for each series becomes the "title" of the small graph that represents that series.

Quattro Pro offers these multiple graph types:

Bar Graphs include the Multiple Bar graph type

Pie and Column Graphs include Multiple 2-D Pie, Multiple 3-D Pie, Multiple 2-D Column and Multiple 3-D Column graphs

You can also create a multiple line graph or a multiple area graph by right-clicking any bar in a multiple bar graph,choosing the Properties command to display an Object Inspector, then overriding the graph type for the series with a line or area type (see Creating a Custom Combination Graph for details).

See Also
Combination Graphs

Bar Graphs

Bar graphs represent each value in a series as a bar. Values are plotted against the y-axis scale; the taller the bar, the greater the value. You can customize bar graph features such as bar width, bar margins, and bar colors, using the bar series Object Inspector. See Customizing Bar Graphs for more information.

2-D Bar graphs are the default graphs in Quattro Pro. When you plot more than one series, the bars appear side by side, providing a good comparison of corresponding values in different series.

Variance graphs show how the values in each series deviate from the zero line, which is an arbitrary baseline you choose through the Scale property in the y-axis Object Inspector.

Stacked Bar graphs plot the corresponding values of different series in vertical stacks, showing both the total reached by the combined values, and the way each value contributes to that total. The first series is plotted on the bottom; additional series are stacked on top, in order.

100% Stacked Bar graphs show the relationship of each value to the total by displaying the percentage of the total contributed by each series. They may be 2-D, rotated 2-D, 3-D, or rotated 3-D.

Comparison graphs have lines connecting the boundaries between series. This makes it easier to compare series values or proportions from one bar to the next. They may be stacked bar or 100% stacked bar graphs, 2-D vertical or rotated.

3-D Stacked Bar graphs also show individual and cumulative values. The first series is plotted on the base of a three-dimensional grid; additional series are stacked on top.

3-D Bar graphs plot series as columns with square tops and bases on a three-dimensional grid. The first series is plotted across the front of the graph, from left to right. Bars that represent additional series are plotted, in order, behind the first.

2.5-D Bar graphs are similar to 3-D bar graphs, except the bars of all series are drawn next to each other (rather than behind each other) on the three-dimensional grid. Each bar projects all the way to the back of the grid.

Step graphs look similar to 3-D bar graphs, but adjacent bars touch, to emphasize the progression of values in the series. In this way, the step graph serves the same purpose as a 3-D unstacked area graph (see Area Graphs for details), but the steps give greater emphasis to individual values.

Rotated Bar graphs plot values as horizontal bars. You might use this graph type instead of a standard bar graph when you want to emphasize that values "move forward," rather than increase. This is also a good bar graph choice if you have long x-axis labels.

Rotated 3-D Bar graphs display series values as horizontal columns on a three-dimensional grid. The first series is plotted at the front of the graph, and additional series are drawn behind the first. Unlike standard 3-D bar graphs, which display legend text along the z-axis, a rotated 3-D bar graph has a legend box.

Rotated 2.5-D Bar graphs plot the values in each series as horizontal plates that project from the front to the back of a three-dimensional grid. The first value in the first series is plotted on the base of the graph, and corresponding values of additional series are layered on top, in order.

Multiple Bar graphs plot each series in its own small bar graph, to provide side-by-side comparisons of different sets of data. Even though each series looks like a separate graph, a multiple bar graph is really one graph. Each unit displays the same y-axis scale, x-axis divisions, and so forth. Any changes to one graph unit are reflected in all the others. You can plot as many series as you like in a multiple bar graph, but the plot for each series gets smaller as you add new series.

To create a multiple line graph or multiple area graph, right-click any bar in a multiple bar graph, choose the Properties command to display an Object Inspector, then override the graph type for the series with a line or area type (see Creating a Custom Combination Graph for details).

Line Graphs

Line graph types plot series values as points, then connect the points in each series with a line. You can customize markers, line styles, and other properties using the line series Object Inspector (see Customizing Line Graphs for details).

2-D Line graphs show how values change over time. Since it is so simple, the 2-D line graph is the easiest graph to read when there are many data series plotted.

XY graphs (often called scatter diagrams) plot all series against the x-axis series, to display a correlation between series as a function of the x-axis data. XY Graphs describes how to create an XY graph.

Ribbon graphs are essentially line graphs plotted on 3-D grids, with each line flattened out into a segmented ribbon. Ribbon graphs are good for showing trends over time, but individual points are less visible than on standard line graphs.

Rotated Line graphs, like standard line graphs, show the pattern of values over time, but the x-axis and y-axis positions are reversed. The rotation leaves more room for long x-axis labels.

Area Graphs

Area graphs (except 3-D unstacked area graphs) plot cumulative rather than individual values. The first series is plotted without modification. The second series is plotted using the top of the first series as the baseline. The third series is graphed on top of the second, and so forth. You can customize area graphs using the area series Object Inspector (see Customizing Area and Surface Graphs for more information).

2-D Area graphs best show how each series affects the whole over time. Although only the first line plotted is an accurate pattern (a dip in a further series might appear as a rise if values beneath it are high), the size of the area corresponding to each series represents its contribution to the whole.

3-D Area graphs are like 2-D area graphs, except series are plotted as three-dimensional areas inside a frame of two walls and a base.

Rotated Area graphs reverse the x-axis and y-axis, so areas are stacked side-by-side. The width of an area indicates the series contribution to the whole.

3-D Unstacked Area graphs plot actual series values, not cumulative values. The first series is plotted at the front of the graph; additional series are drawn behind it. Like line graphs, 3-D unstacked area graphs are useful for showing changes over time.

Pie and Column Graphs

Pie and column graphs plot a single series. Each value is plotted as a percentage of the whole. You can change pie and column graph properties using Object Inspector menus (see Customizing Pie and Doughnut Graphs and Customizing Column Graphs for more information).

Pie graphs plot each value in a series as a "slice" of the "pie."

3-D Pie graphs have a thicker look for a more dramatic appearance, but otherwise are exactly like 2-D pie graphs.

2-D Multiple Pie graphs display different data series as side-by-side pie graphs. Use this graph type to provide a comparison of how individual values in the series affect the whole, and how these proportions change from series to series.

3-D Multiple Pie graphs display data a little more dramatically than the 2-D version, but you use them in the same way, to provide comparisons of how each value in a series affects the whole, and how these proportions change from series to series.

Column graphs represent values as sections of a vertical rectangle. The first value in the series is plotted on the bottom; additional series are stacked on top, in order.

3-D Column graphs plot series values as sections of a column. Like the 2-D version, 3-D column graphs plot series values from the bottom up.

2-D Multiple Column graphs display each series as an individual column graph. Like multiple pie graphs, multiple column graphs provide a comparison of how individual values in the series affect the whole, and how these proportions change from series to series. In addition, the column arrangement leaves more room for labels.

3-D Multiple Column graphs plot each series as individual 3-D column graphs. They have the same purpose as 2-D multiple column graphs.

Doughnut graphs are like pie graphs with holes. For details, see Doughnut Graphs.

Surface Graphs

Surface graphs plot rows and columns as lines that form a "mesh" on a 3-D frame. When you graph a square array of spreadsheet cells, values in columns correspond to mesh lines from left to right. Rows of data correspond to mesh lines from front to back. Each data point is represented by an intersection of two mesh lines. The way the surface of the mesh is shaded depends on the type of surface graph (3-D Surface, 3-D Contour, or 3-D Shaded Surface) you choose.

3-D Surface graphs shade the area between each series with a different color.

3-D Contour graphs have shading that follows the grid lines, making it easy to compare points on the surface with values on the y-axis scale.

3-D Shaded Surface graphs shade a single-colored surface as though a light is shining down from directly above the graph. Flat areas, where values are the same from series to series, or from one point to another in the same series, are lightest. Places where there are great differences between adjacent values have the darkest shading.

You can customize surfaces in these graph types using the area series Object Inspector (see Customizing Area and Surface Graphs for details).

XY Graphs

XY graphs plot data against two scaling axes. The x-axis scale is determined by the x-axis series, which in XY graphs contains data, not text labels. The y-axis scale is calculated from the data in all the other series you plot.

Each series value is plotted as a pair of coordinates. The first coordinate is an x-axis series value. The second coordinate is the corresponding value in the series you're plotting.

The first coordinate determines where the data point is placed relative to the x-axis. The second coordinate gives the data point's position in relation to the y-axis.

To create an XY graph:

1. On the spreadsheet page, select all series except the x-axis series (Quattro Pro will automatically designate an x-axis series only if the series contains labels). You can also select legend labels if you want.
2. Choose Graph|New.
3. To select the x-axis series, click the x-axis series button, point to a block on the spreadsheet, then press Enter.
4. Choose OK.

XY graphs are useful for illustrating statistical trends. Graphing Frequency Distributions describes how to plot a frequency distribution using an XY graph.

High-Low Graphs

The usual purpose of high-low (open-close) graphs is to track daily stock prices. You need at least two series: the highest and lowest price a stock reached each day. You may also have at least two other series: the opening and closing prices for the stock each day.

High-low graphs use the first two series to create a set of I-beams. The top of each I-beam shows the high daily value, and the bottom of the I-beam shows the low value. The way Quattro Pro plots additional series depends on how many series you assign to the high-low (open-close) graph.


When you plot only three series, the third series contains the close data. This data is shown as a tick mark to the right of the I-beam.


When you plot four series, the third series contains the open data, and the fourth series contains the close data. Each open value is plotted as a tick mark to the left of the I-beam, and each close value is shown as a tick mark to the right of the I-beam.


When you plot more than four series in a high-low graph, the additional series are plotted as lines.

To create a high-low graph,

1. Choose Graph|New.
2. Assign blocks to each series by pointing, then choose OK. Quattro Pro creates a bar graph. (See Assigning Series by Pointing for details.)
3. Choose Graph|Type. 2-D graphs are selected automatically.
4. Choose the High-Low icon (the first icon in the second row), then choose OK.

Adding a volume series

A common use for the fifth series is to display the volume data, or number of shares of stock that changed hands that day. Volume data is usually much larger than high-low (open-close) data, so the high-low data does not show up well in the graph. The High Low-Bar graph has special formatting to deal with this problem. High Low-Bar graphs plot the first four series like regular high-low graphs. The fifth series, however, is plotted as bars against the secondary y-axis. The secondary y-axis scale is adjusted to plot the bars on the lower one-quarter of the graph, where the bars are unlikely to cover the high-low (open-close) values. To do this, Quattro Pro calculates the High and Increment values for the secondary Y-Axis then multiplies them by four (the Increment adjustment makes the scale less crowded.) To create a High Low-Bar graph with volume series:

1. Choose Graph|New.
2. Assign blocks to each series by pointing, then choose OK. Quattro Pro creates a bar graph. (See Assigning Series by Pointing for more information.)
3. Choose Graph|Type.
4. Choose Combo.
5. Choose the High Low-Bar icon (the first icon in the third row), then choose OK.

You can change the volume series from bars to a line or area by overriding the graph type for the series (see Changing Graph Series Options for details).

High-Low Styles

Quattro Pro offers four different high-low styles:


I-Beam is the default style. High and low values are at each end of the I-beam. Open and close are represented by left and right tick marks, respectively.


Line style connects corresponding high and low values with a line, and shows open and close values as left and right tick marks.


Bar style give you a "bar and whisker" or "candle" graph. A line connects high and low values. A bar spans the open and close values. When the close value is higher than the open, the bar is white. When the open value is higher than the close, the bar is red. (Retain this light/dark relationship if you change fill colors).


Marker style assigns different-colored markers to high, low, open, and close values, and connects each set of corresponding values with a line.

To change the style of your high-low graph:

1. Right-click any high-low marker and choose the Properties command (this property is global, so you don't have to select a specific series). The bar series Object Inspector appears.
2. Choose Hi-Lo Bar Style.
3. Select a bar style, then choose OK.

Tip: You can use the line series Object Inspector to give each marker in the Marker high-low bar style a different look. Change the graph type to line, right-click a series and choose the Properties command to display an Object Inspector, then change the marker style, weight, and appearance. When you're finished, change the graph type back to High-Low.

Doughnut Graphs

Like pie and column graphs, doughnut graphs plot a single series. Each value is plotted as a percentage of the whole. Doughnut graphs can be 2-D or 3-D. The only difference is that 3-D graphs have height as well as diameter. You can add use the drawing tools in the graph window to add graphics and text to the interior "hole."

To change doughnut graph properties, right-click any part of the graph and choose the Properties command to display its Object Inspector. For example, if right-click and choose the Properties option for a doughnut section, the pie graph properties Object Inspector appears. As with pie graphs, you can "explode" any section.

See Also
Pie and Column Graphs

Radar Graphs

Radar graphs are circles with a line for each x-axis value extending from the center to the edge, like the spokes of a wheel. Y-axis values for each series are graphed on the spokes.

Radar graphs can highlight trends. Growth trends show as outward spirals; static or fluctuating values look more like circles or stars. Radar graphs can also simplify series comparisons; the series with the highest values occupies the most area.

To display a radar graph, choose Graph|Type|2-D, then click the Radar button.

You can customize radar graph markers and lines. To change radar graph properties, right-click any part of the graph and choose the Properties command to display an Object Inspector. For example, if you right-click one of the series markers and choose the Properties command or choose Property|Series|series number, the line series properties Object Inspector appears.

See Also

Line Graphs

Customizing Line Graphs

Text Graphs

Text graphs do not plot data; instead they give you a clean slate on which to draw graphics and add text boxes. A text graph can serve as a title screen to a slide show, as an area to display a bulleted list of text, as a vehicle for original art, or as a temporary scratchpad where you create a drawing before cutting and pasting it to a different graph. You also can import a graphic image from another application and insert it as a floating text graph on the notebook page.

Creating text graphs

Creating a text graph is easy because Quattro Pro automatically sets the default graph type to "text" when no data is selected. Like other types of graphs, text graphs can exist as floating graphs on the notebook page, or they can appear solely in a window. No matter which kind of text graph you create, you must add text and drawings in a graph window, where all the tools are located.

To create a floating text graph,

1. Select an empty cell on the notebook page.
2. Click the graph button on the SpeedBar. The mouse pointer changes to a small graph.
3. Drag the pointer over the notebook area where you want the graph frame to appear. Release the mouse button, and a blank graph appears (a textless text graph).

Once you create the empty floating graph, double-click it to bring the text graph into a graph window.

To create a text graph in a window,

1. Click an empty cell on the notebook page.
2. Choose Graph|New.
3. Enter a name for the graph. If you don't type in a new name, Quattro Pro assigns the default name shown in the dialog box. The data range of the empty cell also is filled in automatically by Quattro Pro.
4. Choose OK. A blank text graph appears in a graph window.

To create a text graph from the Graphs page,

1. Choose Graph|New.
2. Choose OK (don't enter anything for the series data). A blank text graph appears in a graph window, ready for you to add drawings or text.

Changing the Graph Type

When you create a graph, Quattro Pro automatically builds a 2-D bar graph. To change the graph type:

1. Choose Graph|Type. The Graph Type dialog box has two parts. On the left is a list of graph categories, on the right is a gallery of icons that match the selected category.
2. Click a graph category from the list, then click an icon to choose a graph type from that category. The name of the graph type you select appears below the graph icons.
3. Choose OK. Quattro Pro rebuilds your graph using the new graph type.

You can also change the graph type from the Graph Setup and Background Object Inspector:

1. Display the graph in a graph window (see Selecting Graphs for Editing for more information).
2. Right-click the graph background and choose the Properties command or choose Property|Graph Setup. Graph Type options are displayed when the Graph Setup and Background Object Inspector appears.
3. Click a graph category from the list, then click an icon to choose a graph type from that category. The name of the graph type you select appears below the graph icons.
4. Choose OK. Quattro Pro rebuilds your graph using the new graph type.

Adding Titles to the Graph

The Graph|Title command creates a title and subtitle for every graph type except Text graphs. This command also places titles on the graph axes. On 2-D graphs, you can add titles for the x-axis, the primary y-axis (the left, or Y1 axis) and for a secondary y-axis (the right, or Y2 axis). 3-D graphs will not display secondary y-axis titles.

To add titles to a graph,

1. Select the graph.
2. Choose Graph|Title.
3. Enter text in the appropriate edit fields, and choose OK.

Tip: If your y-axis scale labels are greater than 1000, you can use the Show Units option to simplify the scale and add an appropriate title--"(thousands)" for example--between the y-axis title and the axis itself. Show Units is one of the Scale property options in the y-axis Object Inspector.

You can change the font, color, or other properties of the Graph Title, Graph Subtitle, and the Graph Title Box, through their Object Inspectors. You can edit the main title and subtitle text directly on the graph, or change the title or subtitle in the edit field of either the Graph|Title dialog box or the Object Inspector.

You must edit axis title text in the Graph|Title dialog box, or in the Axis Title Object Inspector. This Object Inspector also lets you change the font, color, and other properties of axis titles.

Adding Labels to Bars and Data Points

A *label series* places labels on the bars, markers, or data points that represent each value in a series.

To define a label series,

1. Enter a series of labels into a block on the spreadsheet, if suitable labels don't already exist.
2. Right-click the bar, line, or area that represents the series and choose the Properties command. The Series Options property is automatically selected.
3. Double-click the Label Series edit field, then point to the block on the spreadsheet page that contains the labels and press Enter.
4. Choose OK. (You can also type the block coordinates directly into the edit field, then choose OK.)

The block you specify as the label series can contain text or data. For example, if you assign the same block to the data series and the label series of a bar graph, Quattro Pro places a label that shows the actual data value over each bar in the series.

To change the properties of these labels, see Series Label. To delete labels, delete the label series.

Saving and Deleting Graphs

When you save a notebook, Quattro Pro automatically saves every graph you've created in that notebook. To see these graphs, click the SpeedTab button at the bottom of the notebook page to turn to the Graphs page.

To delete a graph, choose Graph|Delete, select the graph name from the list that appears, and choose OK. If you have the Graphs page selected, you can choose an icon, then choose Cut or press Delete to delete the graph.

Floating graphs appear both on the spreadsheet page and in a graph window. If you select a floating graph, then choose Cut, the graph is erased from the spreadsheet, but it is NOT deleted from the notebook. You must use Graph|Delete, or cut the graph icon in the Graphs page, to delete the graph from the notebook.

See Also

Renaming Graphs

Selecting Graphs for Editing

Renaming Graphs

When you create a new graph in a window, you can either accept the default name Quattro Pro assigns, or enter a new one. When you create floating graphs, Quattro Pro automatically gives them the default names of Graph 1, Graph 2, and so on, according to the order in which they are created in the notebook. If you rename the graph, to give it a more descriptive name, it is easier to find the graph in dialog box lists and on the Graphs page.

You can rename a graph only from the Graphs page, which is the last page in the notebook. To change the graph name,

1. Click the SpeedTab button to display the Graphs page.
2. Right-click the icon of the graph you want to rename. Then, choose Icon Properties|Name.
3. Type the new name in the dialog box that appears, and choose OK.

See Also
The Graphs Page

Selecting Graphs for Editing

Before you can edit or view a stored graph, you must select the graph.

On the spreadsheet page, select a floating graph by clicking it. On the Graphs page, which is the last page in the notebook, select a graph by clicking its icon. In either place, you can use Graph menu commands to change the graph type, add graph titles, rearrange the data series, and view the selected graph full-screen.

A graph must be displayed in a graph window before you can use Object Inspectors to change its properties or enhance it with drawings and text. To display a graph in a graph window,


Double-click a floating graph on the spreadsheet page


Double-click a graph icon on the Graphs page


Choose Graph|Edit anywhere in the notebook, select the graph name from the list that appears, and choose OK.

Copying Graphs Within a Notebook

Ther are two ways to copy a graph within a notebook: use the Graph|Copy command, or copy and paste the graph through the Windows Clipboard.

The Graph|Copy command lets you copy the entire graph, or just selected characteristics of the graph. You can choose to copy the style, the data, the annotation objects, or any combination of these attributes, from one graph to another within the same notebook. You choose two graphs, one to copy "from" (the source graph) and one to copy "to" (the target graph). The target graph can be an existing graph or a new graph.

Style	copies all properties that affect the appearance of the graph such as text fonts, line styles, and fill colors. Style also copies the graph type, and series overrides. For example, if you change the background color of the source graph, choose a dotted line style for the y-axis grid, remove the box around the legend, and plot the fourth series on a secondary y-axis, all these changes are transferred to the target graph when you copy the style.
Data	builds the destination graph from the same data as the source graph. When you choose this option, Quattro Pro copies a reference to the data used in the source graph. (Data is not physically copied to a different location in the notebook.) Any changes you subsequently make to the data affect the source graph and all copies. The Data option also copies x-axis and legend series, legend label overrides, label series, and graph title, subtitle, and axis title text (but not the style of this text).
Annotate	(annotation objects) copies all text and drawings you created for the source graph using the drawing tools on the graph window SpeedBar. Annotate also includes any graphics you imported for the source graph.

Caution: To preserve annotation objects in the destination graph, uncheck Annotate. Annotate erases all annotation items in the destination graph, even when there are none to copy from the source graph.

The options you choose can produce quite different results:

If you check Style, Data, and Annotate, Quattro Pro builds an exact copy of the original graph. (You can copy the source graph through the Clipboard to get the same results.)

If you check Style and Annotate, but not Data, Quattro Pro builds a graph that has the same graph type and properties as the source graph, but uses the data of the destination graph. (The source graph serves as a style template for the destination graph).

If you copy to a new graph, Quattro Pro automatically creates this graph using data from the source graph (whether or not you checked the Data option in the dialog box). If you don't copy Style to a new graph, you create a 2-D bar graph.

To copy a graph using the Graph|Copy command,

1. Choose Graph|Copy.
2. Select a graph from the "From" list. This graph is the source of the style, data, and annotation objects you choose to copy.
3. Select a graph from the "To" list, or type in the name of a new graph.
4. Check Style, Data, Annotate, or any combination of the three, to select the items to copy. To omit an item from the copy procedure, uncheck it.

5. Choose OK.

To copy a graph through the Windows Clipboard,

1. Select the graph.

In the graph window, hold down Ctrl and click the graph background. To include annotations, first select the graph, then hold down Shift and click each annotation object (or drag to select a group of annotation objects).

On a spreadsheet page or the Graphs page, click the floating graph or graph icon. (All annotation objects are selected automatically.)

2. Click the Copy button on the SpeedBar.
3. Open the destination notebook (or use the Window menu to choose the notebook, if it is already open).
4. Select the page where you want to paste the graph.

To place the copy as a floating graph on a spreadsheet page, select the page.

If you want the graph to appear only in a window and on the Graphs page, click the SpeedTab button to display the Graphs page. (You cannot paste a graph directly into a graph window.)

5. Click the Paste button.

Note: If you paste a graph onto a spreadsheet page, then change your mind and click the Cut button or press Del, you have not erased the graph from the notebook. (The graph still exists in a graph window and on the Graphs page.) You must use Graph|Delete, or select the icon on the Graphs page and click the Cut button, to delete the graph entirely.

See Also

Copying Graphs Between Notebooks

Copying Graphs Between Notebooks

To copy a graph between notebooks, use the Clipboard. This process copies the graph, but does not copy the data.

1. Select the graph.

To select a graph in a graph window, hold down Ctrl and click the graph background. To include annotations, first select the graph, then hold down Shift and click each annotation object.

To select a graph on the spreadsheet page or Graphs page, click the floating graph or graph icon. (All annotation objects are selected automatically.)

2. Choose Copy.
3. Open the notebook you want to copy to (or use the Window menu to choose the notebook, if it is already open).
4. To place the copy as a floating graph on a notebook page, select the page and choose Paste. If you want the graph to appear only in a window, click the SpeedTab button to move to the Graphs page, then choose Paste. You cannot paste a graph directly into a graph window.

The notebook where the copy is pasted receives a record of where the original data is located. (The data is not physically copied to the target notebook; instead it is *linked* to the target notebook.) Each time you open the target notebook, Quattro Pro checks whether the source notebook is open. If it is not, a dialog box with three options, Open Supporting, Update References, and None, appears. See Loading Supporting Data for explanations of these options.

Note: If you paste a graph onto the notebook page, then change your mind and choose Cut or press Delete, you have not erased the graph from the notebook. The graph still exists in a graph window. You must use Graph|Delete, or select the icon on the Graphs page and choose Cut, to delete the graph entirely.

Inserting an Existing Graph on a Page

To place any graph in the notebook as a floating graph on a spreadsheet page,

1. Go to the page where you want the graph to appear.
2. Choose Graph|Insert.
3. Select a graph from the list that appears.
4. Choose OK. The mouse pointer changes to a miniature graph.
5. Click the upper left corner of the area where you want the graph to appear, or drag the mouse to specify the size and placement of the graph.
6. Release the mouse button.

