

Microsoft Excel Macro Functions Contents

[Alphabetical List of Macro Functions](#)

[Macro Functions Listed by Category](#)

[New and Changed Macro Functions Listed by Category](#)

[Visual Basic Equivalents for Macro Functions and Commands](#)

[Changing Links to Microsoft Excel version 4.0 addins](#)

[See Also](#)

Alphabetical List of Macro Functions

A1.R1C1
ABSREF
ACTIVATE
ACTIVATE.NEXT
ACTIVATE.PREV
ACTIVE.CELL
ACTIVE.CELL.FONT
ADD.ARROW
ADD.BAR
ADD.CHART.AUTOFORMAT
ADD.COMMAND
ADD.MENU
ADD.OVERLAY
ADD.TOOL
ADD.TOOLBAR
ADDIN.MANAGER
ALERT
ALIGNMENT
ANOVA1
ANOVA2
ANOVA3
APP.ACTIVATE.MICROSOFT
APP.ACTIVATE
APP.MAXIMIZE
APP.MINIMIZE
APP.MOVE
APP.RESTORE
APP.SIZE
APP.TITLE
APPLY.NAMES
APPLY.STYLE
ARGUMENT
ARRANGE.ALL
ASSIGN.TO.OBJECT
ASSIGN.TO.TOOL
ATTACH.TEXT
ATTACH.TOOLBARS
AUTO.OUTLINE
AXES
BEEP
BORDER
BREAK
BRING.TO.FRONT
CALCULATE.DOCUMENT
CALCULATE.NOW
CALCULATION

CALLER
CANCEL.COPY
CANCEL.KEY
CELL.PROTECTION
CHANGE.LINK
CHART.ADD.DATA
CHART.TREND
CHART.WIZARD
CHECK.COMMAND
CLEAR
CLEAR.OUTLINE
CLEAR.ROUTING.SLIP
CLOSE
CLOSE.ALL
COLOR.PALETTE
COLUMN.WIDTH
COMBINATION
CONSOLIDATE
CONSTRAIN.NUMERIC
COPY
COPY.CHART
COPY.PICTURE
COPY.TOOL
CREATE.NAMES
CREATE.OBJECT
CREATE.PUBLISHER
CUSTOM.REPEAT
CUSTOM.UNDO
CUT
DATA.DELETE
DATA.FIND
DATA.FIND.NEXT
DATA.FIND.PREV
DATA.FORM
DATA.LABEL
DATA.SERIES
DEFINE.NAME
DEFINE.STYLE
DELETE.ARROW
DELETE.BAR
DELETE.CHART.AUTOFORMAT
DELETE.COMMAND
DELETE.FORMAT
DELETE.MENU
DELETE.NAME
DELETE.OVERLAY
DELETE.STYLE
DELETE.TOOL

DELETE.TOOLBAR
DEMOTE
DEREF
DESCR
DIALOG.BOX
DIRECTORY
DISABLE.INPUT
DISPLAY
DOCUMENTS
DUPLICATE
ECHO
EDIT.COLOR
EDIT.DELETE
EDIT.OBJECT
EDIT.REPEAT
EDIT.SERIES
EDIT.TOOL
EDITION.OPTIONS
ELSE
ELSE.IF
EMBED
ENABLE.COMMAND
ENABLE.TIPWIZARD
ENABLE.TOOL
END.IF
ENTER.DATA
ERROR
ERRORBAR.X
ERRORBAR.Y
EVALUATE
EXEC
EXECUTE
EXPON
EXTEND.POLYGON
EXTRACT
FCLOSE
FILE.DELETE
FILES
FILL.AUTO
FILL.DOWN
FILL.GROUP
FILL.LEFT
FILL.RIGHT
FILL.UP
FILTER
FILTER.ADVANCED
FILTER.SHOW.ALL
FIND.FILE

FONT
FONT.PROPERTIES
FOPEN
FOR
FOR.CELL
FORMAT.AUTO
FORMAT.CHART
FORMAT.CHARTTYPE
FORMAT.FONT
FORMAT.LEGEND
FORMAT.MAIN
FORMAT.MOVE
FORMAT.NUMBER
FORMAT.OVERLAY
FORMAT.SHAPE
FORMAT.SIZE
FORMAT.TEXT
FORMULA.ARRAY
FORMULA.CONVERT
FORMULA.FILL
FORMULA.FIND
FORMULA.FIND.NEXT
FORMULA.FIND.PREV
FORMULA.GOTO
FORMULA.REPLACE
FOURIER
FPOS
FREAD
FREADLN
FREEZE.PANES
FSIZE
FTESTV
FULL
FULL.SCREEN
FUNCTION.WIZARD
FWRITE
FWRITELN
GALLERY.3D.AREA
GALLERY.3D.BAR
GALLERY.3D.COLUMN
GALLERY.3D.LINE
GALLERY.3D.PIE
GALLERY.3D.SURFACE
GALLERY.AREA
GALLERY.BAR
GALLERY.COLUMN
GALLERY.CUSTOM
GALLERY.DOUGHNUT

GALLERY.LINE
GALLERY.PIE
GALLERY.RADAR
GALLERY.SCATTER
GET.BAR
GET.CELL
GET.CHART.ITEM
GET.DEF
GET.DOCUMENT
GET.FORMULA
GET.LINK.INFO
GET.NAME
GET.NOTE
GET.OBJECT
GET.PIVOT.FIELD
GET.PIVOT.ITEM
GET.PIVOT.TABLE
GET.TOOL
GET.TOOLBAR
GET.WINDOW
GET.WORKBOOK
GET.WORKSPACE
GOAL.SEEK
GOTO
GRIDLINES
GROUP
HALT
HELP
HIDE
HIDE.OBJECT
HISTOGRAM
HLINE
HPAGE
HSCROLL
IF
INITIATE
INPUT
INSERT.OBJECT
INSERT.PICTURE
INSERT.TITLE
JUSTIFY
LAST.ERROR
LEGEND
LINK.FORMAT
LINKS
LIST.NAMES
MAIL.ADD.MAILER
MAIL.DELETE.MAILER

MAIL.EDIT.MAILER
MAIL.FORWARD
MAIL.LOGOFF
MAIL.LOGON
MAIL.NEXT.LETTER
MAIL.REPLY
MAIL.REPLY.ALL
MAIN.CHART
MAIN.CHART.TYPE
MCORREL
MCOVAR
MENU.EDITOR
MERGE.STYLES
MESSAGE
MOVE
MOVEAVG
MOVE.TOOL
NAMES
NEW
NEW.WINDOW
NEXT
NOTE
OBJECT.PROPERTIES
OBJECT.PROTECTION
ON.DATA
ON.DOUBLECLICK
ON.ENTRY
ON.KEY
ON.RECALC
ON.SHEET
ON.TIME
ON.WINDOW
OPEN
OPEN.DIALOG
OPEN.LINKS
OPEN.MAIL
OPEN.TEXT
OPTIONS.CALCULATION
OPTIONS.CHART
OPTIONS.EDIT
OPTIONS.GENERAL
OPTIONS.LISTS.ADD
OPTIONS.LISTS.DELETE
OPTIONS.TRANSITION
OPTIONS.VIEW
OUTLINE
OVERLAY
PAGE.SETUP

PARSE
PASTE
PASTE.LINK
PASTE.PICTURE
PASTE.PICTURE.LINK
PASTE.SPECIAL
PASTE.TOOL
PATTERNS
PAUSE
PIVOT.ADD.DATA
PIVOT.ADD.FIELDS
PIVOT.FIELD
PIVOT.FIELD.GROUP
PIVOT.FIELD.PROPERTIES
PIVOT.FIELD.UNGROUP
PIVOT.ITEM.PROPERTIES
PIVOT.ITEM
PIVOT.REFRESH
PIVOT.SHOW.PAGES
PIVOT.TABLE.WIZARD
PLACEMENT
POKE
PRECISION
PREFERRED
PRESS.TOOL
PRINT
PRINT.PREVIEW
PRINTER.SETUP
PROMOTE
PROTECT.DOCUMENT
PTTESTM
PTTESTV
PUSHBUTTON.PROPERTIES
QUERY.GET.DATA
QUERY.REFRESH
QUIT
RANDOM
RANKPERC
REFTEXT
REGISTER
REGRESS
RELREF
REMOVE.PAGE.BREAK
RENAME.COMMAND
RENAME.OBJECT
REPLACE.FONT
REPORT.DEFINE
REPORT.DELETE

REPORT.GET
REPORT.PRINT
REQUEST
RESET.TOOL
RESET.TOOLBAR
RESTART
RESULT
RESUME
RETURN
ROUTE.DOCUMENT
ROUTING.SLIP
ROW.HEIGHT
RUN
SAMPLE
SAVE
SAVE.DIALOG
SAVE.AS
SAVE.COPY.AS
SAVE.TOOLBAR
SAVE.WORKBOOK
SAVE.WORKSPACE
SCALE
SCENARIO.ADD
SCENARIO.CELLS
SCENARIO.DELETE
SCENARIO.EDIT
SCENARIO.GET
SCENARIO.SHOW
SCENARIO.SHOW.NEXT
SCENARIO.SUMMARY
SCROLLBAR.PROPERTIES
SELECT
SELECT.ALL
SELECT.CHART
SELECT.END
SELECT.LAST.CELL
SELECT.PLOT.AREA
SELECT.SPECIAL
SELECTION
SEND.KEYS
SEND.MAIL
SEND.TO.BACK
SERIES
SERIES.AXES
SERIES.ORDER
SERIES.X
SERIES.Y
SET.CRITERIA

SET.DATABASE
SET.EXTRACT
SET.NAME
SET.PAGE.BREAK
SET.PREFERRED
SET.PRINT.AREA
SET.PRINT.TITLES
SET.UPDATE.STATUS
SET.VALUE
SHORT.MENUS
SHOW.ACTIVE.CELL
SHOW.BAR
SHOW.CLIPBOARD
SHOW.DETAIL
SHOW.INFO
SHOW.LEVELS
SHOW.TOOLBAR
SIZE
SLIDE.COPY.ROW
SLIDE.CUT.ROW
SLIDE.DEFAULTS
SLIDE.DELETE.ROW
SLIDE.EDIT
SLIDE.GET
SLIDE.PASTE
SLIDE.PASTE.ROW
SLIDE.SHOW
SOLVER.ADD
SOLVER.CHANGE
SOLVER.DELETE
SOLVER.FINISH
SOLVER.GET
SOLVER.LOAD
SOLVER.OK
SOLVER.OPTIONS
SOLVER.RESET
SOLVER.SAVE
SOLVER.SOLVE
SORT
SOUND.NOTE
SOUND.PLAY
SPELLING
SPELLING.CHECK
SPLIT
SQL.BIND
SQL.CLOSE
SQL.ERROR
SQL.EXEC.QUERY

SQL.GET.SCHEMA
SQL.OPEN
SQL.RETRIEVE
SQL.RETRIEVE.TO.FILE
STANDARD.FONT
STANDARD.WIDTH
STEP
STYLE
SUBSCRIBE.TO
SUBTOTAL.CREATE
SUBTOTAL.REMOVE
SUMMARY.INFO
TABLE
TERMINATE
TEXT.BOX
TEXT.TO.COLUMNS
TEXTREF
TRACER.CLEAR
TRACER.DISPLAY
TRACER.ERROR
TRACER.NAVIGATE
TTESTM
UNDO
UNGROUP
UNHIDE
UNLOCKED.NEXT
UNLOCKED.PREV
UNREGISTER
UPDATE.LINK
VBA.INSERT.FILE
VBA.MAKE.ADDIN
VIEW.3D
VIEW.DEFINE
VIEW.DELETE
VIEW.GET
VIEW.SHOW
VLINE
VOLATILE
VPAGE
VSCROLL
WAIT
WHILE
WINDOW.MAXIMIZE
WINDOW.MINIMIZE
WINDOW.MOVE
WINDOW.RESTORE
WINDOW.SIZE
WINDOW.TITLE