If the graph appears to have too much background area, change the Aspect Ratio to Floating Graph.

You can insert the same graph in more than one place in the notebook. Any time you make changes to a graph, all floating graphs associated with that graph reflect the changes.

To replace a floating graph with a different graph see Source Graph.

Moving and Resizing Floating Graphs

To change the size or position of a floating graph, first click anywhere inside the frame to select it. Square handles appear at eight positions on the border of the graph.

To move the floating graph, drag it to the new position.

To resize only height or width, drag a side handle.

To change the height and width of the rectangle at the same time, drag a corner handle diagonally.

To learn about floating graph properties, see Graph Object.

How to Use Quattro Pro Help

Quattro Pro uses the Windows Help system to present an online version of its manuals. Quattro Pro Help includes all standard Windows Help features, as well as several features unique to Borland Help systems.

Object Help identifies SpeedBar buttons and other onscreen features as you work. To learn about it, click the following green, underlined text:

Object Help

To learn more about standard Windows Help features, including Help menus and buttons, click the following green, underlined text (place the mouse pointer over the text and press the left mouse button):

Standard Windows Help

Click one of the following topics for an explanation of Quattro Pro Help's unique features:

Clickable Signposts

Glossary Menu

Multiple Help Files

Clickable Signposts

The main Help Contents screen displays a clickable picture for each category of Help information. Topics in that category display the same clickable picture or signpost to the left of the topic title.

No matter where you are in the Help system, if you left-click the picture by the topic title, Help jumps to the introductory screen for the category. If you click the green question-mark icon at the top of this window, you will return to the How to Use Quattro Pro Help topic.

Glossary Menu

Quattro Pro Help adds a Glossary menu to the standard Windows Help *File*, *Edit*, *Bookmark*, and *Help* menus. To open the Quattro Pro Help glossary, choose Glossary|Definitions.

Click any word in the list to display a popup window with its definition.

Multiple Help Files

Modern Help systems are alternatives to printed documentation and may be as extensive. To ensure that Quattro Pro Help will load and exit quickly on all supported computers, we have divided it into several linked Help files. This way you don't need to load all Help information every time you want Help on a specific topic.

If you click the History button after moving from one Help file to another, all titles in the History list that are located in the former Help file will be prefixed with its file name.

Linking Notebooks

Why Use Links?

A notebook link is a reference to a cell or block in another notebook. The reference is live; which means Quattro Pro updates the link results when the referenced data changes, as if the data were in the same notebook.

Notebook links are identical to page and block references except that they begin with the name of the notebook to link to. Quattro Pro provides commands for keeping notebook links accurate even if one of the notebooks is closed.

If you want to create a dynamic link to another application, see Creating DDE Links for details.

There are several advantages to setting up notebook links instead of multipage notebooks:

To build models that fit into memory. By dividing an application over several notebooks and accessing only the linked values from closed notebooks, you can build larger models.

To share information. By linking pertinent information in different notebooks, you can eliminate redundancy. Since Quattro Pro updates primary formulas when you make a change, your data is always up to date. And by using one formula for several notebooks, you save disk space.

To divide work among several people. If you use linked notebooks for a single application, you can divide the application's work among several people.

To preserve portability to non-3-D spreadsheets or databases. Since only one page of a notebook is translated when you save it in some other formats, you may want to link the first pages of different notebooks instead of pages within the same notebooks.

See Also

Creating Links

Translating Files

Creating DDE Links

Creating Links

Links are formulas with file names in them. The file name precedes the page reference and is enclosed in brackets; for example, `+[BUDGET]B:A15`.

There are four ways to create a notebook link, explained in these topics:

Typing Links

Pointing Out Links

Pasting Links

Naming Links

In addition, the following topics include important linking information:

Shortcuts for Linking Notebooks

Linking to Closed Notebooks

See Also

Creating DDE Links

Moving and Copying Links

Loading Supporting Data

Maintaining Links

Typing Links

To create a notebook link by typing, type the link as a page reference and cell reference preceded by the notebook name in brackets. Use this format:

`+[Drive:\Path\File name.Extension]Page:Cell reference`

For example, the following formula enters the value from cell B9 on page C in the BUDGET.WB1 file from the C:\ARCHIVE directory into the active cell,

`+[C:\ARCHIVE\BUDGET.WB1]C:B9`

The plus sign is needed only if the link is the first item in a cell entry (otherwise, Quattro Pro regards the reference as a label).

`Drive` identifies the drive containing the file. This is necessary only if the notebook you're referencing is not on the same drive as the primary notebook.

`Path` is the DOS path to the directory containing the file. This is necessary only if the notebook is not in the same directory as the primary notebook.

`File name` is the name of the file.

`Extension` is a three-letter suffix separated from the file name by a period (.). It is necessary only if the file has a different extension from the primary notebook (.WB1 or .WQ1).

`Page` is any valid page name, page range, or group name (such as C, Sales, or D..F, or YearToDate).

`Cell reference` is any valid cell address, pair of block coordinates, or block name. However, block coordinates or a block name are valid only if the link contains an @function to operate on the block. The link in the example above (beginning with +) simply links the active cell with the referenced cell.

You can enter all parts of the link in either uppercase or lowercase. Don't include blank spaces in any part of the reference.

If you keep all linked notebooks in the same directory and use the same extension for each, you only have to enter a file name for links. This is because the default drive, path, and extension are the same as the primary notebook. Also, if you move all linked notebooks to a different directory, you won't need to edit the references.

See Also

Pointing Out Links

Pasting Links

Naming Links

Pointing Out Links

When entering formulas, it's easier to point to a block rather than to type its address.

To point out a link,

1. If the notebook you want to reference isn't open, use File|Open to access it.
2. After typing +, or at the appropriate place in the formula, activate the target notebook you want to reference. You can do this in one of these ways:


Click anywhere in the notebook.


Open the Window menu and choose the window from the numbered list at the bottom of the menu. You can click Window in the menu bar or use the shortcut Alt+W.


Press the Next Window key, Ctrl+F6, until you're in the notebook you want.

3. By default, the last-selected cell in the target notebook appears in the formula. To change the reference to another cell or block, point elsewhere in the notebook.
4. If there is more to the formula, finish typing it. Press Enter.

The selector returns to its original place in the primary notebook and enters the block (along with the linked file's name) in the input line.

If you're entering an @function (such as @SUM) that allows multiple block references, you can enter a comma after the first block reference (which returns you to the primary notebook), then point to the next block in any notebook. After the last one, type) (close parenthesis) to complete the @function, then press Enter to complete the formula. For example, your formula might look like this:

```
@SUM([NOTEBK1]A:A1,[NOTEBK2]A:B1)
```

See Also

Typing Links

Pasting Links

Naming Links

Loading Files

Using Windows

Pasting Links

The quickest way to link cells between notebooks is with Edit|Paste Link. After you copy a block to the Clipboard, using Paste Link in another notebook (or even in the same notebook) creates an individual link to each cell in the copied block.

To create these links,

1. Select the block of cells to which you want to link. The selection can even be a noncontiguous block.
2. Click the Copy button in the SpeedBar.
3. Select the upper left cell of the destination block for the links.
4. Choose Edit|Paste Link.

Each pasted cell contains a formula beginning with a plus sign followed by


the notebook name in brackets (if different from the active notebook)


the page name and a colon (if different from the active page)


the cell address

See Also

Typing Links

Pointing Out Links

Naming Links

Naming Links

If you often link to a certain block, you can store the reference in a named block (either in the active notebook or in a separate one) to simplify the process. This offers several advantages:

To reference a different file or block, you only need to edit the reference in the named block instead of editing each individual link.

If the link statement is long, you can reference a short named block instead of typing the entire link.

When you move a named block in a supporting notebook while the primary notebook is closed, the primary notebook is updated to reflect the change the next time you open it. With links that refer to cell addresses, the primary notebook isn't updated.

For example, suppose you often reference the total of cells F12..F36 on page B in the file BUDGET. Instead of typing `@SUM([BUDGET]B:F12..F36)` wherever you want that sum, you can name a cell TOTAL, and enter `@SUM([BUDGET]B:F12..F36)` in it. This way, wherever you want that sum, just enter +TOTAL instead of typing the full link.

See Also

Typing Links

Pasting Links

Pointing Out Links

Shortcuts for Linking Notebooks

If you're consolidating information from several notebooks that have the same layout, you can use wildcards to link to the same place in all of them.

Choose File|Open to open the primary notebook and all its supporting notebooks. Then, close all notebooks you don't want to reference. To reference corresponding values in all open notebooks, use a wildcard character in place of the notebook names in the link. See Wildcard Link Examples for sample links.

As soon as you finish entering the link, the formula is revised to include specific file names. This means that if you open new notebooks after using a wildcard in a link, they aren't referenced in the formula.

You can use ? and * wildcards in links. As in DOS, a question mark (?) stands for any single character, and an asterisk (*) stands for any number of characters. See How Wildcards Work in Links for examples.

See Also

Creating Links

Using Workspaces

Loading Files

Wildcard Link Examples

These examples show wildcard characters used in notebook links:

Link	Effect
`@SUM([*]A:A1)`	adds the values in A1 in page A of all open notebooks (except the active one), and places the result in the active cell.
`@AVG([*]BUDGET:B3..C10)`	finds the average of all values in block B3..C10 in the page named BUDGET of each open notebook.
`@SUM([*]TOTALS)`	sums the values of all cells in the block TOTALS in any open notebooks with a block named TOTALS. These blocks need not be the same size, or in the same relative positions.

See Also

Creating Links

Shortcuts for Linking Notebooks

How Wildcards Work in Links

You can use ? and * wildcards in links. As in DOS, a question mark (?) stands for any single character, and an asterisk (*) stands for any number of characters. This table shows how different wildcard characters work in notebook links:

Link syntax	Effect
`[]`	Looks in the active notebook.
`[*]`	Links to all open notebooks.
`[AB*]`	Links to all open notebooks whose names begin with AB.
`[A*B]`	Links to all open notebooks whose names begin with A and end with B.
`[A?B]`	Links to all open notebooks with three-character names that begin with A and end with B.
`[AB???]`	Links to all open notebooks with five-character names that begin with AB.

See Also

Creating Links

Shortcuts for Linking Notebooks

Linking to Closed Notebooks

A notebook doesn't have to be open for you to link to it. You can type a link to a closed notebook. When you open a primary notebook, you can also choose Update References from the dialog box that appears. This loads only the linked values from supporting notebooks, leaving the supporting notebooks closed. This is useful if all your linked notebooks won't fit into memory.

Linking closed notebooks has one disadvantage: You can't update closed files. For best results, structure linked notebooks hierarchically, with links that flow only one way. Then you can work on notebooks one at a time. As long as you start with the notebook at the bottom of the hierarchy and proceed upward (choosing Update References for each notebook you open), the notebook values will always be correct.

See Also

Loading Supporting Data

Creating Links

Shortcuts for Linking Notebooks

Moving and Copying Links

These rules apply when you copy and move notebook links:

When you copy a link, relative cell references in the formula adjust like other cell addresses: relative cell references change to reference position, while absolute cell references stay the same.

When you copy a link from one notebook to another, the link remains, even if you copy it into the notebook it references. However, if you move a link into the notebook it references, Quattro Pro removes the link, leaving only the page and block reference.

If you move a formula to another notebook without moving the cells it references, Quattro Pro creates a link to the original notebook.

When you move a block in a supporting notebook, any links to it in open primary notebooks adjust accordingly. However, links in closed notebooks are not updated. To avoid this problem, use block names in links when possible. Each time you open a primary notebook, block names that have been moved are updated.

See Also

Copying Formulas

Creating Links

Maintaining Links

Using Named Blocks

Loading Supporting Data

When you open a file that contains links, Quattro Pro checks whether all supporting notebooks are open. If not, a dialog box with these options appears:

Open Supporting opens all supporting notebooks. If those notebooks contain links to closed notebooks, those notebooks open, too, until all supporting notebooks are open.

Update References accesses linked values in closed supporting notebooks without opening them. This takes up less memory than opening the notebooks. If you later decide to open the supporting notebooks, choose Tools|Update Links|Open Links.

None temporarily replaces links to closed supporting notebooks with the value NA (not available). This lets you examine or edit notebooks when you don't need the linked values. If you later want to replace the NA placeholders with the actual link values, use Tools|Update Links|Refresh Links or Open Links to update values, or open the supporting notebooks, respectively. You can use these commands at any time.

See Also

Creating Links

Maintaining Links

Maintaining Links

Use the Tools|Update Links command to maintain notebook links. You can


Open any or all supporting notebooks linked to the current notebook.


Refresh links to recalculate data in the active notebook based on data in any or all closed supporting notebooks.


Delete all link references to any or all supporting notebooks.


Change all links that reference one supporting notebook to another.

See Also

Creating Links

Moving and Copying Links

Opening Supporting Notebooks

If the active notebook contains links to one or more closed notebooks, you can use Tools|Update Links|Open Links to open any or all supporting notebooks:

1. Choose Tools|Update Links|Open Links. Quattro Pro displays a list of unopened notebooks referenced in the active notebook.
2. Select the notebook you want to open. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).
3. Choose OK. Quattro Pro opens the selected notebooks. The active notebook remains active.

Tools|Update Links|Open Links opens only notebooks that are directly linked to the active notebook. If additional notebooks are linked to those directly linked, you need to activate the directly linked notebooks and choose Open Links again. For example, suppose Notebook A refers to Notebook B, which refers to Notebook C. Activating Notebook A and choosing Open Links lists only Notebook B. To open Notebook C, activate Notebook B and choose Open Links.

See Also

Creating Links
Updating Link Values
Removing Links
Changing Notebook Links

Updating Link Values

If a notebook is linked to values in closed notebooks, you can use the Tools|Update Links|Refresh Links command to access values in any or all supporting notebooks:

1. Choose Tools|Update Links|Refresh Links. Quattro Pro displays a list of unopened notebooks referenced in the active notebook.
2. Select the notebook whose values you want to access. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).
3. Choose OK. Quattro Pro reads linked values from the selected notebooks into the active notebook.

Tools|Update Links|Refresh Links is similar to the Update References option offered when you access a primary notebook, except you can use it at any time, and you can specify one notebook at a time from which to access values.

Refresh Links is also useful when you're working on a network. While you're using one notebook, others can be working on supporting notebooks (which remain closed to you). If people announce over the network when they save or close a file, you can use Refresh Links to access updated values in the primary notebook.

See Also

Loading Supporting Data

Creating Links

Removing Links

Changing Notebook Links

Removing Links

You can delete all links to any or all supporting notebooks with one command:

1. Choose Tools|Update Links|Delete Links. Quattro Pro displays a list of notebooks referenced in the active notebook.
2. Select any notebook whose links you want to delete. You can select multiple notebooks by holding down the Ctrl key (as you click individual notebooks) or the Shift key (as you click the last notebook in a range to be selected).
3. Choose OK. Quattro Pro replaces all link references to the selected notebooks with ERR values, but doesn't remove any formulas containing ERR values. You can edit them to replace the ERR values with valid references.

Caution: Edit|Undo can't restore deleted links.

See Also

Creating Links

Moving and Copying Links

Updating Link Values

Changing Notebook Links

Changing Notebook Links

You can use Tools|Update Links|Change Link to unlink the active notebook from a supporting notebook and link it to another. This is useful after you rename a notebook.

To switch links from one notebook to another:

1. Choose Tools|Update Links|Change Link. Quattro Pro displays a list of notebooks referenced in the active notebook.
2. Select the notebook whose links you want to change. In the To edit field, type the name of the notebook you want to change links to.
3. Choose OK. In the active notebook, all link references to the selected notebook change to the new notebook name.

Note: The new supporting notebook must have the same layout as the previous one because the same relative cells are referenced. If it doesn't, formulas that reference linked cells may show errors or produce meaningless values. To avoid this, enter all links as block names so the correct values are included.

See Also

Creating Links

Updating Link Values

Removing Links

Using Macros

See Also

Macro Command Descriptions

What Is a Macro?

A macro is a sequence of commands Quattro Pro runs automatically. When running a macro, Quattro Pro performs the actions listed in it. You can store macros in the notebook they apply to or in a macro library file for use by other notebooks. You can also attach macros to buttons, and you can run macros from another application using Dynamic Data Exchange.

Macros can reproduce the behavior of keys on the keyboard, mouse actions, and menu commands. Special macro commands can perform actions such as prompting the user for input, looping a macro repeatedly, or controlling other Windows applications. Use them to automate complex or repetitive command sequences (like printing a standard report), to enter frequently used labels with a keystroke, or to build complete applications for use by people with little Quattro Pro experience.

See Also

Creating Macros

Running Macros

Using Macros

Macro Command Descriptions

Creating Macros

Quattro Pro offers two main ways to create macros:

Recording Macros as you perform a task

Typing Macros directly into a notebook page

It is often helpful to combine these methods. You can record a command sequence, then paste it into a macro you are typing.

See Also

Planning Macros

Using Macros

Macro Command Descriptions

Planning Macros

Whether you record or type a macro, try these tips:

Plan your macro carefully. Making it efficient saves you the trouble of modifying it later to add new functionality. If it involves menu commands, step through them first, writing down each action involved. If it involves special macro commands, know exactly what you're going to enter as the command arguments. Keep track of the selector after each macro command; its position can affect the macro's behavior.

Try sketching a flow chart of your macro before entering it. A flow chart is a diagram showing the actions and decisions a macro steps through while running. Boxes in the chart contain an action the macro performs. Diamonds contain decisions the macro must make to determine the next step; for example, a cell achieving a certain value or the selector reaching a specific location. Arrows show where to go after a step is completed. Arrows exiting a diamond have a decision listed next to them dictating the next step. If a decision isn't listed, the step at the end of the arrow is always the next step. Flow charts don't show actual macro commands; they show the planning and logic behind the macro.

See Also

Creating Macros

Using Macros

Macro Command Descriptions

Recording Macros

With Quattro Pro, you can record actions as you perform them. Recording converts actions into macro commands and stores them as labels in a block. To record a macro,

1. Choose Tools|Macro|Record.
2. Select the block to store the macro in. If there's no chance of overwriting data in cells below, select a single cell; Quattro Pro fills the cells below as necessary. Selecting a block larger than a cell confines recorded commands to that block only, and stops recording when the block is full.
3. Choose OK to place Quattro Pro in recording mode (noted by the REC indicator on the status line).
4. Perform your task. Quattro Pro records each action, writing it as one or more macro commands in the first column of the block previously specified.
5. Choose Tools|Macro|Stop Record to finish recording. The macro is now stored in the block.
6. Move to the first cell of the macro, and choose Block|Names|Create. Enter a name for the macro and choose OK.

For best results, follow the suggestions in Macro Recording Tips.

By default, Quattro Pro records macros as command equivalents with normal cell references (like A1..B26). If you like, you can change recording mode to record keystroke by keystroke and can record with relative cell references. Choose Tools|Macro|Options to change the default settings.

See Also

Running Macros

Using Macros

Macro Command Descriptions

Macro Recording Modes

By default, Quattro Pro records actions as command equivalents.This recording method is called logical recording.

To record macros keystroke by keystroke, choose Keystroke from the Tools|Macro|Options dialog box. Macros recorded in keystroke mode are less efficient than logical macros and do not include mouse actions.

Using keystroke macros to emulate menu operations isn't recommended:


You can't easily tell what the macro does by reading it.


The macro works only in menu systems where shortcut keys are exactly the same as those in the system for which the macro was written. For example, if /BC is used in a macro to perform Block|Copy, it works incorrectly in menu systems where /BC performs another action (say, Block|Combine).


Because each keystroke must be processed as a separate macro command, the macro runs more slowly.

You should use keystrokes only to develop macros that must run under 1-2-3. Otherwise, use command equivalents, special macro commands that aren't dependent on the menu system in use.

See Also

Command Equivalents

Recording Macros

Using Macros

Macro Command Descriptions

Recording Relative Cell References

By default, addresses in macro commands use standard cell addressing like A1..C6. By choosing Relative from the Tools|Macro|Options dialog box, you can record macro commands that use relative references.

Relative references are cell addresses specified as an offset from the cell selector. For example, the cell selector's relative reference is []P(0):C(0)R(0). The first piece of the reference, ([]), specifies that you're referring to the active notebook. The second piece, P(0):, is the number of pages from the selector (in this case, zero); it's optional (except when selecting an entire row or column, which is discussed next). The third piece, C(0), is the number of columns from the selector. The final piece, R(0), is the number of rows from the selector. The cell below the selector is []C(0)R(1), the cell to the right of it is []C(1)R(0), and the cell beneath the selector (on the next notebook page) is []P(1):C(0)R(0).

Caution: If [] doesn't precede a relative reference, the relative reference is offset from the cell containing the macro command, not the cell selector. For example, if `{BLANK C(1)R(0)}` was stored in A:A16, running it would erase the cell to the right of it (A:B16).

You can precede a relative reference with a colon (:) to affect the same cell, but on the active page. For example, if `{BLANK :C(1)R(0)}` was stored in A:A16, running it when page B is active erases the cell B:B16.

You can omit C( ) or R( ) to select entire rows or columns. For example, `{SELECTBLOCK []P(0):C(0)}` selects the column containing the active cell; `{SELECTBLOCK []P(0):R(0)}` selects the row containing the active cell. To select the three columns to the right of the active cell, use `{SELECTBLOCK []P(0):C(1)..C(3)}`.

Relative references can use negative or positive offsets. For example, if the selector is in cell A2, you could use []C(0)R(-1)..C(2)R(1) to specify the block A1..C3.

Using relative references increases macro portability, but hinders readability, since it is harder to track which blocks the macro affects. If your macros use @CELLPOINTER frequently, try relative references.

See Also

Accessing Other Notebooks in Macros

Recording Macros

Using Macros

Macro Command Descriptions

Macro Recording Tips

Here are some handy tips to make macro recording easier:

Use the shortest steps possible to perform your task. The longer a recording session goes, the more errors can result.

Use any method available to choose menu commands when recording. The resulting macro reads the same when recorded logically.

You can't record a macro that pauses for user input. If you want the macro to pause, edit it after recording it. (See Typing Macros for details.)

See Also

Recording Macros

Using Macros

Macro Command Descriptions

Typing Macros

You can type macros directly into cells as labels instead of recording them. This method requires precision (one incorrect keystroke can invalidate a macro). Some macros can only be typed, such as those that prompt for user input or that use Dynamic Data Exchange (DDE) to communicate with other Windows applications.

To type a macro,

1. Move to the first cell of the block where you want to store the macro.
2. Enter the macro commands that perform your task. You can enter multiple macro commands in the same cell, or in the cell below the last command entered. If using more than one cell, be sure the entries proceed downward (A1, A2, A3) with no empty cells between them. Each cell must contain a label or a text formula.
3. Leave an empty cell, {QUIT}, or {RETURN} at the end of the macro. Otherwise the macro interprets labels below it as macro commands.

When you finish entering a macro, name its first cell using Block|Names|Create. Then you can run the macro by choosing its name from the Tools|Macro|Execute dialog box or pressing a Ctrl key shortcut (see Running Macros for details). Block names make macros more readable and easier to remember. Give your macros names reflecting their task. For example, PERCENT could be the name of a macro that changes the numeric format of the current cell to percentage.

Caution: When naming macros, avoid names that duplicate macro commands. Otherwise the macro commands become invalid. For example, if you name a macro READLN, which is the name of a predefined macro command, and then try to read a line from a file into cell A6 using {READLN A6}, Quattro Pro assumes you're running the READLN macro instead of running the predefined {READLN} command.

See Also

Creating Macros

Macro Typing Tips

Macro Syntax and Arguments

Entering Macro Commands

Self-Modifying Macros

Macro Typing Tips

Here are some suggestions to follow when typing macros:


Use Tools|Macro|Record to create sections of a macro that consist of fixed keystrokes and menu choices. This speeds entry time and reduces errors.


Add comments to your macros to make them easier to understand and debug.


List block names to the left of the macros or cells they reference; then you can use Block|Names|Labels to create the block names quickly.


Place macros in an area unaffected by macro commands. For example, don't place macros to the right of a block where rows will be inserted or deleted. The ideal place for a macro depends on what operations it performs. If you want to use a macro with many notebooks, place it in a macro library (see Using Macro Libraries for more information). If only one notebook uses the macro, add a page named Macros to that notebook and place the macro there.