WINDOWS
WORKBOOK.ACTIVATE
WORKBOOK.ADD
WORKBOOK.COPY
WORKBOOK.DELETE
WORKBOOK.HIDE
WORKBOOK.INSERT
WORKBOOK.MOVE
WORKBOOK.NAME
WORKBOOK.NEW
WORKBOOK.NEXT
WORKBOOK.OPTIONS
WORKBOOK.PREV
WORKBOOK.PROTECT
WORKBOOK.SCROLL
WORKBOOK.SELECT
WORKBOOK.TAB.SPLIT
WORKBOOK.UNHIDE
WORKGROUP
WORKSPACE
ZOOM
ZTESTM

Macro Functions Listed by Category

Command-Equivalent

Control

Customizing

Database & List Management

DDE/External

Engineering

Information

Lookup & Reference

Statistical

Text

New and Changed Macro Functions Listed by Category

Command-Equivalent

Customizing

Database & List Management

DDE/External

Information

Statistical

Text

See Also

Worksheet Functions

Visual Basic Reference

Command-Equivalent Macro Functions

A1.R1C1
ACTIVATE
ACTIVATE.NEXT
ACTIVATE.PREV
ACTIVE.CELL.FONT
ADD.ARROW
ADD.OVERLAY
ADDIN.MANAGER
ALIGNMENT
APP.MAXIMIZE
APP.MINIMIZE
APP.MOVE
APP.RESTORE
APP.SIZE
APPLY.NAMES
APPLY.STYLE
ARRANGE.ALL
ASSIGN.TO.OBJECT
ATTACH.TEXT
ATTACH.TOOLBARS
AUTO.OUTLINE
AXES
BORDER
BRING.TO.FRONT
CALCULATE.DOCUMENT
CALCULATE.NOW
CALCULATION
CANCEL.COPY
CELL.PROTECTION
CHANGE.LINK
CHART.ADD.DATA
CHART.WIZARD
CLEAR
CLEAR.OUTLINE
CLEAR.ROUTING.SLIP
CLOSE
CLOSE.ALL
COLOR.PALETTE
COLUMN.WIDTH
COMBINATION
CONSOLIDATE
CONSTRAIN.NUMERIC
COPY
COPY.CHART
COPY.PICTURE

CREATE.NAMES
CREATE.OBJECT
CREATE.PUBLISHER
CUT
DATA.LABEL
DATA.SERIES
DEFINE.NAME
DEFINE.STYLE
DELETE.ARROW
DELETE.FORMAT
DELETE.CHART.AUTOFORMAT
DELETE.NAME
DELETE.OVERLAY
DELETE.STYLE
DEMOTE
DISPLAY
DUPLICATE
EDIT.COLOR
EDIT.DELETE
EDIT.REPEAT
EDIT.SERIES
ENABLE.TIPWIZARD
ERRORBAR.X
ERRORBAR.Y
EXTEND.POLYGON
FILE.CLOSE
FILE.DELETE
FILL.AUTO
FILL.DOWN
FILL.GROUP
FILL.LEFT
FILL.RIGHT
FILL.UP
FILTER
FILTER.ADVANCED
FILTER.SHOW.ALL
FIND.FILE
FONT
FONT.PROPERTIES
FORMAT.AUTO
FORMAT.CHART
FORMAT.CHARTTYPE
FORMAT.FONT
FORMAT.LEGEND
FORMAT.MAIN
FORMAT.MOVE
FORMAT.NUMBER
FORMAT.OVERLAY

FORMAT.SHAPE
FORMAT.SIZE
FORMAT.TEXT
FORMULA.FILL
FORMULA.FIND
FORMULA.FIND.NEXT
FORMULA.FIND.PREV
FORMULA.GOTO
FORMULA.REPLACE
FREEZE.PANES
FULL
FULL.SCREEN
FUNCTION.WIZARD
GALLERY.3D.AREA
GALLERY.3D.BAR
GALLERY.3D.COLUMN
GALLERY.3D.LINE
GALLERY.3D.PIE
GALLERY.3D.SURFACE
GALLERY.AREA
GALLERY.BAR
GALLERY.COLUMN
GALLERY.CUSTOM
GALLERY.DOUGHNUT
GALLERY.LINE
GALLERY.PIE
GALLERY.RADAR
GALLERY.SCATTER
GOAL.SEEK
GRIDLINES
GROUP
HIDE
HIDE.OBJECT
HLINE
HPAGE
HSCROLL
INSERT
INSERT.PICTURE
INSERT.TITLE
JUSTIFY
LEGEND
LINK.FORMAT
LINKS
LIST.NAMES
INSERT
MAIL.ADD.MAILER
MAIL.DELETE.MAILER
MAIL.EDIT.MAILER

MAIL.FORWARD
MAIL.LOGOFF
MAIL.LOGON
MAIL.NEXT.LETTER
MAIL.REPLY
MAIL.REPLY.ALL
MAIN.CHART
MAIN.CHART.TYPE
MENU.EDITOR
MERGE.STYLES
MOVE
NEW
NEW.WINDOW
NOTE
OBJECT.PROPERTIES
OBJECT.PROTECTION
OPEN
OPEN.LINKS
OPEN.MAIL
OPEN.TEXT
OPTIONS.CALCULATION
OPTIONS.CHART
OPTIONS.EDIT
OPTIONS.GENERAL
OPTIONS.LISTS.ADD
OPTIONS.LISTS.DELETE
OPTIONS.TRANSITION
OPTIONS.VIEW
OUTLINE
OVERLAY
PAGE.SETUP
PARSE
PASTE
PASTE.LINK
PASTE.PICTURE
PASTE.PICTURE.LINK
PASTE.SPECIAL
PATTERNS
PIVOT.ADD.DATA
PIVOT.ADD.FIELDS
PIVOT.FIELD
PIVOT.FIELD.GROUP
PIVOT.FIELD.PROPERTIES
PIVOT.FIELD.UNGROUP
PIVOT.ITEM.PROPERTIES
PIVOT.ITEM
PIVOT.REFRESH
PIVOT.SHOW.PAGES

PIVOT.TABLE.WIZARD
PLACEMENT
PRECISION
PREFERRED
PRINT
PRINT.PREVIEW
PRINTER.SETUP
PROMOTE
PROTECT.DOCUMENT
QUERY.GET.DATA
QUERY.REFRESH
QUIT
REMOVE.PAGE.BREAK
RENAME.OBJECT
REPLACE.FONT
REPORT.DEFINE
REPORT.DELETE
REPORT.PRINT
RESUME
ROUTE.DOCUMENT
ROUTING.SLIP
ROW.HEIGHT
RUN
SAMPLE
SAVE
SAVE.AS
SAVE.COPY.AS
SAVE.WORKBOOK
SAVE.WORKSPACE
SCALE
SCENARIO.ADD
SCENARIO.CELLS
SCENARIO.DELETE
SCENARIO.EDIT
SCENARIO.SHOW
SCENARIO.SHOW.NEXT
SCENARIO.SUMMARY
SELECT
SELECT.ALL
SELECT.CHART
SELECT.END
SELECT.LAST.CELL
SELECT.PLOT.AREA
SELECT.SPECIAL
SEND.MAIL
SEND.TO.BACK
SERIES
SERIES.AXES

SERIES.ORDER
SERIES.X
SERIES.Y
SET.PAGE.BREAK
SET.PREFERRED
SET.PRINT.AREA
SET.PRINT.TITLES
SHORT.MENUS
SHOW.ACTIVE.CELL
SHOW.BAR
SHOW.CLIPBOARD
SHOW.DETAIL
SHOW.INFO
SHOW.LEVELS
SHOW.TOOLBAR
SIZE
SLIDE.COPY.ROW
SLIDE.CUT.ROW
SLIDE.DEFAULTS
SLIDE.DELETE.ROW
SLIDE.EDIT
SLIDE.PASTE
SLIDE.PASTE.ROW
SLIDE.SHOW
SOLVER.ADD
SOLVER.CHANGE
SOLVER.DELETE
SOLVER.FINISH
SOLVER.LOAD
SOLVER.OK
SOLVER.OPTIONS
SOLVER.RESET
SOLVER.SAVE
SOLVER.SOLVE
SORT
SOUND.NOTE
SOUND.PLAY
SPELLING
SPLIT
STANDARD.FONT
STANDARD.WIDTH
STYLE
SUBTOTAL.CREATE
SUBTOTAL.REMOVE
SUMMARY.INFO
TABLE
TEXT.BOX
TEXT.TO.COLUMNS

TRACER.CLEAR
TRACER.DISPLAY
TRACER.ERROR
TRACER.NAVIGATE
UNDO
UNGROUP
UNHIDE
UNLOCKED.NEXT
UNLOCKED.PREV
UPDATE.LINK
VBA.INSERT.FILE
VBA.MAKE.ADDIN
VIEW.3D
VIEW.DEFINE
VIEW.DELETE
VIEW.SHOW
VLINE
VPAGE
VSCROLL
WINDOW.MAXIMIZE
WINDOW.MINIMIZE
WINDOW.MOVE
WINDOW.RESTORE
WINDOW.SIZE
WORKBOOK.ACTIVATE
WORKBOOK.ADD
WORKBOOK.COPY
WORKBOOK.DELETE
WORKBOOK.HIDE
WORKBOOK.INSERT
WORKBOOK.MOVE
WORKBOOK.NAME
WORKBOOK.NEW
WORKBOOK.NEXT
WORKBOOK.OPTIONS
WORKBOOK.PREV
WORKBOOK.PROTECT
WORKBOOK.SCROLL
WORKBOOK.SELECT
WORKBOOK.TAB.SPLIT
WORKBOOK.UNHIDE
WORKGROUP
WORKSPACE
ZOOM

Control Macro Functions

ARGUMENT

BREAK

ELSE

ELSE.IF

END.IF

FOR

FOR.CELL

GOTO

HALT

IF

NEXT

PAUSE

RESTART

RESULT

RETURN

SET.NAME

SET.VALUE

STEP

VOLATILE

WAIT

WHILE

Customizing Macro Functions

ADD.BAR
ADD.CHART.AUTOFORMAT
ADD.COMMAND
ADD.MENU
ADD.TOOL
ADD.TOOLBAR
ALERT
APP.TITLE
ASSIGN.TO.TOOL
BEEP
CANCEL.KEY
CHECK.COMMAND
COPY.TOOL
CUSTOM.REPEAT
CUSTOM.UNDO
DELETE.BAR
DELETE.CHART.AUTOFORMAT
DELETE.COMMAND
DELETE.MENU
DELETE.TOOL
DELETE.TOOLBAR
DIALOG.BOX
DISABLE.INPUT
ECHO
EDIT.TOOL
ENABLE.COMMAND
ENABLE.TOOL
ENTER.DATA
ERROR
HELP
INPUT
LINK.FORMAT
MESSAGE
MOVE.TOOL
ON.DATA
ON.DOUBLECLICK
ON.ENTRY
ON.KEY
ON.RECALC
ON.TIME
ON.WINDOW
PASTE.TOOL
PRESS.TOOL
RENAME.COMMAND
RESET.TOOL
RESET.TOOLBAR

SAVE.TOOLBAR
SHOW.TOOLBAR
WINDOW.TITLE

Database & List Management Macro Functions

DATA.DELETE

DATA.FIND

DATA.FIND.NEXT

DATA.FIND.PREV

DATA.FORM

EXTRACT

SET.CRITERIA

SET.DATABASE

SET.EXTRACT

SQL.BIND

SQL.CLOSE

SQL.ERROR

SQL.EXEC.QUERY

SQL.GET.SCHEMA

SQL.OPEN

SQL.RETRIEVE

SQL.RETRIEVE.TO.FILE

DDE & External Macro Functions

APP.ACTIVATE.MICROSOFT

APP.ACTIVATE

EDIT.OBJECT

EDITION.OPTIONS

EMBED

EXEC

EXECUTE

FCLOSE

FOPEN

FPOS

FREAD

FREADLN

FSIZE

FWRITE

FWRITELN

INITIATE

INSERT.OBJECT

POKE

REGISTER

REGISTER.ID

REQUEST

SEND.KEYS

SET.UPDATE.STATUS

SUBSCRIBE.TO

TERMINATE

UNREGISTER

Engineering Macro Functions

FOURIER

SAMPLE

Information Macro Functions

ACTIVE.CELL

CALLER

DIRECTORY

DOCUMENTS

FILES

GET.BAR

GET.CELL

GET.CHART.ITEM

GET.DEF

GET.DOCUMENT

GET.FORMULA

GET.LINK.INFO

GET.NAME

GET.NOTE

GET.OBJECT

GET.PIVOT.FIELD

GET.PIVOT.ITEM

GET.PIVOT.TABLE

GET.TOOL

GET.TOOLBAR

GET.WINDOW

GET.WORKBOOK

GET.WORKSPACE

LAST.ERROR

LINKS

NAMES

REPORT.GET

SCENARIO.GET

SELECTION

SLIDE.GET

SOLVER.GET

VIEW.GET

WINDOWS

Lookup & Reference Macro Functions

ABSREF

DEREF

EVALUATE

FORMULA.CONVERT

REFTEXT

RELREF

TEXTREF

Statistical Macro Functions

ANOVA1

ANOVA2

ANOVA3

DESCR

EXPON

FTESTV

HISTOGRAM

MCORREL

MCOVAR

MOVEAVG

PTTESTM

PTTESTV

RANDOM

RANKPERC

REGRESS

TTESTM

ZTESTM

Text Macro Functions

SPELLING.CHECK

For information on the selected worksheet function , see the [Alphabetical List of Worksheet Functions](#).

New and Changed Command-Equivalent Macro Functions

ACTIVE.CELL.FONT

ADD.CHART.AUTOFORMAT

ADD.COMMAND

Adds a command to a menu. Now adds commands to the Microsoft Excel Visual Basic module shortcut menu.

ADDIN.MANAGER

ATTACH.TOOLBARS

AXES

Controls whether the axes on a chart are visible. Now has arguments for new chart format that replaces overlays on chart.

AUTO.OUTLINE

CHART.ADD.DATA

CHART.TREND

CHART.WIZARD

Formats a chart. Can now control the number of rows and columns to use for the category labels and series labels in a chart.

CLEAR.ROUTING.SLIP

CLEAR.OUTLINE

CLOSE

Closes a file. Can now allow a document to be routed via email before closing.

CUSTOMIZE.TOOLBAR

Displays the Customize Toolbar dialog box. Now displays toolbars new to Microsoft Excel 5.0.

DATA.LABEL

DELETE.CHART.AUTOFORMAT

DEMOTE

Demotes the selected rows or columns in an outline. This is now equivalent to a new tool button.

ENABLE.TIPWIZARD

ERRORBAR.X

ERRORBAR.Y

FILTER.ADVANCED

FILTER.SHOW.ALL

FILTER

FIND.FILE

FONT.PROPERTIES

FORMAT.AUTO

Formats cells from a built-in gallery of formats. Now allows additional formatting options for your spreadsheet.

FORMAT.CHART

FORMAT.CHARTTYPE

FORMAT.MOVE

Moves an object to a new location. Has new syntax for pie-chart and doughnut-chart explosions.

FUNCTION.WIZARD

GALLERY.CUSTOM

GALLERY.DOUGHNUT

GET.DOCUMENT

Returns information about a document. Has new return values.

GET.PIVOT.FIELD

GET.PIVOT.ITEM

GET.PIVOT.TABLE

INSERT.OBJECT

Creates an embedded object whose source data is supplied by another application. Now allows an application's icons to be displayed.

INSERT.TITLE

INSERT.PICTURE

MAIL.ADD.MAILER
MAIL.DELETE.MAILER
MAIL.EDIT.MAILER
MAIL.FORWARD
MAIL.LOGOFF
MAIL.LOGON
MAIL.NEXT.LETTER
MAIL.REPLY.ALL
MAIL.REPLY
MENU.EDITOR
NEW

Creates a new document. Can now create a worksheet, macrosheet, or chart in the new workbook format.

OPEN

Opens a file. Now allows a template to be open for editing, allows a special file converters to be used on the file, and sets various file sharing attributes.

OPEN.TEXT
OPTIONS.CALCULATION
OPTIONS.CHART
OPTIONS.EDIT
OPTIONS.GENERAL
OPTIONS.LISTS.ADD
OPTIONS.LISTS.DELETE
OPTIONS.VIEW
PAGE.SETUP

Controls the printed appearance of your documents. Now allows additional page setup attributes to be set.

PASTE.SPECIAL

Pastes the specified components from the copy area into the current selection. Now has additional arguments for pasting into a chart.

PATTERNS

Changes the appearance of cells and objects. Can now allow you to retain the cell and object patterns of Microsoft Excel 4.0.

PIVOT.ADD.DATA
PIVOT.ADD.FIELDS
PIVOT.FIELD.GROUP
PIVOT.FIELD.UNGROUP
PIVOT.FIELD
PIVOT.ITEM.PROPERTIES
PIVOT.ITEM
PIVOT.REFRESH
PIVOT.SHOW.PAGES
PIVOT.TABLE.WIZARD
PRINT

Prints a document. Now allows you to print specific areas on a sheet or selected sheet within the new workbook format.

PROMOTE

Promotes the selected rows or columns in an outline. This is now equivalent to a new tool button.

PROTECT.DOCUMENT

Adds or removes protection on the active document. Now allows Microsoft Excel Visual Basic modules to be protected.

QUERY.GET.DATA
QUERY.REFRESH
RENAME.OBJECT
ROUTE.DOCUMENT
ROUTING.SLIP

SAVE.AS

Saves a document with a new filename, file type, or password. Now allows a file to be saved in additional file formats.

SAVE.COPY.AS

SCENARIO.ADD

SCENARIO.EDIT

SCENARIO.SUMMARY

Summarizes the results of scenarios. Now allows you to specify different types or reports.

SELECT.ALL

SELECT

Selects cells or objects. Now allows trend bars, error bars, and other chart items to be selected.

SERIES.AXES

SERIES.ORDER

SERIES.X

SERIES.Y

SET.PRINT.AREA

Defines the print area. Now allows you to specify a range to be set for the print area.

SHOW.DETAIL

Expands or collapses the detail under the specified Group or Ungroup buttons. Can now work with PivotTables.

SORT

Sorts the rows or columns of a selection. Can now recognize headers on lists. Will also allow custom sorting, sorting within PivotTables, and case-sensitive sorting.

SUBTOTAL.CREATE

SUBTOTAL.REMOVE

SUMMARY.INFO

TEXT.TO.COLUMNS

TRACER.CLEAR

TRACER.DISPLAY

TRACER.ERROR

TRACER.NAVIGATE

VBA.INSERT.FILE

VBA.MAKE.ADDIN

VIEW.DEFINE

VIEW.DELETE

VIEW.SHOW

WORKBOOK.ACTIVATE

Activates a document in a workbook. Additional arguments have been added for new workbook format.

WORKBOOK.DELETE

WORKBOOK.HIDE

WORKBOOK.INSERT

WORKBOOK.NAME

WORKBOOK.NEW

WORKBOOK.NEXT

WORKBOOK.PREV

WORKBOOK.PROTECT

WORKBOOK.SCROLL

WORKBOOK.SELECT

Selects documents in a workbook. Additional arguments have been added to allow worksheets to be replaced by other worksheets.

WORKBOOK.TAB.SPLIT

WORKBOOK.UNHIDE

New and Changed Customizing Macro Functions

<u>ADD.COMMAND</u>	Adds a command to a menu. New shortcut commands have been added.
<u>ADD.MENU</u>	Adds a menu to a menu bar. Now you can add submenus.
<u>ADD.TOOL</u>	Adds one or more tools to a toolbar. Can now add toolbars for Microsoft applications, Microsoft Excel Visual Basic, workgroups, and worksheet auditing.
<u>CHECK.COMMAND</u>	Adds or removes a check mark to and from a command name or menu. Now works with submenus.
<u>DELETE.COMMAND</u>	Deletes a command from a custom or built-in menu. Can now delete commands from submenus.
<u>DELETE.MENU</u>	Deletes a menu from a menu bar. Can now delete submenus.
<u>EDIT.TOOL</u>	
<u>ENABLE.COMMAND</u>	Enables or disables a custom command or menus. Now works with commands on submenus.
<u>LINK.FORMAT</u>	
<u>RENAME.COMMAND</u>	Changes the name of a built-in or custom menu command or the name of a menu. Now works with submenus.
<u>SHOW.BAR</u>	Displays the specified menu bar. Will now display menu bars for Microsoft Visual basic modules, dialog sheets, and macro sheets.

New and Changed Database & List Management Macro Functions

DATA.FORM

Displays the data form. Has new information on using the macro command with worksheets created in Microsoft Excel 4.0.

SQL.BIND

SQL.CLOSE

SQL.ERROR

SQL.EXEC.QUERY

SQL.GET.SCHEMA

SQL.OPEN

SQL.RETRIEVE

SQL.RETRIEVE.TO.FILE

New and Changed DDE/External Macro Functions

APP.ACTIVATE.MICROSOFT

EDIT.OBJECT

Starts an application associated with the selected object and makes the object available for editing. Now allows a macro to be paused while the other application is editing the object.

INSERT.OBJECT

Creates an embedded object. Can now display an application's icon.

New and Changed Information Macro Functions

<u>GET.BAR</u>	Returns the name and position number of a specified command on a menu. Has new return values.
<u>GET.CELL</u>	Returns information about a cell. Has new return values.
<u>GET.DOCUMENT</u>	Returns information about a sheet in a workbook. Has new return values.
<u>GET.OBJECT</u>	Returns information about the specified object. Has new return values.
<u>GET.PIVOT.FIELD</u>	Returns information about a field in a PivotTable.
<u>GET.PIVOT.ITEM</u>	Returns information about an item in a PivotTable.
<u>GET.PIVOT.TABLE</u>	Returns information about a PivotTable.
<u>GET.TOOL</u>	Returns information about a button or buttons on a toolbar. Has new return values.
<u>GET.TOOLBAR</u>	Returns information about a toolbar or all toolbars. Has new return values.
<u>GET.WINDOW</u>	Returns information about a window. Has new return values.
<u>GET.WORKBOOK</u>	Returns information about a workbook. Has new return values.
<u>GET.WORKSPACE</u>	Returns information about a workspace. Has new return values.
<u>OPTIONS.LISTS.GET</u>	
<u>SCENARIO.GET</u>	Returns information about scenarios defined on a workbook. Has new return values.