See Also

Typing Macros

Using Macros

Macro Command Descriptions

Macro Syntax and Arguments

Macro commands are like @functions in that they have specific grammatical rules, or syntax. The syntax for macros is fairly straightforward:

{*COMMANDNAME Argument1,Argument2,Argument3...*}

COMMANDNAME is the exact name of the command. Arguments are values providing instructions to the command. Not all macro commands require arguments; when they do, they require a specific type of information. Some examples of macro commands are

```
{Query.Criteria_Table B27..B29}
{BlockCopy A1,A2..A37}
{Search.Find "3rd Quarter Profits"}
{BEEP 3}
{GETNUMBER "How old are you?",AGE}
{CONTENTS E15,F15,15}
```

Many command equivalents contain a period (.) in their command name.

See Macro Syntax Rules and Macro Command Arguments for descriptions of allowable macro structures and arguments.

Macro Syntax Rules

The syntax rules for macro commands are as follows:

- The command must begin and end with braces { }.
- There must be a space between the command name and the first argument. For example, `{GETNUMBER"Number?",A1}` results in a syntax error; `{GETNUMBER "Number?",A1}` works correctly.
- Separate multiple arguments with commas. By changing the Punctuation option of the International property in the Application Object Inspector, you can use semicolons or periods instead.
- Arguments must be the correct type; if a string is required, the argument must be a valid character string or a syntax error occurs.
- If an argument contains spaces or punctuation, enclose it in quotation marks. For example,

  `{GETLABEL "Hello, world",A4}`

  is allowed, but not

  `{GETLABEL Hello, world,A4}`
- You must enter the entire macro command in a single cell.
- You can enter the command in any case (upper or lower).
- You can include more than one macro command in a cell. For example, `{BEEP}{GETLABEL "Hello, world",A4}{QUIT}`.

Because macro commands are labels, Quattro Pro won't recognize a syntax error when you type it in; an error occurs when you try to run the macro. To save debugging time, pay careful attention to the format of macro commands as you enter them and record macros whenever possible.

Macro Command Arguments

Arguments in macro commands require specific information to be supplied with the command. There are four types of arguments: numbers, strings, locations, and conditions.

Number arguments require any numeric value, entered as


an actual number (such as `2` or `0.45`)


a formula resulting in a number (such as `A3*15`)


a cell address or named block containing a numeric value or formula (such as `C10`, where C10 contains a valid number or formula)

String arguments require a text string, entered as


an actual string in quotation marks (`"Borland"`)


a reference to a cell or named block containing a label


a comma-separated list of property or command equivalent settings, enclosed in quotation marks ("Currency,2"). If a setting in the list would normally take a single quotation mark, enter two quotation marks ("Prefix,""Windows Default"",No"). If a setting contains spaces or punctuation, enclose it in two sets of quotation marks.


a formula resulting in a label, such as `@UPPER("hello")` (See Types of Formulas for a discussion of text formulas. See String @Functions for a discussion of string @functions.)

Location arguments require a reference to a cell or a block. The reference can be


a block name


coordinates for a block containing one or more cells; for example, `A1`, `A1..A4` or `A..B:C4..D22`


the relative reference (see Recording Relative Cell References) of a block containing one or more cells; for example, `[]C(0)R(0)`, `[]P(-2):C(0)R(22)`, or `[]C(0)R(0)..C(3)R(10)`


coordinates for a noncontiguous block, enclosed in parentheses; for example, `(A1,B1..B7,C1..C7)` or `(A1,B:C27..C52)`


a label or text formula resulting in any of these options; for example, `+"A"&"2"`, which results in A2

Condition arguments require a logical expression; that is, a formula that can be evaluated as either true or false; for example, `+C4 > 500`.

Some commands accept a combination or choice of argument types. For example, {LET} stores either a label or a number in a cell, depending on the argument type.

Caution: Unlike @functions, block references in macros are not updated when the macro is copied. If you move the contents of a cell, or insert or delete a row or column, the macro reference might be wrong. For this reason, use block names whenever possible. Quattro Pro updates block names to reference the correct location.

Entering Macro Commands

Each macro is a sequence of macro commands. These commands include command equivalents (which emulate menu commands) and unique functions you can't do directly in Quattro Pro, such as sounding the computer's speaker, pausing for a fixed length of time, or prompting for input. Macro Command Index describes each macro command.

You can enter macro commands by typing them, using the Macros button (on the notebook SpeedBar when the input line is active), or using the Macros key (Shift+F3). This displays a menu of command categories (see Macro Command Descriptions for descriptions of each category and their commands).

To insert a macro command with the Macros key:

1. Press Shift+F3 (or click the input line and choose the Macros button) to display a list of macro command categories.
2. Choose the category containing the desired command.
3. Choose the command to insert.

Quattro Pro inserts the macro command name on the input line. When a list of categories or commands is displayed, you can search for an item in the list by pressing F2 and typing the first few letters of the item's name. Quattro Pro highlights the first item in the list that begins with the text you type.

About Command Equivalents

The commands in this category perform operations normally done with menus or dialog boxes. Options normally set in a dialog box are passed as arguments to a command equivalent, like passing them to a subroutine. Command equivalents make macros easier to read and understand, and let the macro run in any Quattro Pro menu system.

Menu commands containing many settings (such as File|Print and Block|Fill) have a set of command equivalents that emulate their operation. The next example shows a macro that emulates Edit|Search And Replace:

```
search_string       Fri

\s                  {GETLABEL "Search For? ",B1}
                    {Search.Reset}
                    {Search.Block ""}
                    {Search.Find +search_string}
                    {Search.ReplaceBy +search_string}
\n                  {Search.Next}
```

Notice that each command equivalent sets one option in the Edit|Search and Replace dialog box. You don't have to set each option every time; if one of the command equivalents is omitted, the default setting is used. The final command ({Search.Next}) performs the fill operation. You can use command equivalent names with @COMMAND to find current Quattro Pro settings. For example, `@COMMAND("BlockFill.Series")` returns the current setting of Series in Block|Fill.

Menu commands with a few settings (such as Block|Move and Block|Copy) have one command equivalent to emulate them. Command equivalents without a period are typically of this type. For example, the following macro command emulates Block|Copy:

```
{BlockCopy A1,A2..A36}
```

If you specify a command equivalent with a question mark (?) after the command name, the macro command displays a dialog box that the user can manipulate. If the name of the command equivalent contains a period, use only the part of of the command name that precedes the period. For example, `{BlockFill?}` displays the Block|Fill dialog box for the user to manipulate. When the user chooses OK, the fill occurs and macro resumes running.

If you specify a command equivalent with an exclamation mark (!) after the command name, the macro command displays a dialog box that the macro can manipulate. You can make the dialog box revert to user control at any point in the macro using the command {PAUSEMACRO}.

Caution: Make sure that a dialog box is displaying or that Quattro Pro is in INPUT or FIND mode when the macro command runs {PAUSEMACRO}; otherwise the macro waits until it's stopped by Ctrl+Break.

The next example shows a macro that uses {PAUSEMACRO} and {RestrictInput} to display a small form for the user to enter a name and phone number.

```
new_name        Nick Piercherd
new_phone       (804) 555-2315

get_recor       {BLANK A:E28..E29}
  d
                {RestrictInput.Enter A:D28..F29}
                {PAUSEMACRO}
                {IF #NOT#@ISSTRING(E28)}{RETURN}
                {IF #NOT#@ISSTRING(E29)}{RETURN}
```

```
{LET new_name,E28}
{LET new_phone,E29}
```

About DDE Macro Commands

Dynamic Data Exchange (DDE) lets Windows applications communicate with one another. Use {INITIATE} to open a channel of communication with another application. This is called initiating a conversation. Every DDE conversation consists of a client and a server. The application initiating the conversation is the client; the other application is the server. Once you've initiated a conversation, you can use {POKE} to send data, {REQUEST} to receive data, or {EXECUTE} to run macros in the server. {TERMINATE} ends the conversation. The {EXEC} command lets you run other Windows applications and execute DOS commands.

The data source that a DDE conversation connects with in the server application is called a topic. Many DDE applications support the topic System and items available from it.

Other DDE applications can call Quattro Pro as their server; to do so, use the server name QPW and specify System or the file name of an open notebook as the topic. When a notebook is the topic, you can enter cell addresses or block coordinates as the item to request, or run Quattro Pro macro commands. When System is the topic, you can run Quattro Pro macros or request these items:

Item	Returns
SysItems	A list of the items you can request from System.
Topics	A list of topics currently available from the server application.
Status	The current status of the application. In Quattro Pro this is the text of the status line indicator (READY, WAIT, LABEL and so on).
Formats	A list of Clipboard formats supported by the application.
Selection	The coordinates of the active block.

Note: You can also request any property from Quattro Pro. The property must be enclosed in parentheses and quotation marks, for example, "(Application.Display)".

About Object Macro Commands

Object commands can create Quattro Pro objects, change their property settings, or move them to new positions. Object commands can create three types of objects:

Drawn objects are objects in a graph window that you'd normally create using the graph SpeedBar. When you're creating a drawn object with a macro command, the position for the object is specified by stating how many pixels it should appear from the upper left corner of the graph background. A pixel is the smallest dot that Windows can display on your screen. You can create drawn objects with {CREATEOBJECT}. The following command activates a graph window and creates a line near the upper left corner of the graph:

```
{GraphEdit "PROFITS"}
{CREATEOBJECT "Line",0,0,25,25}
```

Dialog controls are objects in a dialog window. When creating a dialog control using a macro command, the position is also specified by stating how many pixels it should appear from the upper left corner of the dialog window. You can create dialog controls with {CREATEOBJECT}. For example, this command creates a push button in the active dialog window:

```
{CREATEOBJECT "Button",43,41,58,77}
```

Floating objects are SpeedButtons and floating graphs. Unlike graph objects and dialog controls, the position of a floating object is specified as an offset from a cell in the notebook. The offset is specified in twips; each twip is 1/1440th of an inch. You can create floating objects with {FLOATCREATE}. For example, the following macro command creates a SpeedButton that's half an inch from the left edge of A:A1:

```
{FLOATCREATE "Button",A:A1,720,0,A:B2,720,360,"Bt1"}
```

In this example, A:A1,720,0 specifies that the upper left corner of the button is in A:A1, and is 720 twips from the left side of the cell (half an inch), as well as 0 twips from the top of the cell. A:B2,720,360 specifies that the lower left corner of the button is in A:B2, 720 twips from the left side of A:B2, and a quarter inch (360 twips) from the top of A:B2.

After creating an object with a macro command, use {SETPROPERTY} to set its name or {GETPROPERTY} to store its name in a cell; then you can change its properties with macro commands at any time.

Selecting, positioning, and sizing objects

After creating a dialog control or floating object, they're selected, so you can reposition them or change their property settings. There are three commands that select Quattro Pro objects:

1. You can use {SELECTBLOCK} to select a block and change or read its properties.
2. You can use {SELECTOBJECT} to select graph objects or dialog controls. When a graph object or dialog control is selected, you can use {MOVETO} and {RESIZE} to move and resize it.
3. You can select a floating object with {SELECTFLOAT}. When a floating object is selected you can use {FLOATMOVE} and {FLOATSIZE} to move and resize it.

Changing/reading property settings

You can use {SETPROPERTY} and {SETOBJECTPROPERTY} to change property settings of Quattro Pro objects. For example, the following macro selects a block and changes its text color:

```
{SELECTBLOCK A:A1..C22}
{SETPROPERTY "Text_Color",5}
```

You can also change a property setting without selecting the block using the following command:

```
{SETOBJECTPROPERTY "A:A1..C22.Text_Color",5}
```

You can use {GETPROPERTY} and {GETOBJECTPROPERTY} to read property settings. {GETPROPERTY} reads settings of the selected object; {GETOBJECTPROPERTY} lets you read property settings without selecting an object.

Accessing Other Notebooks in Macros

You can use links in macros just as you would in formulas. Macros can run macros in other notebooks, access data from them, or store user input in them.

If you run a macro that's in a different notebook, such as a macro library, the macro behaves as though it were stored in the active notebook; block coordinates entered in dialog boxes refer to the active notebook. However, coordinates stored in the macro as arguments to a macro command refer to the macro library. For example, `{BRANCH A1}` branches to cell A1 of the macro library, not the active notebook. To refer to a block in a different notebook, use linking. For example, the following command branches to page B, cell A1 of the notebook MACROS:

```
{BRANCH [MACROS]B:A1}
```

To refer to a block in the active notebook, precede its coordinates with empty brackets. For example, `{BRANCH []A1}` branches to cell A1 in the active notebook.

See Also

Linking Notebooks

Relative references

Using Macro Libraries

Using Macro Subroutines

A subroutine is a macro stored separately from the macros using it. You name the subroutine just like other macros. You can call the subroutine from another macro, and Quattro Pro runs the subroutine's macro commands. After running the subroutine, the command immediately following the call (in the main macro) runs. Using subroutines makes the macro more readable and easier to debug.

To call a subroutine from within a macro, type its name inside braces. For example, `{go_right}` calls the subroutine named go_right.

You can pass arguments to a subroutine for use by its commands. These arguments are stored in cells referenced by the subroutine. For example,

```
{set_cost C10,36}
```

calls the subroutine set_cost and passes two arguments (C10 and 36) to it. In order for the subroutine to know what to do with the arguments, you must define them within the subroutine using {DEFINE}, which tells Quattro Pro where to store the arguments and whether they should be interpreted as values or labels.

Whenever you call a subroutine, Quattro Pro stores the return point in an internal list called a stack. One return point on this stack clears when the subroutine encounters a {RETURN} command or empty cell. If you don't clear all of these locations, the stack fills up, causing the error Too Many Nesting Calls.

Use {BRANCH} instead of a subroutine when a macro


doesn't need to pick up where it left off after calling the subroutine


needs to return to a different point in the macro than the point right after the subroutine call


calls the main macro as a subroutine

{BRANCH} can move to or return from any cell in a macro.

See Also
{Subroutine}

Self-Modifying Macros

You can use macro commands and text formulas to create self-modifying macros (also called dynamic macros) that alter themselves while running. For example, to save the active notebook under the name ACCTx.WB1, where x is a value stored in cell A5, enter the following text formula into a cell:

```
+"{FileSaveAs ""ACCT"&@STRING(A5,0)&".WB1""}"
```

The result of this formula runs as a macro command. For example, if cell A5 contains the value 5 in the previous example, Quattro Pro runs `{FileSaveAs "ACCT5.WB1"}`; if cell A5 changes to 10 and the notebook recalculates, Quattro Pro runs `{FileSaveAs "ACCT10.WB1"}`.

You can also use macro commands to change a portion of the macro while it's running. The next example shows a macro that uses {GET} to convert user keystrokes into macro actions.

```
cell_type       b

type_indic      {INDICATE @UPPER(@CELLPOINTER("type"))}
                {LET cell_type,@cellpointer("type")}
                {IF cell_type="l"}{INDICATE LABEL}
                {IF cell_type="b"}{INDICATE BLANK}
                {IF cell_type="v"}{INDICATE VALUE}
                {GET next_key}
                {IF @UPPER(next_key)="{ESC}"}{BRANCH
                  clear_indic}
next_key        {Esc}
                {BRANCH type_indic}

clear_indic     {INDICATE}
                {QUIT}
```

You can use these concepts to create powerful macros that completely change themselves based on their environment. See Types of Formulas for more information on text formulas.

Using Macro Libraries

The best way to store macros is in a macro library, a special notebook reserved for macros. You can use it to store macros you want to access from any notebook. There are many advantages to storing macros in a library:


It simplifies linked access.


It makes it easier to keep track of where the macros are.


If Quattro Pro can't find the macro you specify within the active notebook, it searches through all open macro libraries until it finds it.


The macros don't interfere with the notebook data and vice versa.


Using one set of macros for a group of notebooks saves disk space.


If there's a set of macros you do want to insert into individual notebooks, you can store them in a library, then copy the macros into notebooks as you need them.


You can have a separate set of macros for each application or type of notebook you work with.

To create a macro library,

1. Open a new notebook. (You can also use an existing notebook.)
2. Create the macros to store in it (or copy them to it from another notebook).
3. Right-click the notebook title bar and choose Macro Library.
4. Choose Yes to define the active notebook as a macro library.
5. Choose OK to save the change.
6. Choose File|Save to save the macro library.

Note: It's a good idea to keep only one macro library open at a time. If two open libraries contain a macro with the same name, it's difficult to predict which macro will run. Quattro Pro searches only open macro libraries for macros; closed libraries are ignored.

Use Window|Hide (or the macro command that emulates it, {WindowHide}) to hide your macro libraries. This prevents novices from altering them.

See Accessing Other Notebooks in Macros for a discussion of how addressing is handled in a macro library.

See Also

Linking Notebooks

Running Macros

Relative References

Running Macros

To run a macro,

1. Choose Tools|Macro|Execute.
2. If desired, use Macro Library to specify the name of the notebook containing the macro. This displays a list of macros available under Macros.
3. Choose the name of the macro to run from Macros or choose Location and enter the macro's block coordinates. If the macro is in another notebook, use full linking syntax (for example, [LIBRARY]A:C26). Linking isn't needed if entering a block name.
4. Choose OK to run the macro.

The MACRO indicator appears on the status line while a macro runs. When Quattro Pro runs a macro, it begins with the first cell in the macro block. It continues down through the column, interpreting everything it encounters as part of the macro, until it finds

 an empty cell

 a cell containing a value

 the {QUIT} macro command

 the {RETURN} macro command

 an invalid macro entry

At that point, the macro stops and control returns to the user. Leave an empty cell, {QUIT}, or {RETURN} after each macro in a notebook, or the macro will try to interpret labels below the macro as commands.

If the macro contains an error, Quattro Pro beeps, displays an error message, and stops the macro. You can use the debugger to pinpoint and correct macro errors.

To stop a macro while it's running, press Ctrl+Break and choose OK to clear the break message and return to Ready mode.

Note: If the Ctrl+Break key has been disabled with {BREAKOFF}, you won't be able to interrupt the macro. If you need to use {BREAKOFF} in the macro, add it as a last step after debugging the macro.

Suppressing screen redraw

By default, Quattro Pro doesn't show the actions a macro performs while it's running (this makes the macro run faster). To see the actions a macro performs, right-click the application title bar and choose None from the Macro property's Macro Suppress-Redraw option. For a complete discussion of screen suppression, see {PANELOFF} and {WINDOWSOFF}.

Running macros from Windows

You can run a macro as you load Quattro Pro by specifying its name after you enter QPW and a file name from the Program Manager or the DOS command line. For example, the following DOS command starts Windows, loads Quattro Pro, retrieves the notebook named EXPENSE, and runs the macro named \F:

```
WIN QPW EXPENSE \F
```

Only `QPW EXPENSE \F` is necessary when using the Program Manager's File|Run command. Windows applications that support Dynamic Data Exchange can make Quattro Pro run macro commands; see their documentation for more details.

See Also

About DDE Macro Commands

Attaching Macros

Using Startup and Exit Macros

Quattro Pro for DOS Macros

Attaching Macros

Many objects in Quattro Pro can have macros attached to them that run when the user performs some action (clicks a button, activates a control, changes a value). There are two ways to assign a macro to an object:

Using properties. You can use the properties of SpeedButtons and graph buttons to specify what macro should run when they're clicked. See Creating SpeedButtons and Creating a Graph Button for details.

Using link commands. Many objects in a dialog window can run macros using link commands. See Working with Link Commands for a complete discussion of object linking and the DOMACRO link command. You can also attach link commands to menu commands created by {ADDMENU} and {ADDMENUITEM}. See Making Menu Commands Act for details.

Attaching a macro to a key

If the macro you're creating is used often, you can give it a special block name that makes it easier to run: name it with a backslash (\) followed by a letter of the alphabet (A-Z). (Use the naming methods described in Typing Macros.) Then you can run the macro from the keyboard by pressing Ctrl and that letter (in either uppercase or lowercase). Because there can be only 26 of these macros (one for each letter of the alphabet), reserve them for macros you use often. Note that some of these macros may override accelerator keys like Ctrl+X and Ctrl+C.

Note: Macros assigned to Alt keys in the DOS version of Quattro Pro run in the Windows version using the Ctrl key.

See Also
Running Macros

Creating SpeedButtons

You can use the SpeedButton tool on the notebook SpeedBar to add buttons (called macro buttons or SpeedButtons) that run macros in a notebook. To create a SpeedButton,

1. Choose the SpeedButton tool from the notebook SpeedBar.
2. To create the SpeedButton in a default size, just click. The button appears, with its top left corner at the position you clicked.

   To create the control in a specific size, hold down the left mouse button and drag the pointer. As you drag, a bounding box indicates the size of the button; release the mouse button to create the SpeedButton.
3. There are two ways to change the appearance of the button:


You can paste a graphic image onto the button: copy a Windows bitmap into the Clipboard, select the SpeedButton (to select a SpeedButton, right-click it and press Esc), and choose Edit|Paste.


You can change the text appearing on the button: right-click it and choose the Properties command, choose Button Properties, then choose Label Text, and enter the text to display on the button. Choose OK to set the new button title.

4. Right-click the SpeedButton and choose the Properties command, choose Button Properties, then choose Macro. Enter the macro commands to run when the button is clicked. You can use {BRANCH} or {Subroutine} to run a macro in a notebook. Choose OK to save the macro.

See {FLOATCREATE} for information on creating SpeedButtons using macro commands.

Once a SpeedButton is created, you can right-click it, choose the Properties command, and choose Button Properties to change the properties listed in the following table:

Property	Description
Macro	Macro commands to run when the button is clicked. For example, `{BEEP}` `{SHIFT+DOWN 3}`. You can use Execute Dialog to specify a dialog box to display (discussed next).
Label Text	The text titling the button.
Border Color	The color of the button's border.
Box Type	The thickness of the button's border, and whether it has a shadow.
Object Name	The name of the object. Used in object linking to manipulate its properties.

You can use the Execute Dialog option of the Macro property to make the SpeedButton display a dialog box. Using this option replaces whatever macro is assigned to the button with a {DODIALOG} command. To display a dialog box with a SpeedButton,

1. Right-click the SpeedButton and choose the Properties command, choose Button Properties, then choose Macro.
2. Click Execute Dialog.
3. Choose the name of the dialog box to display from the Activate Dialog list box.
4. Choose Result Cell and point to a cell in the notebook. After the dialog box is closed, this cell contains 1 or 0. The value 1 means the dialog box was closed by OK; the value 0 means it was

canceled.

5. Choose Values Block and point to a block in the notebook that contains initial settings for the dialog box. When the dialog box is closed, its final settings are written back into this block. (See Setting a Control's Value for more information on the setting block.)
6. Choose OK to create the {DODIALOG} command that displays this dialog box.
7. Choose OK to save the macro.

You can make a text box in a graph run macros during a slide show (making it into a graph button) using its Graph Button property; for more information on creating graph buttons, see Creating a Graph Button.

See Also

Attaching Macros

Using Startup and Exit Macros

Creating a Graph Button

Quattro Pro for DOS Macros

Using Startup and Exit Macros

Sometimes it's helpful to run a macro each time you open or close a notebook. For example, you can "stamp" the current date and time or open another notebook after closing one. You can also run a macro each time you start Quattro Pro. For details, see these topics:

Notebook Startup Macros

Autoload Macros

Notebook Exit Macros

Application Startup Macros

Notebook Startup Macros

To create a notebook startup macro, enter the macro into a notebook and name it _NBSTARTMACRO. Each time you open a notebook, Quattro Pro looks for a macro with that name and runs it, if found. If that macro isn't found, Quattro Pro looks for the startup macro name entered as a Startup property in the application Object Inspector; \0 is the default. (See Running a Macro Automatically for information on changing the startup macro name.)

Note: To ensure that an autoload macro runs in a notebook regardless of whether it's linked to another, begin the macro with {ESC}. This cancels the dialog box that appears when opening a notebook containing links. You can use Tools|Update Links to access linked values or to load supporting notebooks after accessing the file.

See Also
Using Startup and Exit Macros

Autoload Macros

One way to run a macro each time you open a notebook is to create an autoload macro. If the notebook file doesn't contain a startup macro named _NBSTARTMACRO, Quattro Pro automatically runs any macro named \0 (backslash zero) when the notebook containing it opens. Each notebook can have one of these autoload macros. To change the name of the autoload macro,

1. Right-click the application title bar.
2. Choose Startup.
3. Choose Startup Macro and enter the autoload macro's new name.
4. Choose OK to save the change.

When the autoload macro is renamed, the old autoload macro name no longer runs automatically in any notebook. For example, if you set the autoload name to `AUTOSTART`, macros named `\0` would no longer run automatically; they'd have to be renamed `AUTOSTART`.