New and Changed Statistical Macro Functions

<u>ANOVA1</u>	Performs single-factor analysis of variance. Can now place output in a new workbook.
<u>ANOVA2</u>	Performs two-factor analysis of variance with replication. Can now place output in a new workbook.
<u>ANOVA3</u>	Performs two-factor analysis of variance without replication. Can now place output in a new workbook.
<u>DESCR</u>	Generates descriptive statistics for data. Can now place output in a new workbook.
<u>EXPON</u>	Predicts a value based on the forecast for the prior period. Can now place output in a new workbook.
<u>FOURIER</u>	Performs a Fourier transform. Can now place output in a new workbook.
<u>FTESTV</u>	Performs a two-sample F-test. Can now place output in a new workbook.
<u>HISTOGRAM</u>	Calculates histograms. Can now place output in a new workbook.
<u>MCORREL</u>	Measures the correlation between two or more data sets. Can now place output in a new workbook.
<u>MCOVAR</u>	Measures the covariance between two or more data sets. Can now place output in a new workbook.
<u>MOVEAVG</u>	Projects values in a forecast period. Can now place output in a new workbook.
<u>PTTESTM</u>	Performs a paired two-sample Student's t-Test for means. Can now place output in a new workbook.
<u>PTTESTV</u>	Performs a two-sample Student's t-Test, assuming unequal variances. Can now place output in a new workbook.
<u>RANDOM</u>	Fills a range with independent random or patterned numbers. Can now place output in a new workbook.
<u>RANKPERC</u>	Returns the ordinal and percent rank of each value in a data set. Can now place output in a new workbook.
<u>REGRESS</u>	Performs multiple linear regression analysis. Can now place output in a new workbook.
<u>SAMPLE</u>	Samples data. Can now place output in a new workbook.
<u>TTESTM</u>	Performs a two-sample Student's t-Test for means, assuming equal variances. Can now place output in a new workbook.
<u>ZTESTM</u>	Performs a two-sample z-test for means. Can now place output in a new workbook.

New and Changed Text Macro Functions

OPEN.DIALOG

Displays the standard Microsoft Excel File Open dialog with the specified file filters.

SAVE.DIALOG

Displays the standard Microsoft Excel File Save dialog with the specified file filters.

ON Functions

ON functions allow you to specify a macro to be run when a certain event occurs. The ON functions turn on and off this special event handling.

ON functions are turned on by specifying the type of event to wait for, such as recalculation, a specific time, or a key to be pressed, and the macro to be run when the event occurs. ON functions are turned off by using the same formula but omitting the argument specifying the macro to be run.

<u>ON.DATA</u>	Runs a macro when data is entered
<u>ON.DOUBLECLICK</u>	Runs a macro when you double-click any cell or object on the specified document or double-click any item on the specified chart
<u>ON.ENTRY</u>	Runs a macro when a document is recalculated
<u>ON.KEY</u>	Runs a macro when a specified key is pressed
<u>ON.RECALC</u>	Runs a macro when a document is recalculated
<u>ON.SHEET</u>	Runs a macro when a specified sheet is selected
<u>ON.TIME</u>	Runs a macro at a specific time
<u>ON.WINDOW</u>	Runs a macro when you switch to a window

A1.R1C1

Macro Sheets Only

Displays row and column headings and cell references in either the R1C1 or A1 reference style. A1 is the Microsoft Excel default reference style.

Syntax

A1.R1C1(logical)

Logical is a logical value specifying which reference style to use. If logical is TRUE, all worksheets and macro sheets use A1 references; if FALSE, all worksheets and macro sheets use R1C1 references.

Example

The following macro formula displays an alert box asking you to choose either A1 or R1C1 reference style. This is useful in an Auto_Open macro if several persons who prefer different reference styles must maintain the same worksheet.

```
A1.R1C1(ALERT("Click OK for A1 style; Cancel for R1C1", 1))
```

Related Functions

List of [Command-Equivalent Functions](#)

ACTIVE.CELL

Macro Sheets Only

Returns the reference of the active cell in the selection as an external reference.

Syntax

ACTIVE.CELL()

Remarks

- If an object is selected, ACTIVE.CELL returns the #N/A error value.
- If a chart window is active, ACTIVE.CELL returns the #REF! error value.
- If you use ACTIVE.CELL in a function or operation, you will usually get the value contained in the active cell instead of its reference, because the reference is automatically converted to the contents of the reference. See the third example following.
- If you use ACTIVE.CELL in a function that requires a reference argument, then Microsoft Excel does not convert the reference to a value.
- If you want to work with the actual reference, use the REFTEXT function to convert the active-cell reference to text, which you can then store or manipulate (or convert back to a reference with TEXTREF). See the second example following.

Tip Use the following macro formula to verify that the current selection is a cell or range of cells:

`=ISREF(ACTIVE.CELL())`

Examples

The following macro formula assigns the name Sales to the active cell:

`SET.NAME("Sales", ACTIVE.CELL())`

In this example, note that "Sales" refers to a cell on the active worksheet, but the name itself exists only in the macro sheet's list of names. In other words, the preceding formula does not define a name on the worksheet.

The following macro formula puts the reference of the active cell into the cell named Temp:

`FORMULA(""&REFTEXT(ACTIVE.CELL()), Temp)`

The following macro formula checks the contents of the active cell. If the cell contains only the letter "c" or "s", the macro branches to an area named FinishRefresh:

`IF(OR(ACTIVE.CELL()="c", ACTIVE.CELL()="s"), GOTO(FinishRefresh))`

In Microsoft Excel for Windows, if the document in the active window is named SALES and A1 is the active cell, then:

`ACTIVE.CELL()` equals `SALES!A1`

In Microsoft Excel for the Macintosh, if the document in the active window is named SALES 1 and A1 is the active cell, then:

`ACTIVE.CELL()` equals `'SALES 1'!A1`

Related Functions

OFFSET Returns a reference offset from a given reference

SELECT Selects a cell, worksheet object, or chart item

List of Information Functions

ALERT

Macro Sheets Only

Displays a dialog box and message and waits for you to choose a button. Use ALERT instead of MESSAGE if you want to interrupt the flow of a macro and force the user to make a choice or to notice an important message.

Syntax

ALERT(message_text, type_num, help_ref)

Message_text is the message displayed in the dialog box.

Type_num is a number from 1 to 3 specifying which type of dialog box to display. If you omit type_num, it is assumed to be 2.

- If type_num is 1, ALERT displays a dialog box containing the OK and Cancel buttons. Choose a button to continue or cancel an action. ALERT returns TRUE if you choose the OK button and FALSE if you choose the Cancel button. See the last example below.
- If type_num is 2 or 3, ALERT displays a dialog box containing an OK button. Choose the button to continue, and ALERT returns TRUE. The only difference between specifying 2 or 3 is that ALERT displays a different icon on the left side of the dialog box as shown in the examples below. So, for example, you could use 2 for notes or to present general information, and 3 for errors or warnings.

Help_ref is a reference to a custom online Help topic, in the form "filename! topic_number".

- If help_ref is present, a Help button appears in the lower-right corner of the alert message. Choosing the Help button starts Help and displays the specified topic.
- If help_ref is omitted, no Help button appears.
- Help_ref must be given in text form.

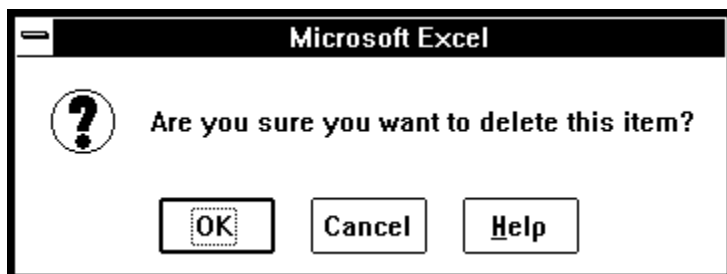
Note In Microsoft Excel for the Macintosh, the ALERT dialog box is not a movable window.

Examples

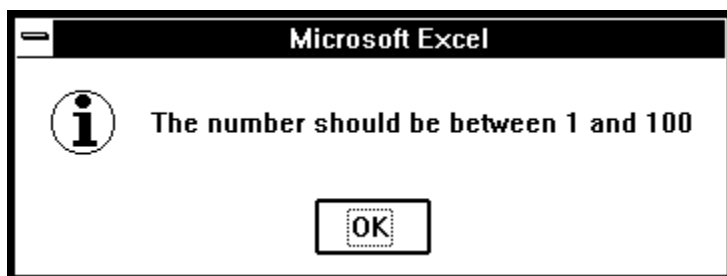
The following dialog boxes show the results of using ALERT with type_num 1, 2, and 3. The first and fourth examples include a Help button.

In Microsoft Excel for Windows, the following macro formulas display these three dialog boxes.

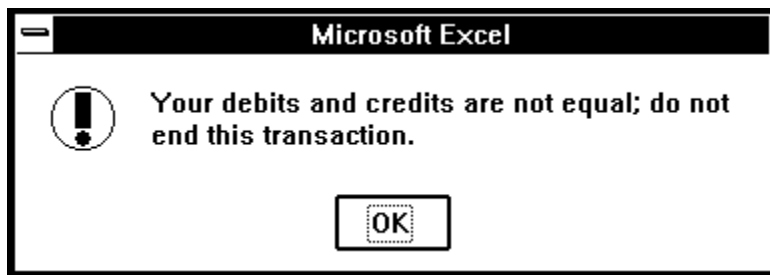
```
ALERT("Are you sure you want to delete this item?", 1, "CUSTHELP.HLP!101")
```



```
ALERT("The number should be between 1 and 100", 2)
```

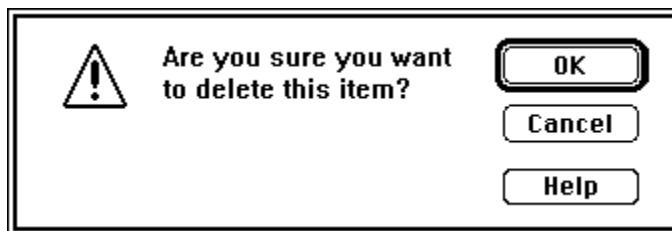


```
ALERT("Your debits and credits are not equal; do not end this transaction.", 3)
```

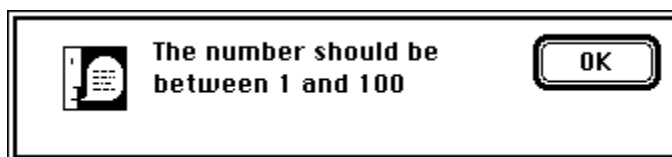


In Microsoft Excel for the Macintosh, the following macro formulas display these three dialog boxes.