Note: To ensure that an autoload macro runs in a notebook regardless of whether it's linked to another, begin it with {ESC}. This cancels the dialog box that appears when you open a notebook containing links. You can use Tools|Update Links to access linked values or to load supporting notebooks after accessing the file.

See Also
Using Startup and Exit Macros

Notebook Exit Macros

Notebook exit macros run after you give the command to close a notebook. To create a notebook exit macro, enter the macro into a notebook and name it _NBEXITMACRO. When you close a notebook containing a macro with that name, Quattro Pro branches to the macro before closing. It is up to the macro to make sure any ongoing operations are completed and ready for exit before it closes the notebook.

See Also
Using Startup and Exit Macros

Application Startup Macros

There are two ways to launch a macro each time you start the application. You can specify an autoload file and the startup macro it contains in the application Object Inspector (see Autoload Macros), or you can define a startup macro in QPW.INI. The .INI method is useful for linking add-ins and performing other procedures at startup without loading an autoload file.

To use the .INI method,

1. Open QPW.INI in a text editor (make a backup copy first).
2. Type the expression InitMacro= followed by a series of macro statements (up to 255 characters). To launch a longer macro, use the {BRANCH} command following InitMacro=.
3. Save QPW.INI as an unformatted text file.

For example, this statement loads two add-ins, DLL1 and DLL2:

```
InitMacro={DLL.LOAD DLL1}{DLL.LOAD DLL1}
```

InitMacro expressions can be used with {ADDMENUITEM}, {ADDMENU}, {MESSAGE}, and similar macro commands to change the application interface or deliver information to users when they start Quattro Pro.

See Also
Using Startup and Exit Macros

Quattro Pro for DOS Macros

If you've written macros using menu-equivalent commands such as {/ Block;Copy} in the DOS version of Quattro Pro, you can run them in Quattro Pro without modification. (See the Quattro Pro for DOS User's Guide for more information on menu-equivalent commands.)

To run Quattro Pro for DOS macros written with keystrokes, such as /EC,

1. Right-click the application title bar.
2. Choose Slash Key and select Quattro Pro - DOS.
3. Choose OK to save the change and return to the active window.

This also lets you press the slash key (/) to display a menu system that's keystroke compatible with Quattro Pro for DOS.

Macros assigned to Alt keys in the DOS version of Quattro Pro run in the Windows version using the Ctrl key.

Debugging Macros

Debugging is the process of isolating and fixing problems in a macro. Quattro Pro contains a powerful utility for debugging macros. With the debugger, you can

run macros in slow motion (step by step), pausing as long as you want between steps

set breakpoints to pause a macro when it reaches a given cell or satisfies a given condition

run macros at full speed until reaching a breakpoint, then either continue in slow motion or at full speed until the next breakpoint

monitor, or trace, changes to specific cells as a macro runs

Isolating a problem in a long macro isn't simple at regular speed. For this reason, the debugger uses a special mode called Debug. In Debug mode, Quattro Pro runs a macro step by step, pausing for a signal from the mouse or keyboard before going on to the next step. This way, you can monitor each phase of the macro.

To run a macro in Debug mode, choose Tools|Macro|Debugger (or press Shift+F2 or the Pause key). This activates Debug mode and displays the DEBUG indicator on the status line. Then run the macro to debug.

See Also

The Debug Window

Running Macros

Setting Breakpoints

Specifying Trace Cells

The Debug Window

When running a macro in Debug mode, a debug window appears in the bottom half of the screen. The first cell of the macro appears in the middle of the debug window, and the MACRO indicator appears on the status line.

The debug window is divided into two sections:

The top (macro) section contains at most three rows. The middle row displays the macro cell running. The top row displays the previous macro cell, if any, and the third row displays the next macro cell, if any. The macro command about to run is highlighted.

The bottom (trace) section displays the contents of trace cells specified using Trace, and it shows how the macro affects them. See Specifying Trace Cells for more about trace cells.

The debug menu

The menu bar at the top of the debug window shows the following debug commands:

Breakpoints	lets you specify up to four cells or blocks where you want to pause the macro.
Conditional	lets you specify up to four cells that contain logical formulas, such as +`A3=10`; when the formula becomes true, the macro returns to running step-by-step.
Trace	lets you specify up to four cells whose contents you want to monitor while a macro is being debugged.
Edit	lets you change the macro you're debugging without leaving DEBUG mode.
Go	runs the macro until it reaches a breakpoint or finishes.
Terminate	stops the macro and removes the debug window from the screen (this is the same as pressing Ctrl+Break).
Reset	removes breakpoints and trace cells set with the Breakpoints, Conditional, and Trace commands.

Stepping through a macro

To run the first command of the macro, click in the debug window or press Spacebar. Clicking repeatedly steps through each macro command until an error is pinpointed. To run the rest of the macro at full speed, choose Go or press Enter.

See Also

Running Macros

Setting Macro Breakpoints

If you know most of a macro is correct, you can use breakpoints to run parts of a macro at full speed and then enter Debug mode. Standard breakpoints pause the macro when the breakpoint cell is reached. Conditional breakpoints pause the macro when the result of a logical formula in the conditional breakpoint cell is true. Up to four standard breakpoints and four conditional breakpoints can be set at one time.

Standard breakpoints

When debugging a macro containing a standard breakpoint, Quattro Pro runs the macro at full speed until it reaches a breakpoint, then pauses the macro. To resume stepping through the macro, click in the debug window; to continue at full speed until the next breakpoint or the end of the macro, choose Go.

Note: If there are several macro commands in the breakpoint cell, you may need to choose Go more than once to make the macro run at full speed.

To set a standard breakpoint,

1. Choose Breakpoints. Quattro Pro displays four breakpoints.
2. Choose the breakpoint to set.
3. Choose Location and specify the cell or block where you want the macro to stop. If you specify a block, Quattro Pro uses its first cell.
4. Sometimes a problem appears after many repetitions of the same commands in a loop macro. If this is the case, you can set a pass count to indicate how many times to pass through the breakpoint before stopping. Choose Pass Count and specify the number of passes. The default, 1, tells Quattro Pro to stop every time it passes through the breakpoint. Setting it to 2 makes Quattro Pro stop every other pass. Choose OK to set the breakpoint and return to the debug window.
5. Repeat steps 1 through 4 to set additional breakpoints.

Breakpoints in action

The following macro includes a loop that continuously increments a single-cell block called COUNTER.

```
counter      0
is_done      +COUNTER>=100

\m           {LET counter,0}
again        {; increment counter}
             {LET counter,counter+1}~
             {BRANCH again}
```

If you specify the cell containing {BRANCH again} as the first breakpoint and leave the pass count at 1, the macro stops at the {BRANCH} command each time it goes through the loop. When you choose Go, it continues, incrementing the counter cell by one. In the top of the debug window, the selector highlights the {BRANCH} command, indicating that it's the next command.

If you specified a pass count of 5 for the first breakpoint in this example, then every time you choose Go, five loops occur, and the counter increments from 0 to 5 to 10 and so on.

Conditional breakpoints

Conditional breakpoints stop the macro when the result of a logical formula becomes true. In the previous figure, the cell is_done contains a logical formula. (The cell containing the formula is formatted so that the formula, rather than the value calculated, appears.) is_done is false (has a value of 0) until 100 or more loops occur. If this cell is specified as a conditional cell, the macro pauses when the counter reaches 100.

You can set up to four conditional breakpoints per notebook. To set a conditional breakpoint,

1. Choose Conditional to display a menu of conditional breakpoints.
2. Choose the conditional breakpoint to set.
3. Specify the cell containing the condition, and choose OK to set the breakpoint and return to the debug window.
4. Repeat steps 1 through 3 to set additional conditional breakpoints.

Conditional breakpoints can be extremely valuable. Suppose you have a macro loop that does some date calculations and you want it to pause whenever December is reached. If the date is in cell A5, assign the first conditional breakpoint to a cell containing the following formula:

```
@MONTH(A5)=12
```

As long as the formula in the condition cell is true (has a value of 1), the macro runs in step-by-step mode. Once the formula is false (has a value of 0), choose Go to let the macro continue until the next conditional formula evaluates as true. This process repeats until all conditional breakpoints are passed.

Clearing trace cells and breakpoints

To remove all breakpoints (standard and conditional) and trace cells set for the notebook, choose Reset from the debug window. You can remove a standard breakpoint by setting its pass count to 0.

See Also

Debugging Macros

Specifying Trace Cells

Macros often affect the contents of one or more cells. By tracing these cells, you can see what the macro is doing. Quattro Pro lets you specify up to four trace cells, whose addresses and current values show during debugging in the debug window. Quattro Pro updates the debug window as the contents of the trace cells change.

To set a trace cell,

1. Choose Trace from the debug window.
2. Choose the trace cell to set (1st Cell through 4th Cell).
3. Specify the cell to trace and choose OK to set the trace cell and return to the debug window.

To specify additional trace cells, repeat these steps.

Editing macro cells

Once you've pinpointed a problem, you can correct it by

1. Choosing Edit from the debug window.
2. Pointing to or entering the address of the cell to edit using Cell.
3. Editing the contents that display in Content.
4. Choosing OK to save the changes and return to the debug window.

See Also

Debugging Macros

Exiting Debug Mode

When a macro finishes running in Debug mode, the debug window closes, but Debug mode remains in effect, as shown by the DEBUG indicator. To exit Debug mode, choose Tools|Macro|Disable Debugger, press Shift+F2, or press Pause.

To stop a macro before debugging finishes, choose Terminate from the debug window or press Ctrl+Break. Then you can debug another macro or exit Debug mode.

Note: When the debug window displays, part of the notebook is obscured. You can move the debug window to see the data it covers.

See Also
Debugging Macros

Building Menus

Quattro Pro offers a variety of macro commands to add and delete menus and menu commands from the active menu bar. The following topics give details:

What is a Menu?

A menu is a collection of onscreen lists that manipulate an application. Each list contains menu items the user can choose to perform commands (such as saving a file) or display other menus (called submenus or cascading menus). The following table lists terms associated with menus:

Term	Meaning
menu bar	A line at the top of the application window that displays menu commands
menu	An onscreen list of items the user manipulates to communicate with the program.
menu command	An item on a menu the user can choose to perform actions (such as displaying a dialog box, saving a file, or displaying another menu).
menu path	The sequence of menu commands chosen to perform an action or display a dialog box. To enter menu paths in a macro command, enter the sequence of menu commands separated by forward slashes. For example, /File/Save As.
dimmed	Describes a menu command that shows in a lighter color (usually gray instead of black) and isn't selectable. Also known as an unavailable or grayed menu command.
divider line	A horizontal line in a menu that separates groups of menu commands.
hint	A line of text appearing on the status line to provide help when a menu command is highlighted.
link command	An action performed by a menu command when chosen.
shortcut key	A keystroke listed to the right of a menu command's name that the user can press to choose it. Also known as an accelerator.
underlined letter	An underlined character in a menu command's name. The user can press this letter to choose the menu command (if its menu is displayed). Also known as a mnemonic.

Included with Quattro Pro is a file called QUIKSAVE.WB1. This file contains a macro that replaces the standard File|Save command. The notebook page named Complete in QUIKSAVE.WB1 contains the macro and information that creates the new menu.

The remainder topics in this section use QUIKSAVE.WB1 to help explain creating menus and menu commands.

Note: You can use the macro commands {MENUBRANCH} and {MENUCALL} to produce pop-up menus that disappear once an item is selected.

See Also

Building Menus

Designing Menus

Adding Menus to the Menu Bar

Changing the menu bar is a process of removing unneeded menus and menu commands with {DELETEMENU} and {DELETEMENUITEM} and then adding your own with {ADDMENU} and {ADDMENUITEM}. The macro _add_menu in the file QUIKSAVE.WB1 uses {DELETEMENUITEM "/File/Save"} to remove the standard Save command from the File menu. The command `{ADDMENU "/File/Save As",A1..G6}` adds the new Save command to the File menu (`"/File/Save As"` indicates that the menu is to be inserted above Save As on the File menu).

{ADDMENU} uses a menu block to construct the menu added. The menu block contains labels that define the menu; the block Complete:A1..G6 in QUIKSAVE.WB1 is a menu block. The first row of the menu block contains the title of the menu to create. This title is preceded by MENU. In QUIKSAVE.WB1, MENU &Save specifies that the name of this menu is Save. Each row in the menu block (except for the first) defines one menu command using a command definition. Each cell within a command definition specifies one aspect of the command to add; every command definition is a row of six adjacent cells.

The first row of a menu block defines the menu command that displays the menu. Any command definitions that begin in the first column of the menu block display on this menu. In the page named Complete in QUIKSAVE.WB1, cell A1 defines the command that displays the menu as Save. The commands Replace and Backup (defined by the labels in row A2..F2 and A3..F3) are on the Save menu because their definitions start in the same column as MENU &Save.

To make a command in the menu block display a menu, start the command definitions below it one column to its right. For example, in QUIKSAVE.WB1, starting the definitions of Same Directory, Backup Directory, and Options in the second column of the menu block has these results:


makes Backup a command that displays a menu


places Same Directory, Backup Directory, and Options on the Backup menu

Starting the command definition of Backup Directory in the third column would make Same Directory a menu containing the Backup Directory command.

See Also
Building Menus

Replacing the Menu Bar

You can make a menu block the active menu bar using {SETMENUBAR}, which takes one argument (the menu block). The commands on the menu replace any commands on the menu bar; the command defined in the first row of the menu block is ignored. To restore the original menu bar, use the macro command {SETMENUBAR}. If Quattro Pro is in Developer mode you can press Ctrl+Shift+N to restore the original menu bar.

See Also

Building Menus

Using Developer Mode

Using Command Definitions

The following topics describe how to work with the six settings (cells) that make a command definition. You can study QUIKSAVE.WB1 to see examples of these settings. Because many of these cells contain long labels, the sample file QUIKSAVE.WB1 breaks down the menu block setting-by-setting, starting with the notebook page Names. The page Complete shows the actual menu block used by the macro _add_menu.

Naming a Menu Command

Making Menu Commands Act

Menu Hints

Assigning a Shortcut Key

Graying Menu Commands

Inserting Check Marks

See Also
Building Menus

Naming a Menu Command

The first cell of a command definition contains the name of the command. Its underlined letter is preceded by an ampersand (&). (The user can press this key to choose the command.) For example, entering &File makes the menu command name display as **File** in the menu. See the page named NAMES in QUIKSAVE.WB1 for examples.

You can use {SETOBJECTPROPERTY} to change a menu command's name after it's created. See Changing Command Properties for details.

See Also
Building Menus
Using Command Definitions

Making Menu Commands Act

The second cell of a command definition contains a link command similar to those in dialog boxes (see The Object Link Dialog Box) or the name of a macro to run when the menu command is chosen. See the page named LINKS in QUIKSAVE.WB1 for examples.

To run a macro stored in a notebook, enter MACRO MacroName into the cell, where MacroName is the name of the macro to run. For example, in QUIKSAVE.WB1 the menu command Backup Directory uses the label MACRO [QUIKSAVE]floppy_backup to make the menu command run the macro floppy_backup (in QUIKSAVE.WB1) when chosen.

Menu commands can display the dialog boxes in Quattro Pro's normal menus using command equivalents. For example, the following link command makes the menu command behave like Block|Copy:

```
ON Clicked DOMACRO {BlockCopy?}
```

In menu commands, a link command is typed instead of specified with a dialog box (see Working with Link Commands for information on link commands in dialog boxes, which are created by Dialog|Links). The following table lists the events a menu command's link command can detect:

Event	Description
Clicked	The user chose the command from the menu.
Init	The menu containing the menu command is about to display.
Activate	The menu command has just been highlighted.
Deactivate	The menu command has just been unhighlighted.

You can attach multiple link commands to the same menu command; separate them with commas in the cell. If you don't want to assign any actions to a menu command, leave the second cell of its definition blank.

The remainder of this section lists the syntax of link commands that you can enter into a menu block. In each link command, *event* is one of the events listed above, *Object*.*ObjectProperty* is a string identifying the object and property setting to change or read, and *Property* is the name of a property in the menu command. See Changing Command Properties for more information on the syntax of *Object*.*ObjectProperty* and *Property*.

EXECUTE

Format: ON *event* EXECUTE *Object*.*Action*

Performs *Action* on *Object* when *event* occurs. *Object* is a dialog box or dialog control.

DOMACRO

Format: ON *event* DOMACRO MacroText

Runs the macro MacroText when *event* occurs. MacroText is a series of macro commands. The following link command saves the active notebook when the menu command is chosen:

```
ON Clicked DOMACRO {FileSave}
```

RECEIVE

Format: ON *event* RECEIVE *Property* FROM *Object*.*ObjectProperty*

Sets the menu command's *Property* property to *Object*'s *ObjectProperty* when *event* occurs. *ObjectProperty* and *Property* don't have to be the same. The following link command enables the menu command whenever Tools|Macro is enabled:

```
ON Init RECEIVE Enabled FROM /Tools/Macro.Enabled
```

SEND

Format: ON *event* SEND *Property* TO *Object*.*ObjectProperty*

Sends the current setting of the menu command's *Property* property to *Object*'s *ObjectProperty* when *event* occurs. *ObjectProperty* and *Property* don't have to be the same. The following link command stores the menu command's title in the the current cell:

```
ON Clicked SEND Title TO Active_Block.Value
```

SET

Format: ON *event* SET *Setting* TO *Object*.*ObjectProperty*

Sets *Object*'s *ObjectProperty* to *Setting* when *event* occurs. Setting is a value appropriate to the property (for example, Yes or Beveled Out). The following link command sets the active notebook's zoom factor to 150 percent:

```
ON Clicked SET 150 TO Active_Notebook.Zoom_Factor
```

See Also

Building Menus

Using Command Definitions

Menu Hints and Help

The third cell of a command definition contains a brief help message that displays on the status line when the menu command is highlighted. Use these hints to make the menus easier to understand. See the page named HINTS in QUIKSAVE.WB1 for examples.

You can also use the third cell of a command definition to link the command to a Help topic in a Windows Help file that you've created. When the command is highlighted, you can press F1 to display the Help topic.

To link the command to Help, add a backslash after the hint message and type the WinHelp context string for the Help topic followed by an at-sign (@) and the name of the Help file. The name of the Help file cannot include a path; the file must be located in the Quattro Pro directory. For example, the following hint links the Replace command to a Help topic with the context SAVE_REPLACE in the Help file QUIKSAVE.HLP:

```
Overwrite the notebook file on disk\SAVE_REPLACE@QUIKSAVE.HLP
```

For details on how to create Windows Help files, refer to the Borland C++ *Tools and Utilities Guide* or the Microsoft Windows Software Development Kit (SDK).

See Also

Building Menus

Using Command Definitions

Assigning a Shortcut Key

The fourth cell of a command definition contains a shortcut key that the user can press to choose the command. If an item doesn't need a shortcut key, leave the fourth cell blank. See the page named HOTKEY in QUIKSAVE.WB1 for examples.

Separate each component of the keystroke with the plus sign (+). For example, if a shortcut key is comprised of Alt and the F2 key, enter it as Alt+F2. If a shortcut key is used by Quattro Pro for some other operation, that operation occurs first. See the keyboard template included with Quattro Pro to find unassigned function keys.

You can assign shortcut keys to existing commands using {SETOBJECTPROPERTY}; see Changing Command Properties for details.

See Also

Building Menus

Using Command Definitions

Graying Menu Commands

The fifth cell of a command definition specifies the areas of Quattro Pro in which the menu command is available. Enter Yes or No for each area, separated by commas, in the following order: Desktop, Notebook, Graph window, Dialog window, Input line, Graphs Page. The string must be enclosed in quotes. A menu command dims when it isn't available and the user can't select it. To make a command available in all areas, leave this cell of the definition blank. See the page named DEPENDENCIES in QUIKSAVE.WB1 for examples. The new Save menu in QUIKSAVE.WB1 is only available when a notebook window is active.

You can also disable an existing menu command with {SETOBJECTPROPERTY}; see Changing Command Properties for details.

See Also

Building Menus

Using Command Definitions

Inserting Check Marks

The sixth cell of a command definition contains Yes if the menu command should appear checked when it's displayed. If this cell is blank (or contains No), the item displays unchecked. You can check or uncheck a menu command after its creation using {SETOBJECTPROPERTY}; see Changing Command Properties for details.

See Also

Building Menus

Using Command Definitions

Deleting Menus

You can remove a menu from the menu bar using {DELETEMENU}, or hide it using {SETOBJECTPROPERTY}. For example, the following macro removes the Data menu:

```
{DELETEMENU "/Data"}
```

See Changing Command Properties for information on using {SETOBJECTPROPERTY} to hide menu commands.

See Also
Building Menus

Adding and Deleting Menu Commands

{ADDMENUITEM} uses the same information as a command definition but adds a single menu command to the menu bar. Each of the six settings discussed earlier is passed as a separate argument to {ADDMENUITEM}.

You can use {DELETEMENUITEM} to remove a single menu command from the menu bar. {DELETEMENUITEM} only removes menu commands that don't display a menu; use {DELETEMENU} to remove menus. You can hide a menu command using {SETOBJECTPROPERTY}; see Changing Command Properties for details.

See Also
Building Menus

Divider Lines

You can use divider lines to group related commands in a menu. The horizontal line between Define Group and Combine in the Tools menu is an example of a divider line. To insert a divider line in a menu block, insert a row between the definitions of the two menu commands it divides. In this row (and in the same column as the names of the command being divided) enter eight hyphens (-) as a label. You can also add divider lines with {ADDMENUITEM}. For example, to place a divider line before Update Links in the Tools menu, use

```
{ADDMENUITEM "/Tools/Update Links","--------"}
```

See Also
Building Menus

Changing Command Properties

Each command on the menu bar has properties you can manipulate. To identify a menu command in @PROPERTY or a macro command, enter the menu path separated by forward slashes (/). Don't include ellipses (...). For example, the following formula returns whether Edit|Paste Format is dimmed on the menu:

```
@PROPERTY("/Edit/Paste Format.Grayed")
```

Each menu command has the following properties:

Title is set to the name of the menu command; the command's underlined letter is preceded by an ampersand (&). Use this property to rename a command or change its underlined letter. For example, the following command changes the underlined letter of File to i:

```
{SETOBJECTPROPERTY "/File.Title","F&ile"}
```

Checked is set to Yes when a check mark is displayed by the command. Use this property to check and uncheck menu commands. For example, the following command checks Edit|Copy:

```
{SETOBJECTPROPERTY "/Edit/Copy.Checked","Yes"}
```

HotKey is set to the shortcut key that you want to invoke the command with. For example, the following macro assigns Shift+F11 to File|Save:

```
{SETOBJECTPROPERTY "/File/Save.HotKey","Shift+F11"}
```

Help line is set to the hint that appears on the status line. For example, the following command removes the hint from Graph:

```
{SETOBJECTPROPERTY "/File/Save.Help_Line",""}
```

Depend On is set to the areas in which the menu command is available, using the same syntax as dependencies in a menu block (see Graying Menu Commands for details). For example, the following command makes File|Close available everywhere except the desktop:

```
{SETOBJECTPROPERTY "/File/Close.Depend_On", "No,Yes,Yes,Yes,Yes,Yes"}
```

Grayed is set to Yes to dim the menu command and disable it. For example, the following command dims Data:

```
{SETOBJECTPROPERTY "/Data.Grayed","Yes"}
```

Enabled is set to No to disable a menu command without dimming it. For example, the following command enables Data:

```
{SETOBJECTPROPERTY "/Data.Enabled","Yes"}
```

Disabled behaves like Enabled, but it's set to No to enable the menu command and Yes to disable it.

Hidden is set to Yes to hide the menu command. For example, the following command hides

Tools|Advanced Math:

```
{SETOBJECTPROPERTY "/Tools/Advanced Math.Hidden","Yes"}
```

Show is the same as Hidden, but it's set to No to hide the menu command and Yes to display it.

See Also

Building Menus

Menu Design Tips

Building a menu is fairly easy, but designing one can present challenges to the novice UI builder. Here are some design guidelines for creating a menu:

Keep menus short. Users lose track after about eight menu commands. Use submenus to put more commands in a menu.

Make menus on the menu bar shallow. Count how many steps it takes to choose a command; keep it down to two or three.

Group menu commands logically. Keep related commands near each other (for example, don't put Save All in the Windows menu when Save is in the File menu). If menu commands are often chosen in sequence, arrange them in the order you choose them, from the top down. Use divider lines to separate groups visually.

Put the most common menu commands at the top. Putting common menu commands at the top makes them easier to find.

Keep it short. The text of menu commands should be as short as possible without being cryptic.

Try to make the first letters of menu commands unique. Then you can make the first letter of each menu command its underlined letter (see Naming a Menu Command). This is easier for the user to remember.

Use ellipses (...) correctly. If a menu command leads to a dialog box, its text should end with an ellipsis. By Windows convention, an ellipsis means a dialog box follows, both in menu commands and in dialog boxes.

Only dim menu commands when appropriate. A menu command should be dimmed when it's irrelevant to what the user is currently doing. See the description of {SETOBJECTPROPERTY} for information on dimming menu commands.

Don't tamper with File, Edit, and Help too much. Windows users expect a consistent set of items in these menus.

See Also

Preparing to Build an Application

Designing Menus

Productivity Enhancements

This version of Quattro Pro for Windows contains many new features to increase your productivity.

Feature	Summary
SpeedBar enhancements	New SpeedBar buttons and Productivity Tools SpeedBar, and a SpeedBar Control menu
Object Help	Point to an object and press Ctrl+ right click to learn about the object
Interactive Tutors	Tutorials that work with your data
Experts	Provide an alternate way to perform certain spreadsheet tasks
@Function Help	Input line and list help
SpeedMenus	Right-click objects for basic commands and properties
Ctrl+key command shortcuts	Standard Ctrl+key combinations for Cut, Copy, Paste, and other commands
Enhanced key compatibility	Set Tab and Enter keys to accustomed behavior
Drag and Drop enhancements	Move cells or extend blocks with the mouse
Centering text over blocks	Center text across several cells
Automatic block names	Instantly name selected blocks and cells
Colored page tabs	Color-code notebook pages
Custom SpeedFill lists	Create your own label lists to use with the SpeedFill button
Goto and Point mode enhancements	Select blocks with F5; improved page Goto; point out blocks in other notebooks
Spell Checker	Check spelling in a notebook, block, or graph (Workgroup edition only)
Input-line parenthesis matching	Check parentheses, braces, and brackets while entering formulas
Model Copy enhancements	Use Special options with Block|Copy| Model Copy
Startup and exit macros	Run a macro when starting or exiting Quattro Pro or notebooks
Error auditing	ERR and NA source tracking
Compiled formulas	Compile formulas for quicker recalculation

SpeedBar Enhancements

These are the main SpeedBar enhancements in Quattro Pro for Windows version 5.0:

New Speedbar buttons

Productivity Tools SpeedBar

SpeedBar Control menu

New SpeedBar Buttons

The notebook and graph SpeedBars have new buttons:

The Block Center button centers text across the active block. Enter text in the leftmost cell of a selected row, then click the Block Center button. The text moves to the center of the active block within that row. For details, see Aligning Cell Entries.

The Experts button displays a selection of online Experts to help you build graphs and perform other tasks.

The Select-All button in the graph SpeedBar selects all objects in the active graph window. The equivalent menu command is Edit|Select All.

Secondary SpeedBars have a new Close SpeedBar button, the red X at their right edge. You can use it with the SpeedBar Control menu to open and close secondary SpeedBars (see Opening and Closing SpeedBars).

Productivity Tools SpeedBar

When you first start Quattro Pro, the new Productivity Tools SpeedBar appears below the notebook SpeedBar. It contains many buttons that are shortcuts for common commands.

To see what each button does, point to it and read the hint line at the bottom of the window. For more information, point, then press Ctrl while you click the right mouse button. Object Help appears.

To close the Productivity Tools SpeedBar, click the red X at the right end of the SpeedBar. You can use the SpeedBar Control menu to display it again (see Opening and Closing SpeedBars).

Opening and Closing SpeedBars

A standard SpeedBar displays at the top of each Quattro Pro window. You can remove that SpeedBar with the Display property in the application Object Inspector (see Hiding Parts of the Window). You can add and remove other (secondary) SpeedBars with the SpeedBar Control menu.

Secondary SpeedBars have a Close SpeedBar button, the red X at their right edge. You can click it to close the SpeedBar, or use the SpeedBar Control menu.

SpeedBar Control Menu

The SpeedBar Control menu lets you add and remove secondary SpeedBars.

To display the SpeedBar Control menu, right-click the SpeedBar any place outside a button or field. The blank space toward the right end of each SpeedBar is convenient.

The SpeedBar Control menu for standard SpeedBars offers one of the options in the following list, Append. All secondary SpeedBars offer the first four options on their SpeedBar Control menus; custom SpeedBars also include the last option:

Append	adds a SpeedBar below the active bar.
Insert	adds a SpeedBar above the active bar.
Replace	closes the active SpeedBar and replaces it with another.
Remove	closes the active SpeedBar. It's often easier to close SpeedBars by clicking the red X at the right end of the SpeedBar.
Customize	appears only for custom SpeedBars. When selected, this option displays the Designer SpeedBar and puts the active SpeedBar into Edit mode (for details, see SpeedBar Designer.

When you choose Append, Insert, or Replace, a list of available SpeedBars appears. Standard SpeedBars are listed above the line; any custom SpeedBars are listed below the line. Choose from the list or click <Browse> to search another directory. You can display as many SpeedBars as you like, either standard or custom.

To save the current SpeedBar arrangement as the default, use the SpeedBars property in the application Object Inspector.

See Also

Selecting Secondary SpeedBars

Object Help

Quattro Pro identifies each SpeedBar button as you point to it. Its name appears at the bottom of the screen.

For more information, point to an object and hold down Ctrl as you right-click the mouse. An Object Help window appears.

Usually, the Object Help window has a Help button. Click it to display a related help topic.

See Also

How to Use Quattro Pro Help

Interactive Tutors

Learning by doing is a most effective technique. Interactive Tutors teach procedures with your own "live" data so you can actually use a feature while learning about it.

To view a catalog of Interactive Tutors topics, choose Help|Interactive Tutors or click the Interactive Tutors button in the Productivity Tools SpeedBar. The Interactive Tutor Catalog appears.

You can choose a topic from the list or click the Index button to search for a keyword in a topic.

When you choose a topic, its first window appears.The buttons at the bottom control your actions. Click the X to cancel the current Interactive Tutor lesson; if you want, you can choose another. The arrow buttons at the right move forward and backward through the Interactive Tutor windows. The Next button is the rightmost arrow at the edge of the window. Click it to advance to the next window.

When you complete a Tutor, you are returned to the Interactive Tutor Catalog, and a check mark appears beside that Tutor. These check marks are saved when you exit Quattro Pro. To erase the check marks, click the Options button at the bottom of the Catalog and check Clear Current Checkmarks. The other two options determine whether check marks are saved when you exit.

Interactive Tutors use Quattro Pro's standard interface, so you learn each feature as you use it. For a more basic approach to several procedures, try using the Experts.

See Also
Object Help
How to Use Quattro Pro Help

Experts

Quattro Pro offers Experts to help you use some features at a basic level with virtually no learning required.

To activate an Expert, click the Experts button or choose Help|Experts to display a list of available Experts; choose one.

A window appears with a description of the first step. Follow the instructions and click the Next Step button when you're ready. You can request help and more information or cancel the session at any time. Just click one of the buttons at the bottom of the Expert window.

See Also

Experts List

Object Help

Interactive Tutors

How to Use Quattro Pro Help

@Function Help

You can display context-sensitive help for Quattro Pro @functions from the input line. Type any @function and press F1 to display a related help topic.

You can also display help within the @function list. To display the list, press Alt+F3. It is organized by @function category. Choose any category to display a list of subcategories, then a list of @functions within that subcategory. At any time, press F1 to display a related help topic.

See Also

How to Use Quattro Pro Help

Entering @Functions

Ctrl+Key Command Shortcuts

You can now use a number of standard Ctrl+key combinations as command shortcuts. See Special Keys for details.

Drag and Drop Enhancements

Now you can use Drag and Drop to move or copy single-cell or multicell blocks. For instructions, see Dragging and Dropping.

Centering Text Over Blocks

You can use the Block Center button or Alignment property in the active block Object Inspector to center text across several cells, a table for example. See Block Centering for instructions.

Automatic Block Names

Block|Names|Labels names single cells according to adjacent labels. Now you can automatically name blocks with more than one cell. For instructions, see Naming Blocks and Cells from Labels.

GoTo and Point Mode Enhancements

You're probably familiar with the "GoTo" key; you can press F5 and instantly move to the specified location. Now, when the GoTo dialog box appears, you can enter a location, choose a block, or choose the first cell of any page.

Now you can also extend blocks with GoTo and point to open notebooks using Alt+W to display the Window menu (see Pointing Out Links)

Spell Checker

The Spell Checker checks the spelling of text in the selected object, whether notebook block or graph. This feature is available in the Workgroup edition of Quattro Pro. For details, see Using the Spell Checker.

Input-Line Parenthesis Matching

A new input-line display feature helps you enter formulas, links, and macros. If you type one of these characters: ([{ , it is colored red in the input line until you type the matching character:)] }. Then, they both turn green. If you forget to type a matching parenthesis in a complex formula with several sets of parentheses or brackets, for example, it's easy to pinpoint the error. Press F2 to display the selected formula in Edit mode, then move the insertion point through the formula. When you reach an unmatched parenthesis or bracket, the character turns red. You can check the formula and insert the matching character where it belongs. You know when you're right. The new character and its match both turn green.

See Also

Entering Formulas

Entering @Functions

Model Copy Enhancements

The Block|Copy|Model Copy option lets you copy a set of formulas so absolute references adjust to the new location of the referenced cell. Several new options help you use Model Copy more precisely. For details, see Model Copy Option Settings.

Startup and Exit Macros

Sometimes it's helpful to run a macro each time you open or close a notebook or start Quattro Pro. For details, see Using Startup and Exit Macros.

Error Auditing

Even relatively small notebooks can have complicated formula dependencies. An NA or ERR in one cell can ripple through the entire notebook, making the original source difficult to trace. Quattro Pro has two new features to simplify that chore, described in these topics:

Tracing Errors with GoTo

Displaying Error Sources in Cells

Compiled Formulas

If you work with large notebooks containing many numerical formulas, using compiled formulas can often boost recalculation speed. For details, see Compiling Formulas.

New Graph Types

Quattro Pro chooses a graph type for each graph you create, based on the number of columns, rows, and data points. Choose Graph|Type to assign a different type. Quattro Pro has these new graph types:

2-D: 100% stacked bar, stacked bar comparison, 100% stacked bar comparison, doughnut, radar.

3-D: 3-D 100% stacked bar, 3-D doughnut.

Rotated: rotated 2-D stacked bar, rotated 2-D 100% stacked bar, rotated 2-D comparison, rotated 2-D 100% comparison, rotated 3-D stacked bar, rotated 3-D 100% stacked bar.

For details, explore the topics listed in Graph Type Property.

Analytical Graphing Overview describes another new graphics feature.

Analytical Graphing Overview

Quattro Pro can calculate new data points and graph them without changing data in the notebook. This feature is called analytical graphing.

For example, suppose your spreadsheet contains daily sales totals. Analytical graphing makes it easy to graph average monthly totals and calculate similar statistics with the Aggregation option. If daily sales vary widely, you can use the Moving Average option to smooth the data points. You can also show general trends by fitting the data to a line (with the Linear Fit option) or an exponential curve with the Exponential Fit option. You can also build tables of analysis results.

See Also

Analytical Graphing

Analyze Property

SpeedBar Designer

Quattro Pro supplies many standard SpeedBars for push-button access to commands and procedures. They change to suit your current activity, and you can add or remove SpeedBars at any time.

As you work with Quattro Pro, you'll probably use some commands and features more often than others. The SpeedBar Designer offers a simple way to create custom SpeedBars for features you use the most, even special macros. You can choose standard buttons from palettes or build your own. Each button has its own Object Help for easy identification. Standard buttons even link to detailed help topics for more information.

For details, see Building Custom SpeedBars.

If you need to create dialog boxes or more complex SpeedBars, consider using the UI Builder, opened with Tools|UI Builder. It's a useful tool for application developers who need its full power and flexibility.

See Also
SpeedBar Control Menu

Databases and Data Exchange

This version of Quattro Pro has several enhancements that help you organize and exchange data:

Database forms	create data entry and query forms without programming.
Password protection enhancements	apply passwords to notebooks and cells; set security limits.
Parse Expert	an option of Tools | Import, imports and parses plain text in one operation.
Text file translation enhancements	load and save text files, including tab-delimited with extension .TXT.

Related new features are:

Data Modeling Desktop, which creates database crosstab structures with the mouse.

Workgroup Desktop, which helps distribute notebooks via MCI, MHS, and other messaging protocols (Workgroup edition only). It has its own Help system.

Database Desktop with optional SQL support lets Quattro Pro read data from external databases using SQL (Structured Query Language).

Other News

It's easy to load Quattro Pro for DOS 5.0 files. Just use *.WQ2 in the File Types edit field for File|Open or File|Retrieve.

You can use Quattro Pro as an OLE server and a DDE server for graphs and RTF text. For more information see OLE and DDE Overview.

OLE and DDE Overview

Quattro Pro acts as a DDE (Dynamic Data Exchange) client and server for formatted data and charts. You can copy these objects from Quattro Pro to the Clipboard, then paste them into client applications as DDE objects. Formatted data can be pasted in many formats, including RTF (Rich Text Format) format to preserve existing type styles, sizes, and column widths.

Quattro Pro also acts as an OLE (Object Linking and Embedding) client and server for notebook blocks and graphs, either linked or embedded.

Follow the instructions provided with client applications for creating DDE and OLE connections from Quattro Pro to other applications. Usually, you copy an object to the Clipboard in Quattro Pro, then activate the client application and use Edit|Paste Special or Paste Link. Or, use Insert|Object or Edit|Insert Object in the client application to launch Quattro Pro and create a new object.

If you embed an object from Quattro Pro in a client application, then double-click to edit the object in Quattro Pro, you'll see these command changes in the File menu: Save changes to Update; if you edit the notebook, Update transfers the changes to the client file. Save As changes to Save Copy As; you can only save a copy--the original "belongs" to the client. If you try to close a notebook that was edited after the last update, you'll be prompted to update the client first.

See Also
Creating DDE Links

Model Analysis Tools

Models are sets of data and related formulas used for research and prediction. They represent situations in business or other fields and show how results vary when certain data changes. Quattro Pro offers a number of tools for making projections or optimizing results based on models you define.These tools are new in this version:

Scenario Manager	changes values in a model, saves conditions and results for different scenarios.
Consolidator	combines data blocks and saves consolidation setups.

The Optimizer, introduced in Quattro Pro for Windows version 1.0, has a new constraint type (Integer), new options, and new Answer Report data.

See Also
Advanced Analysis Tools

Advanced Analysis Tools

A new command, Tools|Analysis Tools offers 19 options for analyzing statistical, engineering, and financial data. For details, see Advanced Analysis Tools Tasks.

Array Features

Quattro Pro has a special @function and other features to help you use data arrays, blocks of data you work with as a group.

Working with arrays can save time and memory when you enter formulas repetitively, but be aware that using arrays can lengthen recalculation times. If you plan to share a notebook containing arrays with other users, alert them to the arrays so the notebook is easier to understand.

The following topics explain the use of arrays in formulas and with a new @function, @ARRAY:

Array Formulas

Block Arrays

Array Constants

Arrays as @Function and Macro Arguments

Importing and Exporting Files

These topics describe how to use and create files in other formats:

Translating Files

Quattro Pro can translate data many spreadsheet and database program formats. The table below shows the types of files Quattro Pro can translate, and the file extensions to use for each type. Quattro Pro looks at the extension to determine the type of translation to perform.

To save a file for use with one of these programs, include the appropriate extension when you save it. For example, to save the notebook MYFILE for use with Symphony version 1.2, use File|Save As and specify the file name MYFILE.WRK.

To load a file created by one of the following programs, include the file's extension when you choose File|Open or Retrieve.

Extension	Program
.WQ1	Quattro Pro for DOS
.WQ2	Quattro Pro for DOS, version 5.0
.DB	Paradox
.DB2	dBASE II
.DBF	dBASE III, III+, and IV
.XLS	Excel
.WKS	1-2-3, version 1A
.WK1	1-2-3, version 2.x
.WK3	1-2-3, version 3.x, 1-2-3 for Windows
.WKE	1-2-3, educational version
.WRK	Symphony, version 1.2
.WR1	Symphony, version 2.0
.RXD	Reflex, version 1
.R2D	Reflex, version 2
.DIF	VisiCalc
.SLK	Multiplan
.TXT	tab-delimited text

Caution: Only the contents of the current page (or if it's empty, the first nonempty page) of a notebook are saved when translating to all formats except 1-2-3 3.x (.WK3).

To create unformatted, tab-delimited text files, use File|Save, File|Save All, or any other command that creates a file in Quattro Pro and change the extenstion to .TXT in the File Name edit field.

If you plan to open the same locally stored data file in more than one application running under Windows at the same time, make sure SHARE.EXE is running. For example, run SHARE if you plan to open a file stored on your C drive in both Excel and Quattro Pro. Without SHARE running, the first application to open a file doesn't properly prevent other applications from changing the file, which makes it easy to overwrite changes unintentionally. For complete information on SHARE, see your DOS manual.

Tip: Add SHARE to your AUTOEXEC.BAT file so it loads whenever you start your computer. Just add SHARE on a line by itself. (SHARE.EXE must be in the root directory or on your path.)

If you omit the file-name extension while loading a file, Quattro Pro looks for the file name you specify

with the following extensions in this order: the File Extension setting in the application Startup property, .WB1, .WQ1, .WK1, .WK3, .WKS, and .XLS. This means you can load a file from some spreadsheet products without specifying the extension as long as a file with the same name doesn't exist with a higher-ranking spreadsheet extension.

Quattro Pro supports the DIF file format developed by Software Arts Products Corporation, which is used by VisiCalc. Although other programs may import/export DIF files, they aren't necessarily the same file format. Consequently, they may not be compatible with Quattro Pro.

Note: If you use unique Quattro Pro features like linking or presentation-quality graphics, you may see a warning when you try to save the file to another file format. The warning states what feature will be lost when the file is translated. If you don't want to lose the feature, save the file as a .WB1 file.

See Also

Translating Allways Files
Translating Impress Files
Translating 123 Release 2 or 3 Files
Translating Multiplan Files
Translating Database Files
Setting the File Extension
Importing Text Files
Data|Query
Importing and Exporting Files

Translating Allways Files

Spreadsheets created with 1-2-3 version 2.x and designed using Allways can be loaded directly into Quattro Pro. When you load a .WK1 file that has a corresponding .ALL file, the Allways formatting is automatically added to the file.

Quattro Pro imports only .ALL files (not .AFS, .ALS, or .ENC files). To import multiple saved formats or font sets, create a separate .WK1 file for each saved format with a corresponding .ALL file. Quattro Pro imports the following Allways options:


These Format options: font selection, line style (the Layout|Options|Line-weight setting is used to gauge overall line thickness), shading, boldface, underline, and font colors. Quattro Pro converts up to 255 different combinations of font, boldface, underline, and italic. Any additional combinations convert to the font specified as Normal style.


Layout options: margins, titles, borders, line weight, paper type, and grid on printing.


The Print range option.


Worksheet options: column width and row heights.

Quattro Pro doesn't import the following:


Inserted graphs (because Allways only points to a graph stored in a separate .PIC file in its "inserted" graphs).


These Worksheet options: page breaks, column page breaks, and display zoom.


Label alignment with spillover to the left (usually found in centered labels).


The display (screen) colors.


These Layout options: page size and borders on the bottom.


These Print options: printer type, orientation, print settings, and port bin print options.

To save an Allways formatted file you previously imported, use File|Save to automatically save it with an Allways file, or use File|Save As to choose Impress or None (no format file).

See Also

Translating 123 Release 2 or 3 Files

For DOS Spreadsheet Users

Translating Impress Files

Translating Files

Translating Impress Files

Quattro Pro can now import 1-2-3 files created with either the WYSIWYG or Impress add-in.

When you retrieve a .WK1, .WKS, or .WK3 file that has a corresponding .FMT or .FM3 file, Quattro Pro asks if you want to read the .FMT or .FM3 file and apply the WYSIWYG/Impress formatting to it.

When retrieving a .WK1 or .WKS file, if there is a corresponding .ALL file in the current directory, you can choose to open it instead. If there is a corresponding .FMT file, Quattro Pro chooses it automatically, and asks you to confirm the choice. The .ENC and .CNF files created by WYSIWYG/Impress are not imported.

Quattro Pro retrieves and applies some graph formats as well as text formats. Not all graph formats are retained, however. Quattro Pro imports the following WYSIWYG/Impress formats:


These Format commands: assigned fonts, lines (except shadow), shading, boldface, underline (except double or wide), italics.


Quattro Pro converts up to 255 different combinations of font, boldface, underline, and italic. Any additional combinations convert to the font used in Normal style in Quattro Pro.


Inserted graphs.


Blank graphs (usually containing annotations).


Named styles, but the descriptions are not preserved.


These Print settings: range, configuration/orientation, layout/compression (which converts to Print-To-Fit), layout margins, layout titles, and grid on printing.


Label Alignment (except left-side spillover).


The following Worksheet settings: row height, page breaks.


Colors (global negative, lines, text color).


Grid display.

Quattro Pro does not import the following formats:


Column page breaks.


Label alignment with left-side spillover.


Formatting embedded in text.

 These Display options: colors, mode, font directory, rows, and options.

 These Format options: line shadow and colors (except global negative, line, double or wide underline, and text color).

 These Print Layout settings: page size and borders on bottom.

 These Print settings: frame and settings.

 The Print Configuration settings (printer type, port bin, and so on).

See Also

Translating 123 Release 2 or 3 Files

DOS Spreadsheet Users

Translating Allways Files

Translating 1-2-3 Release 2 or 3 Files

When loading a 1-2-3 version 2.x file, linking is converted to Quattro Pro's syntax, as shown in the following example:

`+<<BUDGET.WK1>>B4` changes to `+[BUDGET.WK1]B4`

To import a 1-2-3 version 3.x file into Quattro Pro, just specify the .WK3 extension when you open or retrieve it.

If the only version 3-specific features in the 1-2-3 file are a 3-D worksheet or LMBCS characters, Quattro Pro opens it and converts it. If, however, there are other version 3-specific features in the 1-2-3 file, any of the following might happen:


Any @function not in Quattro Pro converts to the value of the original cell (string or numeric).


Any .WK3 graph feature not supported in Quattro Pro is lost.


Due to the way Quattro Pro stores numbers, any extremely large number (larger than 10 raised to the power of 308) converts to ERR, and any number smaller than 10 raised to the power of negative 308 converts to 0.


Numeric format of Automatic, Negative Color, Label and Parenthesis convert to the Quattro Pro format General.


Numeric formatting of blank cells will not be identical to the original once converted.

To export a Quattro Pro file to 1-2-3 version 3 format, just save it with the .WK3 file extension.

See Also

DOS Spreadsheet Users

Translating Allways Files

Translating Impress Files

Translating Multiplan Files

Since Multiplan doesn't save its files in .SLK format by default, you must do the following in Multiplan before translating the files in Quattro Pro:

1. Choose Transfer|Options|Symbolic.
2. Choose Transfer|Save, and save the file with the extension .SLK.

Alternatively, you could do the following to save the file in 1-2-3 format in preparation for translating to Quattro Pro:

1. Choose Transfer|Options|Other.
2. Choose Transfer|Save, and save it with the extension .WK1 or .WKS depending on your version of Multiplan.

See Also
Translating Files

Translating Database Files

Quattro Pro can translate files to and from dBASE, Paradox, and Reflex formats. You specify the file format you want by selecting its extension in the Save As dialog box.

dBASE II, dBASE III, and dBASE IV use the same file-name extension (.DBF) but different file formats. To save Quattro Pro files in dBASE II format, use the .DB2 extension, then change the extension to .DBF (in DOS) before opening the file in dBASE II. In like manner, use the .DB4 extension to save Quattro Pro files in dBASE IV format, then change the extension to .DBF before opening the file in dBASE IV. To load dBASE II files into Quattro Pro, however, use the .DB2 or .DBF extension.

When you load any database file into Quattro Pro, it converts database field names to spreadsheet labels that serve as column headings and assigns block names to the cells beneath each name.

When you save a Quattro Pro notebook in Paradox, Reflex, or dBASE format, a dialog box appears with options for ordering the spreadsheet data into a database file structure:


The **Fields** box lists database field names based on the data in the first row of the first spreadsheet page. If the first row contains data that is invalid as a database field name--for example, containing numbers, spaces, symbol characters, or duplicate labels--Quattro Pro uses the spreadsheet column letters to represent the field names. To change a field name, choose the default name in the Fields box and type a new name in the Name edit field.
To mark a field to be deleted, highlight it on the list and press Del. An asterisk appears next to the field name. To restore it, press Del again.