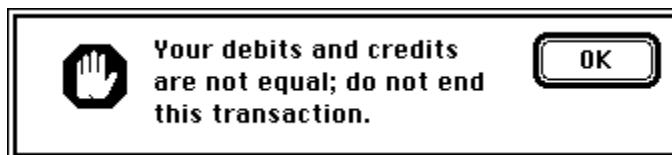
```
ALERT("Are you sure you want to delete this item?", 1, "'Custom Help'!101")
```



```
ALERT("The number should be between 1 and 100", 2)
```



```
ALERT("Your debits and credits are not equal; do not end this transaction.", 3)
```



A common use of the ALERT function is to give the user a choice of two actions. The following macro formula in an Auto_Open macro asks which reference style to use when the worksheet is opened.

```
A1.R1C1(ALERT("Click OK for A1 style; Cancel for R1C1", 1))
```

Related Functions

- INPUT Displays a dialog box for user input
- MESSAGE Displays a message in the status bar
- List of Customizing Functions

ARGUMENT

Macro Sheets Only

Describes the arguments used in a custom function, which is a type of macro, or in a subroutine. A custom function or subroutine must contain one ARGUMENT function for each argument in the macro itself. There are two forms of the ARGUMENT function. In the first form, only name_text is required; in the second form, only reference is required. Use the first form if you want to store the argument as a name. Use the second form if you want to store the argument in a specific cell or cells.

Syntax 1

For name storage

ARGUMENT(name_text, data_type_num)

Syntax 2

For cell storage

ARGUMENT(name_text, data_type_num, reference)

Name_text is the name of the argument or of the cells containing the argument. Name_text is required if you omit reference.

Data_type_num is a number that determines what type of values Microsoft Excel accepts for the argument. The following table lists the possible data types.

Data_type_num	Type of value
1	Number
2	Text
4	Logical
8	Reference
16	Error
64	Array

- Data_type_num can be a sum of the preceding different numbers to allow for more than one possible type of data. For example, if data_type_num is 7, which is the sum of 1, 2, and 4, then the value can be a number, text, or logical value.
- Data_type_num is an optional argument. If you omit data_type_num, it is assumed to be 7.
- If the value that is passed to the function macro is not of the type specified by data_type_num, Microsoft Excel first attempts to convert it to the specified type. If the value cannot be converted, the macro returns the #VALUE! error value.

Reference is the cell or cells in which you want to store the argument's value.

- If you specify reference, the value that is passed to ARGUMENT is entered as a constant in the specified cell, and name_text becomes an optional argument because you can refer to the cell with either reference or name_text.
- If you omit reference, name_text is defined on the macro sheet and refers to the value that is passed to ARGUMENT. Once name_text is defined, you can use it in formulas.

Remarks

- Custom functions and subroutines can accept from 1 to 29 arguments.
- If a macro contains an ARGUMENT function and you omit the corresponding argument in the function that starts the macro, the macro uses the #N/A error value as the value of the argument.

Examples

To create a custom function that calculates profit, use the following functions to specify arguments for cost, sales, and sales volume:

```
ARGUMENT("UnitsSold", 1)
```

```
ARGUMENT("UnitCost", 1)
```

```
ARGUMENT("UnitPrice", 1)
```

Related Function

<u>RESULT</u>	Specifies the data type a custom function returns
<u>VOLATILE</u>	Makes custom functions recalculate automatically

List of Control Functions

BEEP

Macro Sheets Only

Sounds a tone. Use BEEP to signal a message, a dialog box, or the end of a macro, or whenever you need to get the user's attention.

Syntax

BEEP(tone_num)

Tone_num is a number from 1 to 4 specifying the tone to be played.

- On most computers, all numbers produce the same sound, the sound that you hear when an error occurs or when you click outside some dialog boxes.
- If tone_num is omitted, it is assumed to be 1.

Remarks

- With a Macintosh, you can control the volume of the tone by using the Control Panels desk accessory.
- With Microsoft Windows version 3.0 or later, you can turn off the tone by using the Control Panel.

Related Functions

ALERT Displays a dialog box and a message

MESSAGE Displays a message in the status bar

List of Customizing Functions

BREAK

Macro Sheets Only

Interrupts a FOR-NEXT, a FOR.CELL-NEXT, or a WHILE-NEXT loop. If BREAK is encountered within a loop, that loop is terminated and the macro proceeds to the statement following the NEXT statement at the end of the current loop.

Syntax

BREAK()

Example

Use BREAK to test for conditions not anticipated by the FOR or WHILE statement. For example, use the BREAK nested in an IF statement to exit a WHILE-NEXT loop when a certain value is encountered:

```
=IF(Counter=8, BREAK())
```

Related Functions

<u>FOR</u>	Starts a FOR-NEXT loop
<u>FOR.CELL</u>	Starts a FOR.CELL-NEXT loop
<u>NEXT</u>	Ends a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
<u>WHILE</u>	Starts a WHILE-NEXT loop

List of Control Functions

DUPLICATE

Macro Sheets Only

Duplicates the selected object. If an object is not selected, returns the #VALUE! error value and interrupts the macro.

Syntax

DUPLICATE()

Related Functions

COPY Copies and pastes data or objects

PASTE Pastes cut or copied data

List of Command-Equivalent Functions

ECHO

Macro Sheets Only

Controls screen updating while a macro is running. If a large macro uses many commands that update the screen, use ECHO to make the macro run faster.

Syntax

ECHO(logical)

Logical is a logical value specifying whether screen updating is on or off.

- If logical is TRUE, Microsoft Excel selects screen updating.
- If logical is FALSE, Microsoft Excel clears screen updating.
- If logical is omitted, Microsoft Excel changes the current screen update condition.

Remarks

- Screen updating is always turned back on when a macro ends.
- You can use GET.WORKSPACE to determine whether screen updating is on or off.

Related Functions

[GET.WORKSPACE](#)

Returns information about the workspace

List of [Customizing Functions](#)

ERROR

Macro Sheets Only

Specifies what action to take if an error is encountered while a macro is running. Use ERROR to control whether Microsoft Excel error messages are displayed, or to run your own macro when an error is encountered.

Syntax

ERROR(enable_logical, macro_ref)

Enable_logical is a logical value or number that selects or clears error-checking.

- If enable_logical is FALSE or 0, all error-checking is cleared. If error-checking is cleared and an error is encountered while a macro is running, Microsoft Excel ignores it and continues. Error-checking is selected again by an ERROR(TRUE) statement, or when the macro stops running.
- If enable_logical is TRUE or 1, you can either select normal error-checking (by omitting the other argument) or specify a macro to run when an error is encountered by using the macro_ref argument. When normal error-checking is active, the Macro Error dialog box is displayed when an error is encountered. You can choose to halt the macro, start single-stepping through the macro, continue running the macro normally, or go to the macro cell where the error occurred.
- If enable_logical is 2 and macro_ref is omitted, error-checking is normal except that if the user chooses the Cancel button in an alert message, ERROR returns FALSE and the macro is not interrupted.
- If enable_logical is 2 and macro_ref is given, the macro goes to that macro_ref when an error is encountered. If the user chooses the Cancel button in an alert message, FALSE is returned and the macro is not interrupted.

Macro_ref specifies a macro to run if enable_logical is TRUE, 1, or 2 and an error is encountered. It can be either the name of the macro or a cell reference. If enable_logical is FALSE or 0, macro_ref is ignored.

Important Both ERROR(FALSE) and ERROR(TRUE, macro_ref) keep Microsoft Excel from displaying any messages at all, including the message asking whether to save changes when you close an unsaved document. If you want alert messages but not error messages to be displayed, use ERROR(2, macro_ref).

Remarks

You can use GET.WORKSPACE to determine whether error-checking is on or off.

Examples

ERROR (FALSE) clears error-checking.

ERROR(TRUE, Recover) selects error-checking and runs the macro named Recover when an error is encountered.

The following macro runs the macro ForceMenus if an error occurs in the current macro:

=ERROR(TRUE, ForceMenus)

Related Functions

<u>CANCEL.KEY</u>	Disables macro interruption
<u>LAST.ERROR</u>	Returns the reference of the cell where the last error occurred
<u>ON.KEY</u>	Runs a macro when a specified key is pressed

List of Customizing Functions

FCLOSE

Macro Sheets Only

Closes the specified file.

Syntax

FCLOSE(file_num)

File_num is the number of the file you want to close. File_num is returned by the FOPEN function that originally opened the file. If file_num is not a valid file number, FCLOSE halts the macro and returns the #VALUE! error value.

Examples

The following function closes the file identified by FileNumber:

```
FCLOSE(FileNumber)
```

Related Functions

<u>CLOSE</u>	Closes the active window
<u>FILE.CLOSE</u>	Closes the active document
<u>FOPEN</u>	Opens a file with the type of permission specified
List of <u>DDE/External Functions</u>	

FONT

Macro Sheets Only

Equivalent to choosing the Font command from the Options menu in Microsoft Excel for the Macintosh version 1.5 or earlier. This function is included only for macro compatibility. Sets the font for the Normal style. Microsoft Excel now uses the FONT.PROPERTIES and DEFINE.STYLE functions. For more information, see FONT.PROPERTIES and DEFINE.STYLE.

Syntax

FONT(name_text, size_num)

FONT?(name_text, size_num)

Related Functions

DEFINE.STYLE Creates or changes a cell style

FONT.PROPERTIES Sets various font properties

List of Command-Equivalent Functions

FOPEN

Macro Sheets Only

Opens a file with the type of permission specified. Unlike OPEN, FOPEN does not load the file into memory and display it; instead, FOPEN establishes a channel with the file so that you can exchange information with it. If the file is opened successfully, FOPEN returns a file ID number. If it can't open the file, FOPEN returns the #N/A error value. Use the file ID number with other file functions (such as FREAD, FWRITE, and FSIZE) when you want to get information from or send information to the file.

Syntax

FOPEN(file_text, access_num)

File_text is the name of the file you want to open.

Access_num is a number from 1 to 3 specifying what type of permission to allow to the file:

Access_num	Type of permission
1 or omitted	Can read and write to the file (read/write permission)
2	Can read the file, but can't write to the file (read-only permission)
3	Creates a new file with read/write permission

- If the file doesn't exist and access_num is 3, FOPEN creates a new file.
- If the file does exist and access_num is 3, FOPEN replaces the contents of the file with any information you supply using the FWRITE or FWRITELN functions.
- If the file doesn't exist and access_num is 1 or 2, FOPEN returns the #N/A error value.

Remarks

Use FCLOSE to close a file after you finish using it.

Example

The following function opens a file identified as FileName using read-only permission:

```
FOPEN(FileName, 2)
```

Related Functions

<u>FCLOSE</u>	Closes a text file
<u>FREAD</u>	Reads characters from a text file
<u>FWRITE</u>	Writes characters to a text file
<u>OPEN</u>	Opens a document
List of <u>DDE/External Functions</u>	

FOR

Macro Sheets Only

Starts a FOR-NEXT loop. The instructions between FOR and NEXT are repeated until the loop counter reaches a specified value. Use FOR when you need to repeat instructions a specified number of times. Use FOR.CELL when you need to repeat instructions over a range of cells.

Syntax

FOR(counter_text, start_num, end_num, step_num)

Counter_text is the name of the loop counter in the form of text.

Start_num is the value initially assigned to counter_text.

End_num is the last value assigned to counter_text.

Step_num is a value added to the loop counter after each iteration. If step_num is omitted, it is assumed to be 1.

Remarks

- Microsoft Excel follows these steps as it executes a FOR-NEXT loop:

Step	Action
1	Sets counter_text to the value start_num.
2	If counter_text is greater than end_num (or less than end_num if step_num is negative), the loop ends, and the macro continues with the function after the NEXT function. If counter_text is less than or equal to end_num (or greater than or equal to end_num if step_num is negative), the macro continues in the loop.
3	Carries out functions up to the following NEXT function. The NEXT function must be below the FOR function and in the same column.
4	Adds step_num to the loop counter.
5	Returns to the FOR function and proceeds as described in step 2.