Use the **Type** settings to change the type of data contained in the column. The default setting is based on the spreadsheet data. If the Type is Numeric, you can specify a number of decimal points for the field.


Use the **Width** edit field to specify the number of characters in the structure for that field name.


The **Write** button proceeds with the translation to the database file format corresponding to the file extension you typed in the Save File dialog box.

See Also

Translating Files

Combining Files

Extracting Part of a Notebook

Importing Text

Tools|Import copies a text file into the active page of a notebook. Quattro Pro can import three types of text files with these dialog box options:

ASCII Text File imports a plain, unformatted text file. Quattro Pro converts the data into a single column of labels. Each line in the file becomes a label in a single cell.

Comma & " Delimited File imports a file that uses commas and quotes to separate text in rows. The delimiters are used to set up columns in the spreadsheet page.

Only Commas imports a file delimited only with commas.

Parse Expert imports a file and breaks it into columns, such as database fields, at the same time. Depending on how the text file is organized, if you use Parse Expert you might need to do some editing when the import is complete.

Note: Don't use Tools|Import to load Paradox, dBASE, Reflex, 1-2-3, Symphony, VisiCalc, or Multiplan files. Instead, use File|Open or Retrieve and specify the appropriate file extension (see Translating Files for details).

To import or export a tab-delimited text file, use File|Save, File|Save As, or another file-creation command and give the extension as .TXT in the File Name edit field.

To import a text file,

1. Select the upper left corner of the block where you want to place the imported data.
2. Choose Tools|Import.
3. Choose the appropriate option: ASCII Text File, Comma & " Delimited File, Only Commas, or Parse Expert.
4. Choose the file name you want using the techniques described in File Handling Options.

The data from the text file is copied into the active page starting with the active cell.

Plain Text Files

Text files contain straight, unformatted text. Before you import a text file, remove any special formatting characters such as bold, underlining, or centering.

Caution: Make sure no lines in the file are longer than 1022 characters, the maximum size of a cell. Characters beyond the 1022-character limit won't be imported.

Quattro Pro enters each line from the text file as a label in the first column of the indicated block (using columns to the right for spillover display). You can then break the long labels into a more usable format with Data|Parse. Blank lines in the imported file become empty cells.

Note: Many word-processing programs produce files that contain special characters you may not be able to see. These characters may produce unwanted results. If your word processor has an option for creating plain text files, use it to create files you intend to import.

See Also

Files with Commas and Quotes

Breaking Text into Data Fields (Parsing)

Using Format Lines

Files with Commas and Quotes

A comma-and-quote delimited file has the following characteristics:

- Data is entered in lines, much like the rows of a spreadsheet page.
- Types of data are separated (or delimited) on each line with commas.
- Text strings are surrounded by quotation marks.

When Quattro Pro imports a comma-and-quote delimited file, it stores each delimited group of data in a separate cell. Data groups that are strictly numbers become value entries, data groups surrounded by quotes are stored as labels, and other entries are ignored.

Importing a file with the Commas Only option is similar to importing a comma-and-quote delimited file. Instead of requiring quotes around labels, Quattro Pro determines whether to enter the data as a label or a value depending on its first character.

See Also

Tools|Import

Breaking Text into Data Fields (Parsing)

Importing and Exporting Files

Breaking Text into Data Fields (Parsing)

After importing a text file, you may need to parse the data into individual cells. Imported text files often require parsing because each line of text translates into a single label. Plain text files without commas, quotes, or other delimiters can usually be parsed with Parse Expert; try it first before you try this procedure (for details, see Importing Text).

The basic procedure for parsing a file is as follows:

1. Select the first label to be parsed (the top left cell in the block of labels).
2. Choose Data|Parse.
3. Use the Create and Edit buttons to set up one or more format lines to indicate how you want to divide the labels.
4. Use the Input edit field to specify the column of labels you want to parse, including all format lines.
5. Use Output to specify the destination for the parsed labels and choose OK.
6. If necessary, resize columns to display the data fully.

See Importing Text for information on the Parse Expert; it imports and parses text at the same time.

See Also

Using Format Lines

Using Input and Output Blocks

Importing and Exporting Files

Using Format Lines

You should place a format line in a new inserted row above the data to be parsed. It uses the following symbols to indicate how text below it will be translated into separate fields:

Symbol	Purpose
|	Begins a format line (shown in the input line only).
V	Begins a value.
L	Begins a label.
T	Begins a time value translated into a serial number.
D	Begins a date value translated into a serial number.
>	Continues an entry.
*	Indicates blank spaces that can be filled in by longer entries underneath the first.
S	Indicates to skip the character in this position, thereby deleting it from the parsed data (this symbol can be entered only by editing the format line).

Quattro Pro bases the format line on the first row of data. You can edit the line to parse the labels differently, and you can insert additional format lines to parse specific areas differently.

Creating Format Lines

To create a format line, select the first cell containing a label to be parsed, choose Data|Parse, and choose Create. A format line appears, moving the existing rows down. Click Format Line Example for details.

Tip: Choose the block Font property to put the format line and the text to be formatted into a monospace font, such as Courier. This helps you make sure that the text in each column starts at the same position.

Editing Format Lines

To edit a format line, select the first cell containing the format line and choose Edit from the Data|Parse dialog box. Quattro Pro opens a dialog box containing a box for editing format lines.

Edit the format line in the edit field by clicking where you want to make a change and using standard editing techniques. The first line of the data is displayed so you can revise the format line to match the data properly. Choose OK to finish editing.

Using Multiple Format Lines

In some cases, you must use more than one format line. To create an additional format line,

1. Select the top left cell of the data to be formatted with the new format line, and choose Create from the Data|Parse dialog box.
2. A format line appears above the data, interpreting the numbers as values and the row headings as labels.

You can insert as many format lines as necessary. Each format line affects the text below it down to the next format line or to the end of the input block.

See Also

Breaking Text into Data Fields (Parsing)

Using Input and Output Blocks

Using Input and Output Blocks

Before you can parse data, you must indicate which block of data to parse and where to copy the parsed data. The input block is the column that contains the text you want to parse and the format line(s) used to parse it. (Remember that labels reside in a single column, even though they may spill over several.)

To specify the input block, click in the Input edit field of the Data|Parse dialog box, and select or type the block to parse. Make sure you include all format lines.

The output block is the destination block for the parsed data. To specify the output block, click in the Output edit field and select or type the top left cell or the entire block where you want the data to begin.

Quattro Pro uses whatever space the parsed data needs, overwriting cells if necessary. To be safe, first specify an output block in another part of the page or notebook where there is no data to be overwritten. Then, after you obtain the results you want, replace the original data with Block|Move or with the Clipboard commands.

Performing the parse

After importing the data, creating and editing format lines, and specifying input and output blocks, choose OK from the Data|Parse dialog box to perform the parse. Resize columns as necessary to reveal their full contents.

See Also

Breaking Text into Data Fields (Parsing)

Using Format Lines

Block|Copy Command

Edit|Cut Command

Edit|Paste Command

Inserting a File into a Notebook

You can copy an entire file into a notebook, pushing existing pages back to make room. Only pages containing data are inserted. To insert a file,

1. Make sure the file you want to insert is closed.
2. Select any cell on the page before which you want to insert the file.
3. Choose Block|Insert|File.
4. Choose the file name you want, using the techniques described in File Handling Options. If you want to insert a file translated from another program, include its file-name extension (see Translating Files for details).
5. Choose OK.
6. If the file has a password, a dialog box appears with space for entering it. Type the password and choose OK.

Every page in the inserted file that contained data is placed on a new page in the active notebook. If only one page is inserted, the page is given the name of the inserted file. If multiple pages are inserted, the new pages are named in the default letter sequence.

If the file is inserted within the boundaries of a named block or a block referenced by a formula, Quattro Pro expands block references to include the new pages.

Note: If inserting a file would expand a named block or cell reference beyond the limit of a notebook (beyond page IV), the reference becomes ERR.

See Also

Combining Files

Extracting Part of a Notebook

Importing and Exporting Files

Combining Files

Tools|Combine lets you copy all or part of a notebook into any area of the active notebook. Unlike File|Retrieve, it doesn't erase the active notebook; it affects only the portion of the pages covered by the inserted block.

You can also perform arithmetic operations with Tools|Combine, to add, subtract, multiply, or divide cells from one notebook to another.

Note: If you make changes to a file to be combined, save it first before combining. Tools|Combine reads data from the stored notebook file, whether it's open or not.

If you prefer, you can copy cells directly from one notebook to another with Block|Copy or with the Clipboard commands. You can also perform arithmetic operations between cells with notebook links (see Linking Notebooks for details).

To combine files,

1. If you intend to combine only part of a file with the active one, make sure you know the exact block name or coordinates from the source file.
2. Select the receiving notebook. Then select the cell where you want the upper left of the source file to begin being copied. Make sure you have enough space so data won't be overwritten unexpectedly.
3. Choose Tools|Combine, and choose the appropriate option in the dialog box:

Copy inserts the exact contents of the incoming file.

Add adds the incoming values to the existing values.

Subtract subtracts the incoming values from the existing values.

Multiply multiplies the incoming values by the existing values.

Divide divides the existing values by the incoming values.

4. Also in the Combine dialog box, choose Entire File to copy the entire file (including all pages) into the existing one, or choose Block(s) to copy a specified block (or blocks) from the file. If you choose Block(s), enter a block name or coordinates in the edit field. You can also specify a noncontiguous block by separating the subblocks with commas. (Because the block names are from another file, you can't display a list of the names.)
5. Choose the file name you want. See File Handling Options for details.

The contents of the specified file or block appear in the active notebook. If you copied the data, Quattro Pro also copies any properties such as alignment and numeric format. Column widths in the receiving notebook don't automatically adjust to correspond to the copied data; you can do this explicitly with the Fit button.

Tools|Combine copies the contents of named blocks, but to avoid confusion with names in the active notebook, it converts references to named blocks into block coordinates.

Caution: Tools|Combine overwrites any cells in the destination block, even if they are protected.

See Also

Copying Combined File Data

The Copy option of Tools|Combine copies data directly from a file into the active notebook. The data is inserted starting at the position of the selector. If you specify the entire file, Quattro Pro copies only the block that contains data.

Tip: If you create a notebook consisting of headings for particular pages in your notebooks, you could copy the headings into other notebooks and avoid having to retype the headings. To avoid overwriting data in the target notebooks, you would reserve the first few rows in all notebooks for headings, and enter the headings in those reserved rows in your headings notebook.

To insert the headings into the new notebook:

1. Select cell A1 in the notebook where you need the headings.
2. Choose Tools|Combine.
3. Choose Copy. Choose Entire File.
4. Enter the name of the file containing the headings.

The headings are added to the top of the notebook.

With the Copy option of Tools|Combine, references in formulas adjust to reflect their new positions, even if they are absolute. If a formula refers to cells outside of the block being copied, the result may be inaccurate. To copy formulas' resulting values instead of the formulas themselves, use Block|Values on the file you're combining to copy them to another page first (see Block|Values for details). Then choose the Blocks option of Tools|Combine to copy only the page with the values.

See Also

Combining Files

Operating on Combined File Data

Importing and Exporting Files

Operating on Combined File Data

The Add option of Tools|Combine combines values in the inserted block with existing values in the notebook and calculates the sum.

The inserted block can contain formulas as well as values. However, the receiving notebook should contain only values--if it contains formulas, convert them to values (with Block|Values) before using the Add option. Addition won't occur in cells with formulas or labels.

Note: Another way to add values between files is to create notebook links (see Linking Notebooks for details).

The Add option of Tools|Combine is useful for combining files for a cumulative total; for example, to compile year-to-date figures from monthly notebooks. You could load one file, select cell A1, and combine the other files with the Add option of Tools|Combine. A cumulative notebook showing year-to-date expense totals would be created.

Before you use the Add option, select the cell where you want the added data to start. Data will be added to values below or to the right of the selector.

If you add an entire file, Quattro Pro adds all cells that contain values. If you specify a block to add, Quattro Pro adds the values from that block only.

When adding values, make sure the source and destination areas are set up similarly; for example, put headings in the same position. The incoming data assumes the properties of the active notebook, except that it retains its numeric format.

Caution: Accurate selector placement is critical when adding combined values. If the selector is one cell off from proper alignment when you combine the files, the combined data may be useless. For this reason, save the active notebook before combining data. Then, if you don't get the results you want, you can retrieve the original file and try again.

When you add values with the Add option of Tools|Combine, Quattro Pro follows these rules:

In the active notebook, cells that contain labels, formulas, ERR, or NA are not altered by the incoming data.

Value entries in the destination block of the receiving notebook are replaced by the sum of the original and incoming values.

When adding incoming values to those existing in the notebook, all incoming formulas convert to their end values, and labels and blank cells are ignored. Formulas resulting in ERR, NA, or strings are ignored also.

Subtracting, multiplying, or dividing data

The Subtract option of Tools|Combine subtracts incoming values from values in the active notebook. The Multiply option multiplies the values together, and the Divide option divides active notebook values by the incoming values. Subtract, Multiply, and Divide have the same requirements and rules as Add.

See Also

Combining Files

Copying Combined File Data

Importing and Exporting Files

Extracting Part of a Notebook

Tools|Extract saves part of a notebook to a separate file, leaving the original file intact.

If you prefer, you can open a blank notebook in a new window and copy a block of data to it with the Clipboard commands or with Block|Copy rather than extracting the block.

Tools|Extract is similar to Block|Copy, with two major differences:

With Extract you can choose to copy values only. When you choose its Values option, the actual formulas aren't copied--just the resulting values.

Extract saves the notebook's block names and graphs along with the specified block. Some block names or graphs may not be meaningful if they refer to cells outside the extracted block. You can delete them, reassign them, or ignore them.

The Tools|Extract dialog box has Formulas and Values options. Formulas saves an exact copy of the block, including formulas. Values converts formulas to their end values.

When you load a file saved with the Formulas option, formulas adjust to reflect their new positions, even if they are absolute.. If a formula refers to cells outside the block being saved, its results may be inaccurate. In this case, it's best to use the Values option.

To extract part of a notebook,

1. Choose Tools|Extract.
2. In the Block(s) edit field, enter the block name or coordinates to be saved. You can specify a noncontiguous block by separating the subblocks with commas.
3. Choose Formulas to save the block exactly as is, or choose Values to save the resulting values instead of the original formulas.
4. Enter the file name you want using the techniques described in File Handling Options.

The extracted data begins at cell A1, regardless of its position in the original file.

If you include the file-name extension used by a different program, the file is translated into the appropriate file format. For a list of acceptable extensions, see Translating Files. To create a tab-delimited text file, use the extension .TXT.

For the extracted data to work with the program, the extracted block must be set up in a way that makes sense to the program. For example, to extract data to a Paradox file, the block should list related data in columns (which are interpreted as fields), and should contain column headings, (which are interpreted as field names).

See Also

Combining Files

Block|Copy Command

Edit|Cut Command

Edit|Paste Command

Importing and Exporting Files

Printing Tasks

These topics describe how to preview and print notebook data and graphs:

Setting Up a Printer

Printing Notebooks

Basic Notebook Printing

Defining a Print Block

Setting Margins and Paper Type

Changing Print Size

Inserting Page Breaks

Entering Headers and Footers

Adding Headings

Print Settings

Previewing a Print Job

Printing to a Binary File

Printing Cell Contents

Printing Graphs

Setting Up a Printer

Before printing from Quattro Pro, use the Windows Control Panel to specify the printers available; the Control Panel also enables printing from all Windows applications. If multiple printers are attached to your system, you can use File|Printer Setup to specify the printer to use, as follows:

1. Choose File|Printer Setup.
2. Select the printer to use from the Printer and Port list box.
3. If desired, use Setup to configure the printer (this is discussed next).
4. Choose OK.

Print defaults

By choosing the Setup command from the File|Printer Setup dialog box, you can specify default print settings such as:

 paper type (wide, letter, envelope)

 paper source on the printer (manual, bin 1, bin 2)

 print orientation (landscape, portrait)

 output size using a percentage (50%, 200%, and so on)

 pen colors in a plotter

 number of copies

 fonts available in the printer (cartridge fonts, soft fonts, internal fonts)

 output resolution (300 dpi, 150 dpi)

 color or black-and-white printing

 the amount of memory in the printer

Since many of these settings are printer-specific, they won't all be available for your printer. Consult your Windows documentation and printer manual for more information on these and additional settings.

Note: Changes made to print defaults affect printing in all Windows applications. You can use File|Page Setup or the Options dialog box of File|Print to set many of these options (output size, paper type, number of copies) for only Quattro Pro.

See Also

Printing Notebooks

Printing Tasks

Printing Notebooks

When a notebook window is active, you can use File|Print to print out parts of the notebook. You can print a simple list or a sophisticated multiple-page report, embellished with headers, footers, and page numbers.

File|Page Setup and the Options command of File|Print govern how output is arranged and produced on the printed page. You can enhance the document's appearance by using block properties in the notebook to add lines, new typefaces, and shading.

See Also

Changing Block and Page Properties

Basic Notebook Printing

File|Page Setup Command

File|Print Command

Printing Tasks

Basic Notebook Printing

To print a notebook, follow these steps:

1. Choose File|Page Setup to adjust print settings.
2. Select the block(s) to print.
3. Choose File|Print. The selected blocks appear in Print Blocks.
4. To print part of a document (for example, to print pages three through seven of a twenty-page document), click From and enter the starting page in the edit field next to it. Click the edit field to the right of To and enter the ending page. The default is All Pages.
5. Specify the number of copies to print in the Copies edit field.
6. If desired, use Options to add headings, grid lines or row and column borders to the document. See Adding Headings for more details.
7. Choose Print to print the data. (If you'd like to print the data later, choose Close to save the current print settings.)
8. Quattro Pro displays a message showing Quattro Pro's progress while sending information to the Print Manager. If you want to stop the print job, choose Cancel, or use the Windows Print Manager to delete the print job after Quattro Pro sends it. If your printer has a large memory buffer, printing may continue for a few moments after cancellation.

Print settings are saved with the notebook, so subsequent printing is easier. In most cases, you can just choose Print from the File|Print dialog box. You can also store print settings under a name for easy retrieval.

See Also

Print Settings

File|Print Command

Printing Tasks

Defining a Print Block

There are several ways to specify which parts of the notebook print:

When choosing File|Print for the first time in a notebook, don't select a block; Quattro Pro sets the print block to all the data in the active page.

Select the block(s) and choose File|Print. If a cell's contents spill over into adjacent cells onscreen, include the spillover cells in the selection; otherwise, only part of the entry will print. The Print Blocks option of the dialog box reflects the selection.

Choose Print Blocks from the File|Print dialog box, and then point to or enter the block(s) to print.

By default, the first column of each contiguous block within the print block prints flush against the left margin.You can check Center Blocks in the File|Page Setup dialog box to center each block between the left and right margins of the printed page.

Note: Extraneous blank columns on the right side of a print block will affect centering by printing the visible data in the block closer to the left margin. Don't include these blank columns in the print block if you want the data in the block to appear centered.

Noncontiguous Blocks

Each subblock that makes up a noncontiguous block prints as though it was selected individually. You can use Print Between Blocks (in the Options dialog box of File|Print) to specify how much space to leave between each subblock's printout. To start each subblock on a new printed page, choose Page Advance. Choosing Lines separates subblocks by the number of lines specified in the Lines edit field. Each line is a sixth of an inch.

Printing 3-D Blocks

If a block spans more than one notebook page, Quattro Pro prints it as though each page was selected individually. You can use Print Between 3D Pages (in the Options dialog box of File|Print) to specify how much space to leave between each notebook page's printout. To start each subblock on a new printed page, choose Page Advance. Choosing Lines separates subblocks by the number of lines specified in the Lines edit field. Each line is a sixth of an inch.

Printing Floating Objects

Any floating objects (graphs, SpeedButtons, and so on) in the print block are printed with the notebook data. Only the selected portion prints; if the floating object covers a page break, the image splits, printing on multiple pages.

See Also

Setting Margins and Paper Type

File|Print Command

Printing Tasks

Setting Margins and Paper Type

You can change margins to best present your data--to center a small block of data on the page, for example.

Note: Use File|Page Setup to set margins and paper type. You can also set margins by dragging margin lines in a print preview. See Previewing a Print Job for more details.

The following list describes all the margin settings available. Defaults are given in inches. (Scroll down for information on using centimeters to specify margins.)

Top	Specifies how much room to leave between the top of each printed page and the header. The default is a third of an inch.
Header	(in Margins) Specifies how much room to leave between the top margin and the first row of data. The header prints in the top of this space. The default is half an inch.
Left	Specifies how much room to leave between the left edge of the paper and the first column of data. The default is four-tenths of an inch.
Right	Specifies how much room to leave between the right edge of the paper and the last column of data (if it fills the page horizontally). The default is four-tenths of an inch.
Footer	(in Margins) Specifies how much room to leave between the bottom margin and the last row of data (if it fills the page vertically). The footer prints in the bottom of this space. The default is half an inch.
Bottom	Specifies how much room to leave between the bottom of each page and the footer. The default is a third of an inch.
Paper Type	Specifies the type of paper loaded into your printer.

To change the margin settings,

1. Choose File|Page Setup.
2. Choose the margin you want to adjust: Left, Top, Bottom, Right, Header, or Footer.
3. For Left, Right, Top, and Bottom, enter the distance from the edge of the page. Use inches if **in** is displayed next to the default margin settings; use centimeters if **cm** is displayed. Use decimals to indicate partial inches or centimeters, such as 0.5 for half. For Header, specify the distance between the top margin and the first row of data; for Footer, the distance between the last row of data and the bottom margin.
4. Choose OK to save the new margin settings.

Using Centimeters

Quattro Pro uses the measurement system specified in the Windows Control Panel to determine whether to use inches or centimeters for margin measurements. You can override the default for a particular margin by entering **in** (for inches) or **cm** (for centimeters) after its value. The default suffix is displayed next to each default margin setting. For example, to set the left margin to three centimeters when the default measurement system is inches, choose Left Margin and enter **3 cm**. Quattro Pro then converts the entry into the default measurement system.

Changing Print Orientation

Normally, Quattro Pro prints data vertically on the page. The direction data prints is called the orientation. Using Print Orientation (in File|Page Setup), you can specify the direction to print data:

Portrait prints the data vertically, on individual pages.

Landscape prints the data horizontally, on individual pages.

When switching orientation, margins switch as well; for example, the top margin stays with the top of the notebook data printed, along with the header and top heading (if any).

See Also

Entering Headers and Footers

File|Print Command

Printing Tasks

Changing Print Size

Scaling (in File|Page Setup) lets you specify a percentage (1 to 1000) to increase or decrease the size of notebook data on the printed page. The margins (except for header and footer margins) don't change.

Print To Fit (in File|Page Setup) shrinks the print block, headers, footers, and headings so that they fit on as few pages as possible. The margins (except for header and footer margins) don't change. Print To Fit only reduces text to fit on fewer pages--it doesn't enlarge text to fill the page. Print To Fit ignores (and doesn't change) the current setting of Scaling (under File|Page Setup), if any.

See Also

Previewing a Print Job

Entering Headers and Footers

Setting Margins and Paper Type

Printing Tasks

Inserting Page Breaks

There are two kinds of page breaks that occur in printing with Quattro Pro: soft page breaks and hard page breaks.

Quattro Pro breaks the notebook data up and prints it on separate pages to avoid running text off the bottom or right of the page. These breaks, called soft page breaks, free you from having to insert hard page breaks to print.

Hard page breaks are those you enter yourself by

Typing the characters **|::** into a cell in the first column of a print block.

Selecting a cell in the first column of a print block and choosing Block|Insert Break. This also inserts a new row into the notebook.

Wherever hard page breaks are placed, the paper advances accordingly. In the case of a noncontiguous block, you can enter the hard page break into the first column of any block included in the print block.

Caution: Don't add any data into a row containing a hard page break. The data won't print.

Unchecking Break Pages (in File|Page Setup) makes Quattro Pro print the data as though one large page is being printed; successive pages are treated like pieces of the same page. This also

disables headers and footers

disables top, bottom, header, and footer margins

Note: Unchecking Break Pages doesn't disable hard page breaks.

Printing Large Notebooks

If the data in the print block won't fit on one printed page, Quattro Pro fits as many columns as possible across the first page, and prints all rows in those columns using as many pages as necessary. Then, Quattro Pro returns to the first row of data and picks up at the column where it left off, until all data in the block has printed. You can combine the pages to create one large notebook.

Quattro Pro won't split a column. If the full column won't fit, it prints on the next page. Use File|Print Preview to see where your pages will break.

See Also

Previewing a Print Job

Setting Margins and Paper Type

Changing Print Size

Printing Tasks

Entering Headers and Footers

Headers and footers are lines of text that print at the top and bottom of each page.

You can also print specific rows or columns on each printed page as headings; see Adding Headings for details.

Quattro Pro separates headers and footers from the notebook data by the amount of space specified in the Header and Footer margin controls, minus the height of the header or footer text. Unless you specify otherwise, headers and footers are left-aligned.

Note: Unchecking Break Pages (in File|Page Setup) disables the header and footer, along with their margins.

To enter a header or footer,

1. Choose Header or Footer from the File|Page Setup dialog box.
2. Enter the text you want to appear as the header and/or footer. Use Backspace or other editing keys to correct mistakes as you type.
3. If you want, choose Header Font and specify the header and/or footer's typeface. Header Font affects both the header and footer. See Header Font for details.
4. Choose OK to save the new header and/or footer.

To change a header or footer, choose Header or Footer again and edit or retype the entry.

Formatting Header and Footer Text

The following table lists special characters you can use in headers and footers to insert dates, times, file names, calculations based on the page number, and multiple lines of text.

Code	Description
|	(vertical bar) Determines the position of the text: left-aligned, right-aligned, or centered.
#d	Enters the current date in the short format specified in the Date Format option of the International property in the application Object Inspector.
#D	Enters the current date in the long format specified in the Date Format option of the International property in the application Object Inspector.
#ds	Enters the current date in Short Date format
#Ds	Enters the current date in Long Date format.
#t	Enters the current time in the short format specified in the Time Format option of the International property in the application Object Inspector.
#T	Enters the current time in the long format specified in the Time Format option of the International property in the application Object Inspector.
#ts	Enters the current time in Short Time format.
#Ts	Enters the current time in Long Time format.
#p	Enters the current page number.
#p+n	Enters the current page number plus the number n.
#P	Enters the number of pages in the document.
#P+n	Enters the number of pages plus the number n.
#f	Enters the name of the notebook printing with no path (BUDGET.WB1).
#F	Enters the name of the notebook printing, including the path (C:\DATA\BUDGET.WB1).
#n	Prints the remainder of the header or footer on a second line.

#	(number sign) Enters the current page number. Use for compatibility with Quattro Pro for DOS.
@	Enters the current date (per your computer's calendar). Use for compatibility with Quattro Pro for DOS.

The vertical bar character|works like a tab; one preceding vertical bar centers the text; two preceding vertical bars **||** right-align the text. To enter this character on most keyboards, hold down Shift and press the backslash key (\). On some keyboards, the vertical bar character appears broken in the middle.

You can also use|characters to align parts of the header or footer. Up to two vertical bar characters can be in a header or footer line.

See Also

Previewing a Print Job

Setting Margins and Paper Type

Changing Print Size

Adding Headings

Printing Tasks

Adding Headings

In addition to headers and footers, you can specify column or row headings to print on each page. Row headings print down the left edge of the page and are called left headings; column headings print at the top of each page, below any specified headers, and are called top headings. To print top or left headings on each page,

1. Choose Options from the File|Print dialog box.
2. Choose Left Heading (for a column of headings) or Top Heading (for a row of headings).
3. To specify a left heading, select the address of any cell in the column to use; the entire column is used as the heading regardless of which cell you select. To specify a top heading, select the address of any cell in the row to use; the entire row is used regardless of which cell you select. Selecting a block makes all columns (for a left heading) or all rows (for a top heading) in the block print as headings.
4. Choose OK to save the new heading.

Caution: Don't include top or left headings in the print block, or they'll print twice. For example, if you've specified A2..G2 as a top heading, and you want to print all data through row 30, specify A3..G30 as the print block.

See Also

Previewing a Print Job

Setting Margins and Paper Type

Changing Print Size

Printing Tasks

Print Settings

You can check Gridlines in the Options dialog box to print the gridlines that normally appear on the notebook page. If Gridlines is unchecked, only lines added using block properties will print.

You can check Row/Column Borders to print the row and column borders that display on a notebook page.

Both the Options dialog box of File|Print and the File|Page Setup dialog box can be reset to default settings using the Reset Defaults button. Reset Defaults doesn't clear named print settings.

Named Print Settings

You may routinely print different documents from a given notebook. Changing the current print settings manually each time you print a different document can be tedious. You can store the current print settings under a name using File|Named Settings. All options in the File|Print and File|Page Setup dialog boxes save under the name specified.You can have multiple names in a notebook, and they are saved with the notebook. To store the current print settings under a name,

1. Specify the print settings as you would normally.
2. Choose File|Named Settings.
3. Choose Create, and enter the name for the new setting.
4. The new named setting appears in the list.
5. Choose Close to save the new name.

Now whenever you want to print using the settings stored under the name, choose File|Named Settings, select the name containing the settings to use, and choose OK. This replaces the current print settings with those stored under the name. If you change any of the current settings, you must update the settings stored under the name. To store the current print settings under an existing name,

1. Choose File|Named Settings.
2. Select the name to update from the list.
3. Choose Update to replace the settings stored under the name with the current print settings.
4. Choose Close to save the change.

To remove a name,

1. Choose File|Named Settings.
2. Choose the name to delete from the list.
3. Choose Delete to remove the name from the list.
4. Choose Close to save the change.

Since named settings are saved with the notebook, use File|Save to save your file after changing them.

See Also

Setting Margins and Paper Type
Changing Print Size
Inserting Page Breaks
Entering Headers and Footers
Adding Headings
Printing Tasks

Previewing a Print Job

You can use File|Print Preview (or the Preview button in the File|Print dialog box) to see an onscreen preview of how the document will appear.

The current zoom level displays to the right of Zoom. When previewing, left-clicking the page zooms in a level, increasing detail; right-clicking zooms out a level, decreasing detail. You can use scroll bars to adjust the section of the page viewed.

The SpeedBar provides a variety of tools for controlling a document while previewing. To identify each tool, point to it; its name displays at the bottom of the preview window. For more information, display Object Help. Point to a SpeedBar object, hold down Ctrl, and right-click the mouse.

See Print Preview Window Keys for a list of keys you can use while previewing.

See Also

Printing Notebooks

Printing Graphs

Printing Tasks

Printing to a Binary File

If you want to print from a machine that doesn't have Quattro Pro installed, or if you take your printing to a service bureau with high quality printers, you can store the document in a file using Redirect Output To File (in the File|Printer Setup dialog box).This file, called a binary file, contains instructions in the printer's native language for creating the document. For example, if your printer is a PostScript printer, Quattro Pro creates a PostScript file. If your printer is an HP LaserJet, it creates a PCL (Printer Control Language) file.

To create a binary file,

1. Choose File|Printer Setup and select the printer that the document will eventually be sent to.
2. Check the check box below Redirect Output To File.
3. Enter the file name in the edit field. If you don't specify a file extension, the extension .PRN is used. To store the file in a directory other than the default, or on a floppy disk, include the complete DOS path.
4. Choose OK.

Printing in Quattro Pro now sends the output to the binary file (overwriting the old file each time), not the printer. Choose File|Printer Setup again and uncheck the check box to revert to normal printing.

To print a binary file from DOS, use the DOS COPY command with the /B parameter:

```
COPY filename.PRN /B LPT1
```

This sends a binary file (/B) to the LPT1 printer port. If your printer is connected to a different port (such as COM1 or PRN), specify it instead of LPT1.

See Also

Setting Up a Printer

Printing Tasks

Printing Cell Contents

You'll usually print data as it appears onscreen. You can, however, print each cell's contents instead by checking Cell Formulas (in the Options dialog box of File|Print).

Cell Formulas lists the contents of each cell, one per line, just as they appear on the input line when you select a cell. This includes each cell's address, and contents as entered, as well as any comments you've added to the regular cell contents. The information is printed in the font specified under Header Font (in the File|Page Setup dialog box).

When you print using Cell Formulas, the following print features are disabled: Center Blocks, Left and Top heading, Gridlines, and Row/Column borders.

See Also

Entering Formulas

Adding Comments to Entries

Printing Tasks

Printing Graphs

Printing a graph is the same as printing notebook data, except:

You can set the graph's aspect ratio before printing (see Graph Window Properties for details).

When you choose File|Print, a simplified dialog box appears when a graph window is active or an icon in the Graphs page is selected; certain controls in File|Page Setup are dimmed to indicate they're unavailable.

You can print multiple graphs from the Graphs page by selecting their icons before printing. You can also print all graphs in a slide show by selecting its icon first.

Graph print settings can't be stored under a name like notebook print settings. (The current graph print settings are saved with the notebook.)

Printing multiple graphs

You can print multiple graphs one after the other using the Graphs page:

1. Go to the Graphs page.
2. Select each graph's icon (by holding down the Shift key while clicking each of them). You can also select a slide show icon to print all the graphs in it.
3. Use File|Page Setup to adjust print settings, if you want to make changes.
4. Choose File|Print and then Print to print the graphs in the order in which they were selected.

See Also

Setting Margins and Paper Type
Entering Headers and Footers
Print Settings
Previewing a Print Job
Printing to a Binary File
Printing Tasks

Blocks and Pages

Changing Block and Page Properties

Using Styles

Using SpeedFormat

Block Properties

Setting Numeric Format

Choosing a Font

Shading Blocks

Aligning Cell Entries

Drawing Lines Around Blocks

Removing Block Protection

Changing Text Color

Limiting Data Input Types

Resizing Rows and Columns

Hiding Rows and Columns

Page Properties

Naming a Page

Enabling Page Protection

Changing Line Color

Coloring Conditional Cells

Setting Label Alignment

Suppressing Zeros

Setting Default Column Width

Removing Borders

Hiding Grid Lines

Changing Page Tab Color

Changing Block and Page Properties

Block properties control numeric format of values, font, shading, alignment of entries, line drawing, block protection, text color, the types of data you can type into a cell, row height, column width, and whether columns or rows are hidden.

The Style list and the SpeedFormat button in the SpeedBar provide quick ways to apply combinations of block properties.

Page properties control the page name, overall page protection, line color, colors in cells that meet various conditions, default label alignment, whether zeros are displayed, default column width, row and column borders, and grid lines.

To change block properties,

1. Select the block you want to affect.
2. With the mouse pointer anywhere in the selected block, click the right mouse button. Or choose Property|Current Object.

To change page properties, do one of the following:


Right-click the page tab.


Select the page tab with the left mouse button, then choose Property|Active Page.

Note: Although it's tempting to preset block properties in large areas of the notebook, doing so consumes memory. It's more efficient to set properties only in the cells or pages you're currently using. Or, change the Normal style to change default properties.

If you move a cell, its properties move with the data and are removed from the original cell. If you copy a cell, the copy takes on the properties of the original cell. If you paste a copied or cut block with Edit|Paste Special, you can control whether the cell's properties are pasted (see Using Paste|Special for details).

See Also

Defining Styles

Using Styles

Using Styles

Quattro Pro contains several predefined styles, each of which is a specific property setting (or settings) identified by a name. Applying the style is faster than using the block Object Inspector.

You apply styles to a selected block with the Style list in the SpeedBar. If you've previously set properties with the block Object Inspector, the style doesn't disturb those settings. The styles included with Quattro Pro are as follows:

Comma	Sets Numeric Format to Comma with two decimals
Comma0	Sets Numeric Format to Comma with no decimals
Currency	Sets Numeric Format to Currency with two decimals
Currency0	Sets Numeric Format to Currency with no decimals
Date	Sets Numeric Format to User Defined (with the date format specified; see Defining Custom Numeric Formats for more information)
Fixed	Sets Numeric Format to Fixed with two decimals
Heading 1	Sets Font to Arial (or Helvetica) 18 point bold
Heading 2	Sets Font to Arial (or Helvetica) 12 point bold
Normal	Sets Numeric Format and Alignment to General; sets Font to Arial (or Helvetica) 10 point regular; sets Shading to White; sets Line Drawing to None; turns Protection on; and sets Text Color to Black.
Percent	Sets Numeric Format to Percent
Total	Inserts a double line above the active cell

See Also
Defining Styles

Using SpeedFormat

Instead of setting properties in individual cells, you can choose a predefined set of properties for different parts of an entire block. This set of properties is called a format, which you apply with the SpeedFormat button in the SpeedBar.

The different parts of the block that a format can affect are:

- column headings (cells in the top row).
- column totals (cells in the bottom row).
- row headings (cells in the left column).
- row totals (cells in the right column).
- the body (remaining cells in the block).

A format consists of a series of property settings for each of the five parts of a block. You can apply a format to the body and selectively to any of the other four parts.

You can also control which properties of a format to apply. Only the checked properties are applied in the body and in the checked parts of the block.

To apply a format,

1. Select the block to be formatted and click the SpeedFormat button.
2. Choose a format from the SpeedFormat dialog box.
3. Uncheck any properties you don't want applied in the block.
4. Uncheck any parts (column headings, row headings, column totals, or row totals) you don't want included as part of the format.
5. Choose OK.

The format is applied to the body and to any of the other four parts you checked.

Using SpeedFormat works similarly to applying styles in that currently existing properties in the block are undisturbed. For example, if you use SpeedFormat to shade cells and later want to remove that shading, you need to right-click that block, choose Block Properties, then select the default setting for the Shading property in the Object Inspector. Simply reusing SpeedFormat and unchecking Shading doesn't change the block's shading.

See Also

Using Styles

Defining Styles

Setting Numeric Format

When you enter a number in an unformatted cell, Quattro Pro displays it according to the numeric format setting determined by the Normal style. By default, this setting is General, which displays numbers exactly as you enter them (unless the column width is too narrow). A number of other formats are available, including formats that add commas, dollar signs, or other characters to your original number.

Note: All numeric formats leave the cell values intact; they only affect the way values are displayed.

To quickly change to one of the more common numeric formats, choose from the styles in the SpeedBar's Style list. Only some of the styles in the list affect the numeric format; the others set other properties. See Using Styles for a description of styles supplied with Quattro Pro.

To choose one of the other numeric formats available,

1. Right-click the cells you want to format and choose the Properties command to display the Object Inspector. Numeric Format is already selected.
2. Choose one of the formats listed.
3. If the format you chose allows a variable number of decimal places, an edit field appears. If you want to display other than the default number of decimal places (2), enter a number from 0 to 15 in the edit field.
4. If you choose Date, you must choose a specific date format.
5. If you choose Time, you must choose a specific time format.
6. If you choose User Defined, you can choose from a list of formats you create yourself. For instructions on creating numeric formats, see Defining Custom Numeric Formats.
7. Choose OK.

If the format you choose creates a number too wide for its column, a string of asterisks appears. You can redisplay the number by widening the column.

The format you assign to a cell stays with the cell, even if you delete its contents. If you move the contents, however, the format moves with the data and is removed from the original cell. If you copy a formatted cell, the copy takes on the format of the original cell. If you paste a copied or cut block with Edit|Paste Special, you can control whether the cell's properties are pasted.

Click Numeric Format Options for descriptions and examples of each numeric format.

See Also
Using Paste|Special
Column Width

Choosing a Font

You can quickly switch the selected block to bold or italic, or change its font size with controls on the SpeedBar.

Click **b** to choose bold, ***i*** for italic. To toggle bold or italic off, click the button again. Click the up or down arrows to increase or decrease the font size.

With the Font property, you can choose from a variety of fonts, sizes, and options.

To assign a font to a block, right-click the block, choose Block Properties to display an Object Inspector, then choose Font. Choose the font, font size, and options you want to use.

If a font has a ***TT*** (TrueType) or ***a*** (ATM) symbol beside it, text in that font will appear in print just as it does onscreen. Fonts with a printer symbol will print on your printer, but may not appear in the correct font onscreen. Fonts without a symbol appear accurately onscreen, but may not appear in correct font when printed.

If you enlarge or reduce the font size, the row height changes to display the tallest letters in the row (unless you've set the row height explicitly; see Row Height for details).

See Also
Using Styles

Shading Blocks of Cells

The Shading property controls the color of cells. The shading color is a mixture of two colors: Color 1 and Color 2. You use the Blend squares to control how much of each color to include in the mix.

To shade a block,

1. Right-click the block, choose Block Properties, then choose Shading in the Object Inspector.
2. Choose the two colors to be mixed by clicking color squares in Color 1 and Color 2.
3. Choose the Blend square that provides the mix you want.

With some color choices, you need to deselect the block to see the new shading. This is because selected cells are shown in reverse color.

To change the colors available, use the notebook Palette property (see Palette for details).

Note: If shading doesn't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

See Also
Using Styles

Aligning Cell Entries

When you enter data into a cell, Quattro Pro aligns it according to the default alignment setting, which is called General. The General setting right-aligns values and left-aligns labels.

To quickly change alignment, select the block you want to affect, then click the left, center, or right alignment button in the SpeedBar.

Instead, you can right-click the block, choose Block Properties, then choose Alignment in the Object Inspector. Choose General, Left, Right, Center, or Center Across Block to align the data. Center Across Block centers text across the cell containing it and all contiguous selected cells to the right of that cell, as long as they don't contain data.

You can also align an individual label by preceding it with a label-prefix character (see Aligning Labels for more information).

Finally, you can set the alignment of all labels later entered in the page with the page Label Alignment property (see Label Alignment for details). This setting determines the affect (on labels only) of the General alignment setting described here. The Label Alignment property setting is overridden by any individual alignment you've set.

See Also
Using Styles

Drawing Lines Around Blocks

With the Line Drawing property, you can draw lines around cells in a block. To draw lines in a block, right-click the block, choose Block Properties, then choose Line Drawing in the Object Inspector. As shown in the Line Draw Property Example, first choose a line type on the right, then click in the sample block to indicate where you want to place that type of line. You can also apply the chosen line type in a preset pattern by clicking one of the three pattern boxes.

The lines you draw in the sample box indicate where lines will be drawn in the selected block. For example, clicking the outer edges of the sample box draws lines only on the outside of the selected block; clicking the inner lines in the sample box draws lines between every row or column in the selected block.

You can combine line drawing options in the same block. For example, you can draw a double-lined border around a block, and single vertical lines between the columns in the block.

To remove lines, choose the No Line type, then click the lines you want to remove in the sample block.

If you change your mind as you're specifying lines, click the No Change line type and click the line you recently changed. It returns the line to the line type in effect when you last chose OK. To cancel all changes, choose Cancel.

To change the screen color of lines, change the page Line Color property (see Line Color for details).

See Also
Using Styles

Removing Block Protection

The page Protection property (click Enabling Page Protection for details) lets you enable or disable overall page protection. You can then remove protection from individual blocks of cells with the block Protection property. That way, you can enter data in those cells, leaving the remainder of the page protected.

To remove or restore protection in a block of cells in a page that has already been protected with the page Protection property, right-click the block of cells,choose Block Properties, then choose Unprotected or Protected.

When a cell is protected, you can't edit, replace, or delete its contents. Nor can you delete a column or row that contains a protected cell.

Note: When you disable protection with the page Protection property, no cells are protected, even those you protected individually with the block Protection property.

With page protection enabled, you can move the selector around the entire page, but you can make changes only to unprotected cells. To restrict the selector to unprotected cells only, use Data|Restrict Input (see Restrict Input for details).

See Also

Using Styles

Using Named Blocks

Changing Text Color

The Text Color property controls the color of cell entries This property controls the color of the characters that make up the data, not the color of the cell shading.

To change the color of cell entries in a block, right-click the cells,choose Block Properties, then choose Text Color in the Object Inspector. Choose the color you want. If you already changed the cell's shading, make sure the Text Color setting will contrast enough to be visible.

Note: If text doesn't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

To change the colors available, use the notebook Palette property (see Palette for details). If you create a new color for the palette that is dithered and later choose it for the Text Color property, your text will appear in a substitute, nondithered color. You can use dithered colors for other elements besides text and drawn lines.

Tip: With most color choices, you need to deselect the block to see the new shading. This is because selected cells are shown in reverse color.

See Also
Using Styles

Limiting Data Types

You can force a block of cells to accept only labels or only dates and times. This feature is especially helpful if you're setting up a notebook for others to use. For example, presetting cells for label-only entry makes it easier to enter phone numbers and social security numbers because they contain hyphens that are otherwise interpreted as minus signs.

To choose the type of data that can be typed into a cell, right-click the block and choose Block Properties, then choose Data Entry Input in the Object Inspector. Choose General, Labels Only, or Dates Only. (Dates Only restricts entries to dates or times.)

If someone types a date, time, or number in a label-only cell, it's converted to a label. If someone tries to type a label or number into a date-only cell, an error message appears.

To return to unlimited data entry, right-click the limited block, choose Block Properties, then choose General for the Data Entry Input setting.

For the list of acceptable date and time formats, see Entering Dates and Times.

See Also

Search Commands

Database Commands

Search Procedure

Using Styles

Resizing Rows and Columns

There are several ways to change the width of columns. You can

 use the Fit button in the SpeedBar to tailor column width to the longest entry in the column. See Setting Automatic Row & Column Sizes for details.

 drag the row or column border with the mouse. See Resizing Rows & Columns with the Mouse for details.

 specify an exact numerical setting with the block Column Width and Row Height properties. See Setting Exact Row & Column Sizes for details.

You can also hide rows or columns from being displayed, yet retain the data for other cells to reference for calculation in the notebook.

Finally, you can set a global width for all columns in the active page with the page Default Width property (see Setting Default Column Width for details).

See Also
Using Styles

Setting Automatic Row & Column Sizes

You can adjust column width to one character wider than the longest entry in a given column. To do this, click the column border and click the Fit button in the SpeedBar. You can also select multiple columns at the same time (they must be contiguous columns) and click Fit to adjust them simultaneously.

To adjust a column based on a group of cells, select them and click Fit. To adjust based on the longest entry in a cell or any cell in the same column below it, select the cell and click Fit.

To further tailor a column's width, you can specify the amount of space to add to the longest entry, using the Column Width property (see Setting Exact Row & Column Sizes for details).

See Also
Resizing Rows & Columns with the Mouse

Resizing Rows & Columns with the Mouse

You can easily change column or row size by dragging its border. You can use this method to resize single or multiple rows or columns.

To resize a single row or a single column, move the cursor over the right edge of the column border to be resized, or over the bottom edge of the row border to be resized. The cursor turns into a double-arrow. Then drag the double-arrow until the row or column has reached the size you want.

You can also resize several rows or several columns in the same page to a uniform size. Select contiguous rows or columns (by dragging in their borders), or select noncontiguous rows or columns (by clicking their borders while holding down the Ctrl key). Decide on a row or column within the selection that you want to govern the uniform size. Then drag the double-arrow at the right of the governing column (or if you're resizing rows, drag the double-arrow at the bottom of the governing row). All the rows or columns are resized to the same dimension even if they started out with different sizes.

See Also

Setting Automatic Row & Column Sizes

Setting Exact Row & Column Sizes

Setting Exact Row & Column Sizes

With the Column Width and Row Height properties, you can adjust single or multiple columns, or single or multiple rows to an exact size. You can also resize rows or columns back to their default size. In the case of columns, you can set an automatic width based on the longest entry plus the amount of space you choose.

To adjust column widths or row heights, select any cell in each column or row you want to resize, or select their borders. You can also select noncontiguous columns or rows.

Next, right-click the selected block, choose Block Properties, then choose Column Width in the Object Inspector, and check Set Width. Then click Characters, Inches, or Centimeters, and enter the number you want in the Column Width edit field. To return column width to the default, choose Reset Width instead.

Note: The setting of the page Default Width property determines the default column width. Columns whose widths you've explicitly adjusted (with the Fit button, the mouse or the Column Width property) are not controlled by the Default Width property (see Setting Default Column Width for details).

You can use the Auto-Width option to resize contiguous columns based on the longest entry instead. The longest entry is chosen based on your initial column or block selection:


If entire columns are selected, the width is based on the longest entry in each column.


If a multi-row block (or just part of a column) is selected, the width in each column is based on the longest entry in each column of the block.


If a single-row block (or just one cell) is selected, the width is based on the longest entry in that row and all cells below it.

The Auto-Width option works like the Fit button in the SpeedBar, but it goes one step further: You can specify the amount of extra column width space to be added beyond the longest entry.

To use the Auto-Width option, select in the notebook as described above, then right-click the block and choose Block Properties. Choose Column Width and Auto-Width in the Object Inspector. Then choose Characters, Inches, or Centimeters and enter the number of extra characters to add to the longest entry.

Tip: If a column you're adjusting contains a long entry that spills over into blank cells to the right, and you don't want the column adjusted to that cell entry's length, specify a multi-row block that stops short of the cell.