- You can interrupt a FOR-NEXT loop by using the BREAK function.

Example

The following macro starts a FOR-NEXT loop that is executed once for every open window:

```
FOR("Counter", 1, COLUMNS(WINDOWS()))
```

Related Functions

<u>BREAK</u>	Interrupts a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
<u>FOR.CELL</u>	Starts a FOR.CELL-NEXT loop
<u>NEXT</u>	Ends a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
<u>WHILE</u>	Starts a WHILE-NEXT loop
List of <u>Control Functions</u>	

FOR.CELL

Macro Sheets Only

Starts a FOR.CELL-NEXT loop. This function is similar to FOR, except that the instructions between FOR.CELL and NEXT are repeated over a range of cells, one cell at a time, and there is no loop counter.

Syntax

FOR.CELL(ref_name, area_ref, skip_blanks)

Ref_name is the name in the form of text that Microsoft Excel gives to the one cell in the range that is currently being operated on; ref_name refers to a new cell during each loop.

Area_ref is the range of cells on which you want the FOR.CELL-NEXT loop to operate and can be a multiple selection. If area_ref is omitted, it is assumed to be the current selection.

Skip_blanks is a logical value specifying whether Microsoft Excel skips blank cells as it operates on the cells in area_ref.

Skip_blanks	Result
TRUE	Skips blank cells in area_ref
FALSE or omitted	Operates on all cells in area_ref

Remarks

FOR.CELL operates on each cell in a row from left to right one area at a time before moving to the next row in the selection.

Example

The following macro starts a FOR.CELL-NEXT loop and uses the name CurrentCell to refer to the cell in the range that is currently being operated on:

```
FOR.CELL("CurrentCell", SELECTION(), TRUE)
```

Related Functions

<u>BREAK</u>	Interrupts a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
<u>FOR</u>	Starts a FOR-NEXT loop
<u>NEXT</u>	Ends a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
<u>WHILE</u>	Starts a WHILE-NEXT loop

List of [Control Functions](#)

FPOS

Macro Sheets Only

Sets the position of a file. The position of a file is where a character is read from or written to by an FREAD, FREADLN, FWRITE, or FWRITELN function. Use FPOS when you want to write characters to or read characters from specific locations. For example, to append text to the end of a file, you must set the position to the end of the file; otherwise, you might accidentally overwrite existing characters in the file.

Syntax

FPOS(file_num, position_num)

File_num is the unique ID number of the file for which you want to set the position. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FPOS returns the #VALUE! error value.

Position_num is the location in the file that a character will be read from or written to.

- The first position in a file is 1, the location of the first byte.
- The last position in the file is the same as the value returned by FSIZE. For example, the last position in a file with 280 bytes is 280.
- If position_num is omitted, FPOS returns the current position of the file that is, the number corresponding to where the next character will be read from or written to.

Whenever you read a character from or write a character to a file, the file's position is automatically incremented.

Examples

The following statement starts a loop that executes until the position in the open file identified as FileNumber reaches the end of the file:

```
=WHILE (FPOS (FileNumber) <=FSIZE (FileNumber) )
```

Related Functions

<u>FCLOSE</u>	Closes a text file
<u>FOPEN</u>	Opens a file with the type of permission specified
<u>FREAD</u>	Reads characters from a text file
<u>FREADLN</u>	Reads a line from a text file
<u>FWRITE</u>	Writes characters to a text file
<u>FWRITELN</u>	Writes a line to a text file

List of DDE/External Functions

FREAD

Macro Sheets Only

Reads characters from a file, starting at the current position in the file. (For more information about a file's position, see FPOS.) If FREAD is successful, it returns the text to the cell containing FREAD and sets the file's position to the start of the following line. If the end of the file is reached or if FREAD can't read the file, it returns the #N/A error value. Use FREAD instead of FREADLN when you need to read a specific number of characters from a text file.

Syntax

FREAD(file_num, num_chars)

File_num is the unique ID number of the file you want to read data from. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FREAD returns the #VALUE! error value.

Num_chars specifies how many bytes to read from the file. FREAD can read up to 255 bytes at a time.

Example

The following function reads the next 200 bytes from the open file identified as FileNumber:

```
FREAD(FileNumber, 200)
```

Related Functions

<u>FOPEN</u>	Opens a file with the type of permission specified
<u>FPOS</u>	Sets the position in a text file
<u>FREADLN</u>	Reads a line from a text file
<u>FWRITE</u>	Writes characters to a text file

List of DDE/External Functions

FREADLN

Macro Sheets Only

Reads characters from a file, starting at the current position in the file and continuing to the end of the line, placing the characters in the cell containing FREADLN. (For more information about a file's position, see FPOS.) If FREADLN is successful, it returns the text it read, up to but not including the carriage-return and linefeed characters at the end of the line (in Microsoft Excel for Windows) or the carriage-return character at the end of the line (in Microsoft Excel for the Macintosh). If the current file position is the end of the file or if FREADLN can't read the file, it returns the #N/A error value.

Syntax

FREADLN(file_num)

File_num is the unique ID number of the file you want to read data from. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FREADLN returns the #VALUE! error value.

Example

The following function reads the next line from the open file identified as FileNumber:

```
FREADLN(FileNumber)
```

Related Functions

<u>FOPEN</u>	Opens a file with the type of permission specified
<u>FPOS</u>	Sets the position in a text file
<u>FREAD</u>	Reads characters from a text file
<u>FWRITE</u>	Writes characters to a text file
<u>FWRITELN</u>	Writes a line to a text file

List of DDE/External Functions

FSIZE

Macro Sheets Only

Returns the number of bytes in a file. Use FSIZE to determine the size of the file, which is the same as the position of the last byte in the file.

Syntax

FSIZE(file_num)

File_num is the unique ID number of the file whose size you want to know. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FSIZE returns the #VALUE! error value.

Example

The following function returns the size in bytes of the open file identified as FileNumber:

```
FSIZE(FileNumber)
```

Related Functions

FOPEN Opens a file with the type of permission specified

FPOS Sets the position in a text file

List of DDE/External Functions

FWRITE

Macro Sheets Only

Writes text to a file, starting at the current position in that file. (For more information about a file's position, see FPOS.) If FWRITE can't write to the file, it returns the #N/A error value.

Syntax

FWRITE(file_num, text)

File_num is the unique ID number of the file you want to write data to. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FWRITE returns the #VALUE! error value.

Text is the text you want to write to the file.

Example

The following function writes the current month to the open file identified as FileNumber:

```
FWRITE(FileNumber, TEXT(MONTH(NOW()), "mmm"))
```

Related Functions

FOPEN Opens a file with the type of permission specified

FPOS Sets the position in a text file

FREAD Reads characters from a text file

FWRITELN Writes a line to a text file

List of DDE/External Functions

FWRITELN

Macro Sheets Only

Writes text, followed by a carriage return and linefeed, to a file, starting at the current position in that file. (For more information about a file's position, see FPOS.) If FWRITELN can't write to the file, it returns the #N/A error value. Use FWRITELN instead of FWRITE when you want to append a carriage return and linefeed to each group of characters that you write to a text file.

Syntax

FWRITELN(file_num, text)

File_num is the unique ID number of the file you want to write data to. File_num is returned by a previously executed FOPEN function. If file_num is not valid, FWRITELN returns the #VALUE! error value.

Text is the text you want to write to the file.

Remarks

In Microsoft Excel for Windows, FWRITELN writes text followed by a carriage return and a line feed. In Microsoft Excel for the Macintosh, FWRITELN writes text followed by a carriage return only.

Example

The following function writes the current month to the open file identified as FileNumber and starts a new line in the file:

```
FWRITELN(FileNumber, TEXT(MONTH(NOW()), "mmm"))
```

Related Functions

<u>FOPEN</u>	Opens a file with the type of permission specified
<u>FPOS</u>	Sets the position in a text file
<u>FREAD</u>	Reads characters from a text file
<u>FWRITE</u>	Writes characters to a text file
List of <u>DDE/External Functions</u>	

GOTO

Macro Sheets Only

Directs a macro to continue running at the upper-left cell of reference. Use GOTO to direct macro execution to another cell or a named range.

Syntax

GOTO(reference)

Reference is a cell reference or a name that is defined as a reference. Reference can be an external reference to another macro sheet. If that macro sheet is not open, GOTO displays a message.

Tip It's often preferable to use IF, ELSE, ELSE.IF, and END.IF instead of GOTO when you want to perform multiple actions based on a condition because the IF method makes your macros more structured.

Examples

If A1 contains the #N/A error value, then when the following formula is calculated, the macro branches to C3:

```
IF(ISERROR($A$1), GOTO($C$3), )
```

You can also use macro names with GOTO statements. The following macro formula branches macro execution to a macro named Compile:

```
GOTO(Compile)
```

Because Compile is a named range, it should not be enclosed in quotation marks.

Related Function

[FORMULA.GOTO](#)

Selects a named area or reference on any open document

List of [Control Functions](#)

HALT

Macro Sheets Only

Stops all macros from running. Use HALT instead of RETURN to prevent a macro from returning to the macro that called it.

Syntax

HALT(cancel_close)

Cancel_close is a logical value that specifies whether a macro sheet, when encountering the HALT function in an Auto_Close macro, is closed.

- If cancel_close is TRUE, Microsoft Excel halts the macro and prevents the document from being closed.
- If cancel_close is FALSE or omitted, Microsoft Excel halts the macro and allows the document to be closed.
- If cancel_close is specified in a macro that is not an Auto_Close macro, it is ignored and the HALT function simply stops the current macro.

Remarks

You can prevent an Auto_Close or Auto_Open macro from running by holding down the SHIFT key while opening or closing the document.

Examples

If A1 contains the #N/A error value, then when the following macro formula is calculated, the macro halts:

```
IF (ISERROR (A1) , HALT ( ) , GOTO (D4) )
```

The following macro formula at the end of an Auto_Close macro ends the macro and prevents the document from being closed:

```
HALT (TRUE)
```

Related Functions

[BREAK](#) Interrupts a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop

[RETURN](#) Ends the currently running macro

List of [Control Functions](#)

HLINE

Macro Sheets Only

Scrolls through the active window by a specific number of columns. Returns the #VALUE! error value if the active document is a chart.

Syntax

HLINE(num_columns)

Num_columns is the number of columns in the active worksheet or macro sheet you want to scroll through horizontally.

- If num_columns is positive, HLINE scrolls to the right.
- If num_columns is negative, HLINE scrolls to the left.
- Num_columns must be between -256 and 256, inclusive.

Example

The following function scrolls the active window by one-half window to the right:

```
HLINE(GET.WINDOW(15)/2)
```

Related Functions

<u>HPAGE</u>	Horizontally scrolls through the active window one window at a time
<u>HSCROLL</u>	Horizontally scrolls through a document by percentage or by column number
<u>VLIN</u>	Vertically scrolls through the active window by rows
<u>VPAGE</u>	Vertically scrolls through the active window one window at a time
<u>VSCROLL</u>	Vertically scrolls through a document by percentage or by row number
List of <u>Command-Equivalent Functions</u>	

LIST.NAMES

Macro Sheets Only

Equivalent to choosing the Paste command on the Name submenu on the Insert menu and selecting the Paste List option button. Lists all names (except hidden names) defined on your worksheet.

LIST.NAMES also lists the cells to which the names refer; whether a macro corresponding to a particular name is a command macro or a custom function; the shortcut key for each command macro; and the category of the custom functions.

Syntax

LIST.NAMES()

Remarks

- If the current selection is a single cell or five or more columns wide, LIST.NAMES pastes all five types of information about worksheet names into five columns. The first column contains cell names. The second column contains the corresponding cell references. The third column contains the number 1 if the name refers to a custom function, the number 2 if it refers to a command macro, or 0 if it refers to anything else. The fourth column lists the shortcut keys for command macros. The fifth column contains a category name for custom functions or the number of the built-in category.
- If the selection includes fewer than five columns, LIST.NAMES omits the information that would have been pasted into the other columns.
- When you use LIST.NAMES, Microsoft Excel completely replaces the contents of the cells it pastes into.

Related Functions

GET.DEF Returns a name matching a definition

GET.NAME Returns the definition of a name

NAMES Returns the names defined in a document

List of Command-Equivalent Functions

NEXT

Macro Sheets Only

Ends a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop and continues carrying out the current macro with the formula that follows the NEXT function.

Syntax

NEXT()

Related Functions

<u>FOR</u>	Starts a FOR-NEXT loop
<u>FOR.CELL</u>	Starts a FOR.CELL-NEXT loop
<u>WHILE</u>	Starts a WHILE-NEXT loop
List of <u>Control Functions</u>	

PAUSE

Macro Sheets Only

Pauses a macro. Use the PAUSE function, instead of choosing the Pause button in the Single Step dialog box, as a debugging tool when you do not wish to step through a macro. You can also use PAUSE to enter and edit data, to work directly with Microsoft Excel commands, or to perform other actions that are not normally available when a macro is running.

Syntax

PAUSE(no_tool)

No_tool is a logical value specifying whether to display the Resume Macro button when the macro is paused. If no_tool is TRUE, the toolbar is not displayed; if FALSE, the toolbar is displayed; if omitted, the toolbar is displayed unless you previously clicked the close box on the toolbar.

Remarks

- All commands and tools that are available when no macro is running are still available when a macro is paused.
- You can run other macros while a macro is paused, but you can pause only one macro at a time. If a macro is paused when you run a second macro containing a PAUSE function, choosing Macro Resume resumes only the second macro; you cannot resume or return to the first macro automatically.
- PAUSE is ignored in custom worksheet functions, unless you manually run them by choosing the Run button from the Macro dialog box, which appears when you choose the Macro command from the Tools menu. PAUSE is also ignored if it's placed in a formula for which the resume behavior would be unclear, such as:

```
IF(Cost<10, AND(PAUSE(), SUM(!$A$1:$A$4)))
```

- If one macro runs a second macro that pauses, Microsoft Excel locks the calling cell in the first macro. If you try to edit this cell, Microsoft Excel displays an error message.
- To resume a paused macro, choose the Resume Macro button on the toolbar or run a macro containing a RESUME function.
- If one macro runs a second macro that pauses and you need to halt only the paused macro, use RESUME(2) instead of HALT. HALT halts all macros and prevents resuming or returning to any macro. For more information, see RESUME.

Tip Since the automatic Resume Macro button can be customized, you can create a custom toolbar that will appear whenever a macro pauses.

Example

The following macro formula checks to see if a variable named TestValue is greater than 9. If it is, the macro pauses; otherwise, the macro continues normally.

```
IF(TestValue>9, PAUSE())
```

Related Functions

<u>HALT</u>	Stops all macros from running
<u>RESUME</u>	Resumes a paused macro
<u>STEP</u>	Turns on macro single-stepping
List of <u>Control Functions</u>	

PRECISION

Macro Sheets Only

Equivalent to selecting or clearing the Precision As Displayed check box in the Calculation tab of the Options dialog box, which appears when you choose the Options command from the Tools menu. Controls how values are stored in cells. Use PRECISION when the results of formulas do not seem to match the values used to calculate the formulas.

Syntax

PRECISION(logical)

Logical is a logical value corresponding to the Precision As Displayed check box in the Calculation tab.

- If logical is TRUE, Microsoft Excel stores future entries at full precision (15 digits).
- If logical is FALSE or omitted, Microsoft Excel stores values exactly as they are displayed.

Caution The PRECISION function may permanently alter your data. PRECISION(FALSE) causes Microsoft Excel to change values on your worksheet or macro sheet to match displayed values. PRECISION(TRUE) causes Microsoft Excel to store future values at full precision, but it does not restore previously entered numbers to their original values.

Remarks

- Precision As Displayed does not affect numbers in General format. Numbers in General format are always calculated to full precision.
- Microsoft Excel calculates slightly faster when using full precision because with Precision As Displayed selected, Microsoft Excel has to round off numbers as it calculates.

Related Functions

FORMAT.NUMBER

Applies a number format to the selection

WORKSPACE

Changes workspace settings

List of Command-Equivalent Functions

QUIT

Macro Sheets Only

Equivalent to choosing the Exit command from the File menu in Microsoft Excel for Windows. Equivalent to choosing the Quit command from the File menu in Microsoft Excel for the Macintosh. Quits Microsoft Excel and closes any open documents. If open documents have unsaved changes, Microsoft Excel displays a message asking if you want to save them. You can use QUIT in an Auto_Close macro to force Microsoft Excel to quit when a particular worksheet or macro sheet is closed.

Syntax

QUIT()

Caution If you have cleared error-checking with an ERROR(FALSE) function, QUIT will not ask whether you want to save changes.

Remarks

When you use the QUIT function, Microsoft Excel does not run any Auto_Close macros before closing the document.

Examples

The following function displays a confirmation alert and quits Microsoft Excel if the user chooses OK:

```
IF(ALERT("Are you sure you want to quit Microsoft Excel?",1), QUIT(),)
```

Related Function

FILE.CLOSE Closes the active document

List of Command-Equivalent Functions

RESTART

Macro Sheets Only

Removes a number of RETURN statements from the stack. When one macro calls another, the RETURN statement at the end of the second macro returns control to the calling macro. You can use the RESTART function to determine which macro regains control.

Syntax

RESTART(level_num)

Level_num is a number specifying the number of previous RETURN statements you want to be ignored. If level_num is omitted, the next RETURN statement will halt macro execution.

For example, if the currently running macro has two "ancestors," that is, was called by one macro that, in turn, was called by another macro, using RESTART(1) in the third macro returns control to the first calling macro when the RETURN statement is encountered. The RESTART(1) formula removes one level of RETURN statements from Microsoft Excel's memory so that the second macro is skipped.

Remarks

RESTART is particularly useful if you frequently use macros to call other macros that in turn call other macros. Use RESTART in combination with IF statements to prevent macro execution from returning to macros that called, either directly or indirectly, the currently running macro.

Related Functions

List of [Control Functions](#)

RESULT

Macro Sheets Only

Specifies the type of data a macro or custom function returns. Use RESULT to make sure your macros, custom functions, or subroutines return values of the correct data type.

Syntax

RESULT(type_num)

Type_num is a number specifying the data type.

Type_num	Type of returned data
1	Number
2	Text
4	Logical
8	Reference
16	Error
64	Array

- Type_num can be the sum of the numbers in the preceding table to allow for more than one possible result type. For example, if type_num is 12, which equals 4 + 8, the result can be a logical or a reference value.
- If you omit type_num, it is assumed to be 7. Since 7 equals 1 + 2 + 4, the value returned can be a number (1), text (2), or logical value (4).

Examples

The following function specifies that a custom function's return value can be a number or a logical value (4+1=5):

```
RESULT (5)
```

Related Functions

<u>ARGUMENT</u>	Passes an argument to a macro
<u>RETURN</u>	Ends the currently running macro
<u>TYPE</u>	Returns a number indicating the data type of a value
List of <u>Control Functions</u>	

RETURN

Macro Sheets Only

Ends the currently running macro. If the currently running macro is a subroutine macro that was called by another macro, control is returned to the calling macro. If the currently running macro is a custom function, control is returned to the formula that called the custom function. If the currently running macro is a command macro started by the user with the Run button in the Macro dialog box or a shortcut key or by clicking an object, control is returned to the user.

Syntax

RETURN(value)

Value specifies what to return.

- If the macro is a custom function or a subroutine, value specifies what value to return. However, not all subroutines return values; the last line in macros that do not return values is =RETURN().
- If the macro is a command macro run by the user, value should be omitted.

Remarks

RETURN signals the end of a macro. Every macro must end with a RETURN or HALT function, but not every macro returns values.

Example

The following function returns the sum of the range B1:B10:

```
RETURN (SUM (B1 : B10) )
```

Related Functions

BREAK Interrupts a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop

HALT Stops all macros from running

RESULT Specifies the data type a custom function returns

List of Control Functions

SET.NAME

Macro Sheets Only

Defines a name on a macro sheet to refer to a value. The SET.NAME function is useful for storing values while the macro is calculating.

Syntax

SET.NAME(name_text, value)

Name_text is the name in the form of text that refers to value.

Value is the value you want to store in name_text.

- If value is omitted, the name name_text is deleted.
- If value is a reference, name_text is defined to refer to that reference.

Remarks

- If you want to define a name as a constant value, you can use the following syntax instead of SET.NAME:
name_text=value
See the first two examples following.
- SET.NAME defines names as absolute references, even if a relative reference is specified. See the third and fourth examples following.
- If you want name_text to refer permanently to the value of a referenced cell rather than to the reference itself, you must use the DEREf function. Use of DEREf prevents name_text from referring to a new value every time the contents of the referenced cell changes. See the last example following.

Tips

- If you need to return an array to a macro sheet (for example, if the macro needs a list of all open windows), assign a name to the array instead of placing the array information in a range of cells. For example:
SET.NAME("OpenDocuments", WINDOWS()) or
SET.NAME("OpenDocuments", {"WORKSHEET1", "WORKSHEET2"})
You can then use the INDEX function with the name you have defined to access items in the array stored in the name.
- When you're debugging a macro and want to know the current value assigned to a name created by SET.NAME, you can halt the macro, choose Define from the Name submenu on the Insert menu, and select the name from the Define Name dialog box.

Examples

Each of these formulas defines the name Counter to refer to the constant number 1 on the macro sheet:

```
SET.NAME("Counter", 1)  
Counter=1
```

Each of these formulas redefines Counter to refer to the current value of Counter plus 1:

```
SET.NAME("Counter", Counter+1)  
Counter=Counter+1
```

The following macro formula defines the name Reference to refer to cell \$A\$1:

```
SET.NAME("Reference", A1)
```

The following macro formula defines the name Results to refer to the cells \$A\$1:\$C\$3:

```
SET.NAME("Results", A1:C3)
```

The following macro formula defines the name Range as the current selection:

```
SET.NAME("Range", SELECTION())
```

If \$A\$1 contains the value 2, the following macro formula defines the name Index to refer to the constant

value 2:

```
SET.NAME("Index", Deref(A1))
```

Related Functions

DEFINE.NAME Defines a name on the active worksheet or macro sheet

SET.VALUE Sets the value of a cell on a macro sheet

List of Control Functions

SHOW.ACTIVE.CELL

Macro Sheets Only

Scrolls the active window so the active cell becomes visible. If an object is selected, SHOW.ACTIVE.CELL returns the #VALUE! error value and halts the macro.