To resize rows, right-click them, choose Block Properties, then choose Row Height. Click Points, Inches, or Centimeters, and enter the number you want in the Row Height edit field. To return the row height to the default (as determined by the largest font used in the row), choose Reset Height instead.

See Also

Setting Automatic Row & Column Sizes

Resizing Rows & Columns with the Mouse

Hiding Rows and Columns

Occasionally, you may want to temporarily remove rows or columns of data from view and from printouts, but still use the data in calculations. With the Reveal/Hide property, you can hide rows or columns from view without losing the data they contain. You can later redisplay the rows or columns with the same property.

To hide rows or columns from view, select a cell in the row or column you want to hide. If you want to hide multiple rows or columns, select a block that encompasses them. Right-click the block, choose Block Properties, and choose Reveal/Hide in the Object Inspector. Choose Rows or Columns and choose Hide.

Columns to the right of the hidden columns move left to fill in the empty space, or rows below the hidden rows move up. However, the identifying row numbers and column letters in the borders don't change. In other words, if you hide column B, the columns onscreen are labeled A, C, D, and so on.

If you use Tools|Extract to save part of a notebook that includes hidden rows or columns (see Extracting Part of a Notebook for details), Quattro Pro saves the hidden rows or columns in the new file, although they will still be hidden from view when you load the file.

To return one or more hidden rows or columns to the screen, right-click a block containing cells on both sides of the hidden area, choose Block Properties, and choose Reveal/Hide. Then choose Rows or Columns and choose Reveal.

You can also reveal a single hidden row or column with the mouse. To reveal a hidden column, place the mouse pointer slightly to the right of the hidden column's border, then drag. For a hidden row, drag from just below the hidden row's border. The hidden row or column will be revealed and sized as you drag.

Naming Pages

Pages are initially named with letters of the alphabet in sequence, from A to Z, continuing from AA to AZ, up to IV. You can give pages descriptive names up to 15 characters long for easier identification. These names are then used in formula references.

To assign a name to a page, right-click its tab. Name is already selected. You can use letters and numbers in the name, as well as the following special characters

~ ` ! % _ | \ ' ?

You can't use spaces or any other special characters. Also, you can't use a name you've previously assigned to a group in the same notebook.

After you choose OK, the new name appears on the tab. Formulas that refer to the renamed page adjust to use the new name. When you point to cells in this page from other pages to build formula references, the new page name appears in the input line.

If, after assigning a page name, you want to rename it to its original letter name, choose Reset.

Note: The Graphs page (the last page in the notebook) cannot be renamed.

See Also
Using Blocks in Formulas

Enabling Page Protection

With the page Protection property, you can prevent changes from being made to data. The page Protection property works in tandem with the block Protection property (see Removing Block Protection for details). Use page protection to set up protection for a page, then use block protection to unprotect the specific blocks of cells you want to allow to be changed.

To turn on protection in a page, right-click its tab and choose Protection. Choose Enable.

To turn off protection, choose Disable. When page protection is disabled, Quattro Pro ignores the status of blocks explicitly protected or unprotected with the block Protection property.

To prevent unauthorized access to an entire notebook, protect it with a password (see Assigning a Password to a File for details).

Changing Line Color

You can draw lines around cells or blocks with the block Line Drawing property (see Line Drawing for details). By default, the lines are black.

To change the color of lines in a page, right-click its tab and choose Line Color. Then choose the color you prefer.

Note: If lines don't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

To change the colors available, use the notebook Palette property (see Palette for details). If you create a new color for the palette that is dithered and later choose it for the Line Color property, your lines will appear in a substitute, nondithered color. You can use dithered colors for other elements besides text and drawn lines.

Coloring Conditional Cells

With the page Conditional Color property, you can change the color of specific types of data: values above or below a specified range, and ERR values. For example, you can use it to display all negative values in red or all values greater than 1000 in green. Choose from these settings in the Conditional Color property:

Smallest Normal Value and Greatest Normal Value let you enter a range of values you consider normal. Cells with values within this range will appear in the Normal Color. The default settings are 0 (smallest) and 1E+300 (largest). You can specify different colors for values above and below this range by clicking the option and clicking a color square in the color palette.

Below Normal Color sets the color of cells whose values are below the Smallest Normal Value.

Normal Color sets the color of cells whose values fall within the range set by Smallest Normal Value and Greatest Normal Value.

Above Normal Color sets the color of cells whose values are above the Greatest Normal Value.

ERR Color specifies the color to use for ERR and NA values generated by formula errors.

Enable indicates whether to use the colors set with this menu.

To turn off conditional colors, uncheck Enable.

To change the colors available, use the notebook Palette property (see Palette for details).

Note: If colored entries don't appear on your printout, you may need to choose a darker color. Light colors print as white on some printers.

See Also
Changing Text Color

Setting Label Alignment

The Label Alignment property setting determines the alignment of labels in the active page. Initially, the setting is Left.

Label entries already existing in the page don't adjust. To change the alignment of existing data, use the Alignment buttons in the SpeedBar or the block Alignment property. All new labels will be aligned according to the new default (unless you precede them with a different label-prefix character). Values (including string values that result from some @functions) are not affected by label alignment settings.

To change the default alignment of labels in a page, right-click its tab and choose Label Alignment. Then choose Left, Center, or Right.

The setting you make here determines the effect of the General setting of the block Alignment property.

See Also

Aligning Labels

Aligning Cell Entries

Suppressing Zeros

With the Display Zeros property, you can suppress the display of any value that equals exactly zero, whether it was entered directly or calculated with a formula.

To suppress zeros from displaying on a page, right-click the page tab and choose Display Zeros. Then choose No. Zero suppression doesn't remove the zero values from the page. They remain in memory and reappear if you set Display Zeros to Yes.

A value must equal exactly zero to be suppressed. For example, a value such as .004 that appears as 0 if decimal precision is 2 or less, will still appear if Display Zeros is set to No, because it is not actually 0.

Caution: When zero suppression is on, it's easy to accidentally write over cells containing formulas that evaluate as zero. Make sure Undo is enabled (see Enabling the Undo Command for more information), or consider protecting the page (see Enabling Page Protection for details).

Setting Default Column Width

By default, columns are wide enough to display approximately nine characters in the default font. You can adjust the width of all columns in the active page by changing the Default Width property setting.

To change the default width of columns for a page, right-click its tab and choose Default Width. Choose Characters, Inches, or Centimeters and enter the number of units in the Column Width edit field.

The Default Width setting doesn't affect columns that were explicitly adjusted using the Fit button, the block Column Width property, or the mouse. Before those columns can be affected by a change in the default width, you must select them and check the Reset Width option of the block Column Width property.

See Also
Resizing Rows and Columns

Removing Row and Column Borders

If you don't need the row and column borders (for example, if your page is set up as a form for data input), you can remove them from the screen with the Borders property.

To remove borders from a page, right-click its tab and choose Borders. Then uncheck Row Borders and/or Column Borders.

Hiding Spreadsheet Grid Lines

The spreadsheet grid displays by default. It separates rows and columns; each rectangle in the grid is a cell. If you don't want to display the grid, you can remove it.

To remove grid lines from a page, right-click its tab and choose Grid Lines. Then uncheck Horizontal and/or Vertical.

Changing Page Tab Color

Color-coding page tabs can help organize your notebooks. For example, all pages for a sales region or calendar quarter might be the same color.

The default page tab color is white. To change the color,

1. Point to a tab and inspect it to display the active page Object Inspector.
2. Point to the color you want and click the mouse.
3. Click OK to apply that color to the active page.

You can create custom colors. For details, see Creating Custom Colors.

Building Custom SpeedBars

The following topics explain how to build custom SpeedBars with the SpeedBar Designer:

SpeedBar Designer Guidelines

Using Button Palettes

Setting Button Properties

Adding Custom Buttons and Labels

Assigning Macros

Removing SpeedBar Objects

Setting SpeedBar Properties

Saving Custom SpeedBars

Closing SpeedBars

Testing SpeedBars and Dialog Boxes

Editing Custom SpeedBars

Using Custom SpeedBars

Exiting SpeedBar Designer

See Also

Application Building

SpeedBar Designer Guidelines

Quattro Pro supplies many standard SpeedBars for push-button access to commands and procedures. They change to suit your current activity, and you can add or remove SpeedBars at any time.

As you work with Quattro Pro, you'll probably use some commands and features more often than others. The SpeedBar Designer offers a simple way to create custom SpeedBars for features you use the most--even special macros. You can choose standard buttons from palettes or build your own. Each button has its own Object Help for easy identification. Standard buttons even link to detailed help topics for more information.

If you need to create dialog boxes or more complex SpeedBars, consider using the UI Builder. It's a useful tool for application developers who need its full power and flexibility. For details, see Application Building.

The Basic Procedure

Follow these steps to create a custom SpeedBar with standard buttons:

1. Choose Tools|SpeedBar Designer to display a blank SpeedBar and SpeedBar design tools.
2. Display one or more standard button palettes.
3. Click a button on the palette two times (the first click activates the palette), then click the new SpeedBar two times to copy the button and position it.
4. Move new buttons into final position.
5. Save the new SpeedBar.
6. Test the new SpeedBar to make sure it works the way you expect. Edit the buttons if necessary.

Once you've created a SpeedBar with standard or custom buttons, you can copy its buttons to other custom SpeedBars. If you want to build a custom SpeedBar that's similar to another, you can edit an existing custom SpeedBar and save it under another name. For detailed directions, see the other Custom SpeedBar topics.

See Also

Building Custom SpeedBars

Using Button Palettes

The SpeedBar Designer includes a number of standard button palettes based on activities performed with the main menu options. They are named in the Button Palette list.

Loading Button Palettes

To load a button palette, choose it in the Button Palette list. If it hides an open SpeedBar, choose SpeedBar|Tile.

As you point to each button on a palette, its name appears at the bottom of the window. For a description, point to the button, hold down Ctrl, then right-click to display Object Help.

Copying Buttons

To copy a button from the active palette to a custom SpeedBar, click the button in the palette, then point to the SpeedBar and click. (If the target palette isn't already active, click the button once to activate the palette, then again to copy the button.)

A positioning pointer appears. Point to the location where you want to place the new button and click again. The button appears, surrounded by handles. To move it, click inside the handles and drag it into position. You can copy buttons from one custom SpeedBar to another the same way.

If you start to copy a button and change your mind, click the Selection tool in the Designer SpeedBar to cancel the operation.

See Also

Building Custom SpeedBars

Setting Button Properties

A copied button has the same properties as the original. You can change the properties of any custom button or label on a custom SpeedBar. To inspect or change the properties of a selected custom object, right-click it and choose the Properties command. The Object Inspector lists some of the properties in the following table, depending on whether the object is a push button, bitmap button, or label.

Property	Settings
Bitmap	Bitmap to display on the bitmap button; choose Browse to search for other bitmaps
Dimension	Distance of the object's upper left corner from the left edge and top of the SpeedBar; the object's width and height (all in pixels)
Help Line	Name or hint text to display at the bottom of the window when the button is pointed to
Label Font	Typeface, size, and style of text for the label object
Label Text	Text to display on the button or label
Object Help	Title and text to display in the object's Object Help window; developers with Winhelp generation capability can specify a help topic jump
Text Draw Flags	Word-wrapping toggle, vertical and horizontal alignment; check Apply to add text as specified

You can also view and set properties of the SpeedBar itself. For details, see Setting SpeedBar Properties.

See Also

Adding Custom Buttons and Labels

Building Custom SpeedBars

Adding Custom Buttons and Labels

Besides standard buttons, you can add these objects to custom SpeedBars:


Push buttons are rectangular button buttons with text.


Bitmap buttons are rectangular button buttons with bitmaps and optional text.


Labels are text for description and identification.

To add an object to a custom SpeedBar,

1. Click one of the object tools in the Designer SpeedBar, then click the custom SpeedBar. The new button or label appears surrounded by handles.
2. Move the selected object, if necessary.
3. Right-click the selected object, choose the Properties command, and set its properties.
4. If the object is a push button or bitmap button, make it perform the desired action by assigning a macro to it as described in the next section.

See Also

Assigning Macros

Setting Button Properties

Building Custom SpeedBars

Assigning Macros

Push buttons and bitmap buttons work by activating assigned macros. The macro can be complex--for example, an application involving several linked notebooks--or simple, even a single command equivalent such as {WindowTitles "Clear"}, which clears locked titles.

To assign a macro to a SpeedBar object, click the object. Then choose Speedbar|Assign Macro to display the Assign Macro dialog box. Enter the page, block, and notebook location of the macro to assign, then click OK.

If you assign a macro to a standard button, it performs its standard operation then runs the macro. For example, you can assign a cell containing {BEEP} to the Paste button from the Edit palette. Then, each time you use that button, it beeps when the paste operation is finished.

See Also
Building Custom SpeedBars

Removing SpeedBar Objects

To remove a button or label from a custom SpeedBar, click the object then press Del or click Cut in the Designer SpeedBar.

Standard buttons are always available for reuse but before you delete a button you've created, consider saving it to a scrap SpeedBar first. Choose SpeedBar|New to open a new SpeedBar, then copy the unwanted button to it. Save that SpeedBar to a meaningful name--SCRAP.BAR, for example--and keep obsolete custom buttons there for future use or redesign.

See Also
Building Custom SpeedBars

Setting SpeedBar Properties

You can also set properties for the entire SpeedBar. To inspect or change the properties of a selected custom SpeedBar, right-click its title. The Object Inspector lists these properties:

Property	Settings
Dimension	SpeedBar size and location; only Height (in pixels) applies when the SpeedBar is docked
Title	Name to display in the SpeedBar title bar
Position Adjust	Specifies how the SpeedBar moves when the Quattro Pro window is resized; doesn't apply when the SpeedBar is docked
Grid Options	Shows/hides a positioning grid and sets the number of pixels between grid lines
Name	Used to identify the SpeedBar in macros, link commands, and formulas
Disabled	Disables all controls in the SpeedBar

By default, all SpeedBar properties are unchecked; this is appropriate for most purposes. You can also set properties for custom SpeedBar buttons and labels.

See Also

Setting Button Properties
Building Custom SpeedBars

Saving Custom SpeedBars

To save a custom SpeedBar, choose SpeedBar|Save or click the Save button in the Designer SpeedBar.

Use SpeedBar|Save As to copy a custom SpeedBar by saving it under another name.

See Also
Building Custom SpeedBars

Closing Custom SpeedBars

To close a SpeedBar you're building or editing, make sure it's selected then click the Remove SpeedBar button in the Designer SpeedBar. If you haven't saved that SpeedBar yet, you'll see a confirmation prompt. Click OK to delete the SpeedBar without saving it.

If you prefer, click the Control-menu box in the upper left corner of the active SpeedBar and choose Close, or double-click the Control-menu box.

See Also
Building Custom SpeedBars

Testing SpeedBars and Dialog Boxes

You can test a SpeedBar without leaving the SpeedBar Designer. Follow these same procedures to test a SpeedBar or dialog box in the UI Builder:

1. Click the Test button or choose Test in the Control-menu box. The active SpeedBar or dialog box enters Test mode. You can restore minimized notebooks, then see if buttons or other controls work as expected.
2. If changes are needed, double-click the Control-menu box to resume Edit mode. Make any necessary edits then test and save the SpeedBar or dialog box as before.

See Also

Building Custom SpeedBars

Application Building

Editing Custom SpeedBars

To edit a closed SpeedBar, choose it in the Custom SpeedBar list. To search other directories, click <Browse>. Then, add and delete tools just as if you were building a new SpeedBar.

When you're done, choose SpeedBar|Save or click the Save button. Choose SpeedBar|Save As to save the edited SpeedBar under another name.

If you want, you can continue editing custom SpeedBars in the UI Builder. Click the UI Builder button to display its SpeedBar and use its features.

See Also

Building Custom SpeedBars

Application Building

Using Custom SpeedBars

While editing SpeedBars, you can "dock" the selected bar--install it for use--without leaving the SpeedBar Designer. Choose SpeedBar|Dock or choose Dock in the Control-menu box for that SpeedBar. The new bar appears near the top of the Quattro Pro window; it is no longer in Edit mode and is ready to use. When the Designer SpeedBar is closed, you can dock a custom SpeedBar by displaying a SpeedBar Control menu and choosing Append, Insert, or Replace (see SpeedBar Control Menu for details).

After docking a custom SpeedBar, you can instantly return it to Edit mode. Display its SpeedBar Control menu, then choose Customize. The Designer SpeedBar appears at the top of the window and the custom SpeedBar is "undocked" and ready to edit.

See Also
Building Custom SpeedBars

Exiting SpeedBar Designer

When you've finished with the SpeedBar Designer, exit by closing all custom SpeedBars or activating another type of window. For example, you can use SpeedBar|Close All to exit the SpeedBar Designer, or you can click the minimized notebook icon at the bottom of the window.

See Also
Building Custom SpeedBars

Using Quattro Pro Windows

These topics describe how to use Quattro Pro's Window menu to select and handle windows within Quattro Pro:

Selecting a Window

If you have more than one window open, you can select the one you want to work in with commands in the Windows menu. When a window is selected, it is referred to as the active window. The window with the highlighted title bar is active.

There are three ways to select an open window:

 Click any part of the window.

 Choose the window from the list at the bottom of the Window menu. A checkmark appears next to the name of the active window.

 Press Ctrl+F6 to move into the next window (in the sequence in which they were created). If that window is beneath other windows, it moves to the top.

See Also

Window Menu

Using Quattro Pro Windows

Duplicating a Window

The Window|New View command displays a duplicate copy of the active notebook in a new window. This is useful when you want to look at different pages in the notebook at the same time, or if you want to see distant parts of one page at the same time. You can create as many duplicate views of a notebook as you want.

You can also see two parts of a notebook at the same time by splitting a window into two panes, as described in Splitting a Window into Panes. Unlike duplicate view, however, you can have only two panes.

To duplicate a window, select it and choose Window|New View.

The duplicate window appears full size in front of other open windows, with cell A1 selected on the first page. Now you can resize the windows so you can see parts of both of them (see Resizing Windows for details). Then scroll them or select different pages as desired.

Most changes you make to either view (whether changes to data or to most properties) are shown in all views. Only changes to remove grid lines or borders (using the page Object Inspector), setting up locked titles, creating panes, or changing the zoom factor (using the notebook Object Inspector) aren't shown in all views.

See Also
Window Menu
Using Quattro Pro Windows

Resizing and Arranging Windows

You can use standard Windows techniques to resize Quattro Pro windows. For instructions on dragging corners or borders of windows with the mouse, choosing commands in the Control menu, or using the Maximize, Minimize, and Restore buttons, see the Windows documentation.

Tiling Windows

The Windows|Tile command displays all open windows without overlapping them. When possible, the windows are all given equal room on the screen.

Cascading Windows

The Window|Cascade command rearranges all open windows in overlapping layers. The top line of each window is revealed so you can see the name of the notebook, graph, or dialog box it contains.

Arranging Icons

After minimizing several windows and moving their icons around in the Quattro Pro desktop, you can quickly place the icons in neat rows with the Arrange Icons command.

After you choose Windows|Arrange Icons, the icons are lined up in a row along the bottom of the screen.

See Also

Window Menu

Using Quattro Pro Windows

Hiding and Showing Windows

You can hide open windows from view. This is handy when you're working with linked notebooks, but don't want the screen cluttered with too many windows. It also affords you confidentiality when you need it.

To hide a window, select it (see Selecting A Window for details) and choose Window|Hide.

To show a hidden window, choose Window|Show. Then choose the name of the window to show, and choose OK.

See Also

Window Menu

Using Quattro Pro Windows

Splitting a Window into Panes

To view different parts of the same notebook, you can duplicate the window, or split the window into two panes. The panes can be vertically or horizontally split.

To divide the window into panes,

1. Move the mouse pointer to the lower-right corner of the window over the pane splitter.

2. The pointer changes to a black double-arrow. Depending on where you position the mouse pointer, the double-arrow points horizontally or vertically.
3. For horizontal panes, display the vertical double-arrow and drag up. Release the mouse button when the dotted line reaches the separation point you'd like. For vertical panes, follow the same procedure, but use the double-arrow that points horizontally and drag left.

The active pane is the one containing the selector. To select the other pane, click anywhere in the pane, or press the Pane key, F6. After you unsynchronize the panes (see Resizing Panes for details), you can scroll the panes independently to display different parts of the notebook.

The following topics explain how to control panes:

Resizing Panes

Unsynchronizing Panes

Closing the Second Pane

See Also

Duplicating a WindowTS_WIN_NEWVIEW

Window Menu

Using Quattro Pro Windows

Resizing Panes

To resize panes,

1. Move the mouse pointer to the pane splitter at the lower right of the left or top pane until the double-arrow appears.
2. Drag to the new position where you'd like the first pane to end.

You can also split the window into panes using Window|Panes. The window is split at the position of the selector.

1. Select the row or column where you want the window to be split.
2. Choose Window|Panes.
3. Choose Horizontal to split the window horizontally at the active row. Choose Vertical to split the window vertically at the active column.

Note: You can point across panes while entering formulas or using dialog boxes.

Some display changes you make in one pane don't affect the appearance of the other pane:


Grid lines, Borders, and Default Column Width properties (in the page Object Inspector)


Row Height, Column Width, and Reveal/Hide properties (in the block Object Inspector)


Creating locked titles with Windows|Locked Titles

For example, if you change column widths in one pane, the previous widths remain in the other. When you return the window display to one pane, Quattro Pro retains column width changes only if they were made in the top or left pane.

See Also

Unsynchronizing Panes

Closing the Second Pane

Duplicating a Window

Splitting a Window into Panes

Using Quattro Pro Windows

Unsynchronizing Panes

By default, panes are synchronized; when you scroll one pane, the other scrolls at the same time. Horizontal panes are synchronized horizontally (columns scroll together) and vertical panes are synchronized vertically (rows scroll together).

You can change this so each pane scrolls independently of the other. This lets you view one part of the notebook in one pane, while you scroll to another part in the other.

To synchronize or unsynchronize panes, choose Window|Panes (either before or after splitting the window). Then uncheck Synchronized to be able to scroll the panes independently, or check it to restore synchronization.

See Also

Resizing Panes
Closing the Second Pane
Duplicating a Window
Using Quattro Pro Windows

Closing the Second Pane

To remove the second pane, choose Window|Panes, then choose Clear.

You can also remove the second pane with the mouse:

1. Move the mouse pointer to the pane splitter between the first and second pane so a black double-arrow appears.
2. Drag the double-arrow back to the top or bottom edge of the window (for horizontal panes) or to the left or right edge (for vertical panes).

The pane on the bottom or right disappears, and the top or left pane again takes up the entire notebook window. Any column width changes, locked titles, or columns that were hidden or revealed in the top or left pane remain in effect.

You can also remove the second pane by choosing Window|Panes, then choosing Clear.

See Also

Resizing Panes

Unsynchronizing Panes

Duplicating a Window

Using Quattro Pro Windows

Locking Rows and Columns

The Window|Locked Titles command locks specific rows and/or columns of a spreadsheet page onscreen as titles. When you scroll the spreadsheet, the titles remain fixed onscreen while the rows below (or columns to the right) scroll as usual.

Locking titles has no effect when printing. To repeat column or row titles on each page of a printed notebook, use the Top and Left heading options in the Options dialog box of File|Print.

To lock titles onscreen,

1. Scroll the spreadsheet so the column(s) or row(s) you want as titles are visible at the upper left of the window. You can't adjust the position of the titles after they're locked.
2. To lock rows, select the row below the last row to be locked. To lock columns, select the column to the right of the last column to be locked. To lock rows and columns at the same time, select the top left cell of the part you want to remain scrollable.
3. Choose Window|Locked Titles.
4. Choose Horizontal to lock all rows above the selector, Vertical to lock all columns to the left, or Both to lock both rows above and columns to the left.

Any previous Window|Locked Titles setting is cleared, and Quattro Pro locks the specified columns or rows in place as titles.

To unlock titles without specifying new ones, choose Window|Locked Titles and choose the Clear option.

Before you can edit data within a locked title, you must choose Edit|GoTo and specify a cell within the locked title area. This creates an adjacent duplicate copy of the locked areas. Any changes you make to the duplicate cells are reflected in the locked titles.

To remove a duplicate column, scroll the window horizontally until it disappears. To remove a duplicate row, scroll vertically.

Note: As with hidden and widened columns, locked titles affect the active pane only. If the window is divided into two panes, locked titles are retained after you close the second pane only if you create them in the top or left pane.

See Also

File|Print Command

Window Menu

Using Quattro Pro Windows