Syntax

SHOW.ACTIVE.CELL()

Related Functions

ACTIVE.CELL Returns the reference of the active cell

FORMULA.GOTO Selects a named area or reference on any open document

List of Command-Equivalent Functions

UNLOCKED.NEXT **UNLOCKED.PREV**

Macro Sheets Only

Equivalent to pressing TAB or SHIFT+TAB to move to the next or previous unlocked cell in a protected worksheet. Use these functions when you want to control which cell is active on a protected sheet.

Syntax

UNLOCKED.NEXT()
UNLOCKED.PREV()

Related Functions

<u>CELL.PROTECTION</u>	Controls protection for the selected cells
<u>PROTECT.DOCUMENT</u>	Controls protection for the active document
List of <u>Command-Equivalent Functions</u>	

VOLATILE

Macro Sheets Only

Specifies whether a custom worksheet function is volatile or nonvolatile. A volatile custom function is recalculated every time a calculation occurs on the worksheet.

Syntax

VOLATILE(logical)

Logical is a logical value specifying whether the custom function is volatile or nonvolatile. If logical is TRUE or omitted, the function is volatile; if FALSE, nonvolatile.

Remarks

- VOLATILE must precede every other formula in the custom function except RESULT and ARGUMENT.
- Normally, a worksheet recalculates a cell containing a nonvolatile custom function only when any part of the complete formula in the cell is recalculated. Use VOLATILE(TRUE) to recalculate the function every time the worksheet is recalculated.
- Most custom functions are nonvolatile by default, but custom functions with reference arguments are volatile by default. Use VOLATILE(FALSE) to prevent these functions from being recalculated unnecessarily often.

Related Function

RESULT Specifies the data type a custom function returns
List of Control Functions

WAIT

Macro Sheets Only

Pauses the macro until the time specified by the serial number.

Syntax

WAIT(serial_number)

Serial_number is the date-time code used by Microsoft Excel for date and time calculations. You can give serial_number as text, such as "4:30 PM", or as a formula, such as NOW()+"00:00:04", instead of as a number. The text or formula is automatically converted to a serial number. For more information about serial_number, see NOW.

Important WAIT suspends all Microsoft Excel activity and may prevent you from performing other operations on your computer. Background processes, such as printing and recalculation, are continued.

Example

Use WAIT with NOW to pause a macro for a specified length of time. For example, the following macro formula waits 3 seconds from the time the functions are evaluated:

```
WAIT(NOW()+"00:00:03")
```

Related Functions

ON.TIME

Runs a macro at a specific time

NOW

Returns the serial number of the current date and time

List of Control Functions

WHILE

Macro Sheets Only

Carries out the statements between the WHILE function and the next NEXT function until logical_test is FALSE. Use WHILE-NEXT loops to carry out a series of macro formulas while a certain condition remains TRUE.

Syntax

WHILE(logical_test)

Logical_test is a value or formula that evaluates to TRUE or FALSE. If logical_test is FALSE the first time the WHILE function is reached, the macro skips the loop and resumes running at the statement after the next NEXT function. If there is no NEXT function in the same column, WHILE displays an error message and interrupts the macro.

Remarks

If you know exactly how many times you'll need to carry out the statements within a loop, in most cases you should use a FOR-NEXT loop.

Example

The following statement starts a loop that executes while the current cell contains a number, text, or a logical value:

```
=WHILE (TYPE (ACTIVE.CELL ()) <5) )
```

The following statement starts a loop that executes until the position in the open file identified as FileName reaches the end of the file:

```
=WHILE (FPOS (FileName) <=FSIZE (FileName) )
```

Related Functions

<u>FOR</u>	Starts a FOR-NEXT loop
<u>FOR.CELL</u>	Starts a FOR.CELL-NEXT loop
<u>IF</u>	Specifies an action to take if a logical test is TRUE
<u>NEXT</u>	Ends a FOR-NEXT, FOR.CELL-NEXT, or WHILE-NEXT loop
List of <u>Control Functions</u>	

ANOVA1

Macro Sheets Only

Performs single-factor analysis of variance, which tests the hypothesis that means from several samples are equal.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

ANOVA1(inprng, outrng, grouped, labels, alpha)

ANOVA1?(inprng, outrng, grouped, labels, alpha)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Grouped is a text character that indicates whether the data in the input range is organized by row or column.

- If grouped is "C" or omitted, then the data is organized by column.
- If grouped is "R", then the data is organized by row.

Labels is a logical value that describes where the labels are located in the input range, as shown in the following table:

Labels	Grouped	Labels are in
TRUE	"C"	First row of the input range.
TRUE	"R"	First column of the input range.
FALSE or omitted	(ignored)	No labels. All cells in the input range are data.

Alpha is the significance level at which to evaluate critical values for the F statistic. If omitted, alpha is 0.05.

Related Functions

[ANOVA2](#) Performs two-factor analysis of variance with replication

[ANOVA3](#) Performs two-factor analysis of variance without replication

List of [Statistical Functions](#)

ANOVA2

Macro Sheets Only

Performs two-factor analysis of variance with replication.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

ANOVA2(inprng, outrng, **sample_rows**, alpha)

ANOVA2?(inprng, outrng, sample_rows, alpha)

Inprng is the input range. The input range should contain labels in the first row and column.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Sample_rows is the number of rows in each sample.

Alpha is the significance level at which to evaluate critical values for the F statistic. If omitted, alpha is 0.05.

Related Functions

[ANOVA1](#) Performs single-factor analysis of variance

[ANOVA3](#) Performs two-factor analysis of variance without replication

List of [Statistical Functions](#)

ANOVA3

Macro Sheets Only

Performs two-factor analysis of variance without replication.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

ANOVA3(inprng, outrng, labels, alpha)

ANOVA3?(inprng, outrng, labels, alpha)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then the first row and column of the input range contain labels.
- If labels is FALSE or omitted, all cells in inprng are considered data. Microsoft Excel will then generate the appropriate data labels for the output table.

Alpha is the significance level at which to evaluate critical values for the F statistic. If omitted, alpha is 0.05.

Related Functions

[ANOVA1](#) Performs single-factor analysis of variance

[ANOVA2](#) Performs two-factor analysis of variance with replication

List of [Statistical Functions](#)

DESCR

Macro Sheets Only

Generates descriptive statistics for data in the input range.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

DESCR(inprng, outrng, grouped, labels, summary, ds_large, ds_small, confid)

DESCR?(inprng, outrng, grouped, labels, summary, ds_large, ds_small, confid)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Grouped is a text character that indicates whether the data in the input range is organized by row or column.

- If grouped is "C" or omitted, then the data is organized by column.
- If grouped is "R" then the data is organized by row.

Labels is a logical value that describes where the labels are located in the input range, as shown in the following table:

Labels	Grouped	Labels are in
TRUE	"C"	First row of the input range.
TRUE	"R"	First column of the input range.
FALSE or omitted	(ignored)	No labels. All cells in the input range are data.

Summary is a logical value. If TRUE, DESCR reports the summary statistics. If FALSE or omitted, no summary statistics are reported.

Ds_large is an integer k. If ds_large is present, DESCR reports the k-th largest data point. If ds_large is omitted, the value is not reported.

Ds_small is an integer k. If ds_small is present, DESCR reports the k-th smallest data point. If ds_small is omitted, the value is not reported.

Confid is the confidence level of the mean. If confid is given, DESCR reports the confidence interval for the input range. If confid is omitted, the confidence interval is 95%. For more information about the confidence interval calculation, see CONFIDENCE.

Related Functions

AVERAGE	Returns the average of its arguments
CONFIDENCE	Returns a confidence interval for a population mean
KURT	Returns the kurtosis of a data set
LARGE	Returns the k-th largest value in a data set
MAX	Returns the maximum value in a list of arguments
MEDIAN	Returns the median of the given numbers
MIN	Returns the minimum value in a list of arguments
MODE	Returns the most common value in a data set
SKEW	Returns the skewness of a distribution
SMALL	Returns the k-th smallest value in a data set
STDEV	Estimates standard deviation based on a sample
VAR	Estimates variance based on a sample

List of [Statistical Functions](#)

EXPON

Macro Sheets Only

Predicts a value based on the forecast for the prior period, adjusted for the error in that prior forecast.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

EXPON(inprng, outrng, damp, stderrs, chart)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Damp is the damping factor. If omitted, damp is 0.3.

Stderrs is a logical value. If TRUE, standard error values are included in the output table. If FALSE, standard errors are not included.

Chart is a logical value. If TRUE, EXPON generates a chart for the actual and forecast values. If FALSE, the chart is not generated.

Related Functions

<u>FORECAST</u>	Returns a value along a linear trend
<u>GROWTH</u>	Returns values along an exponential trend
<u>MOVEAVG</u>	Returns values along a moving average trend
<u>TREND</u>	Returns values along a linear trend

FOURIER

Macro Sheets Only

Performs a Fourier transform.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

FOURIER(inprng, outrng, inverse, labels)

FOURIER?(inprng, outrng, inverse, labels)

Inprng is the input range. The number of cells in the input range must be equal to a power of two (2, 4, 8, 16, ...).

Outrng is the first cell in the output range or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Inverse is a logical value. If TRUE, an inverse Fourier transform is performed. If FALSE or omitted, a forward Fourier transform is performed.

Labels is a logical value.

- If labels is TRUE, then the first row or column of inprng contains labels.
- If labels is FALSE or omitted, all cells in inprng are considered data. Microsoft Excel generates appropriate data labels for the output table.

Related Function

[SAMPLE](#) Samples data

List of [Statistical Functions](#)

FTESTV

Macro Sheets Only

Performs a two-sample F-test.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

FTESTV(inprng1, inprng2, outrng, labels)

FTESTV?(inprng1, inprng2, outrng, labels)

Inprng1 is the input range for the first data set.

Inprng2 is the input range for the second data set.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then the first row or column of inprng1 and inprng2 contain labels.
- If labels is FALSE or omitted, all cells in inprng1 and inprng2 are considered data. Microsoft Excel generates appropriate data labels for the output table.

Related Functions

[FDIST](#) Returns the F probability distribution

[FINV](#) Returns the inverse of the F probability distribution

[FTEST](#) Returns the result of an F-test

List of [Statistical Functions](#)

HISTOGRAM

Macro Sheets Only

Calculates individual and cumulative percentages for a range of data and a corresponding range of data bins.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

HISTOGRAM(inprng, outrng, binrng, pareto, chartc, chart, labels)

HISTOGRAM?(inprng, outrng, binrng, pareto, chartc, chart, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Binnrg is an optional set of numbers that define the bin ranges. The values must be in ascending order. The values are interpreted as more than value A up to value B, more than value B up to value C, and so on. One additional bin is created for values for less than the minimum value specified in binrng.

Pareto is a logical value.

- If pareto is TRUE, data in the output table is presented in both ascending-bin order and descending-frequency order.
- If pareto is FALSE or omitted, data in the output table is presented in ascending-bin order only.

Chartc is a logical value. If chartc is TRUE, HISTOGRAM generates a cumulative percentages column in the output table. If both chartc and chart are TRUE, HISTOGRAM also includes a cumulative percentage line in the histogram chart. If omitted, chartc is FALSE.

Chart is a logical value. If chart is TRUE, HISTOGRAM generates a histogram chart in addition to the output table. If omitted, chart is FALSE.

Labels is a logical value.

- If labels is TRUE, then the first row or column of inprng contains labels.
- If labels is FALSE or omitted, all cells in inprng are considered data. Microsoft Excel generates appropriate data labels for the output table.

Related Function

[MODE](#) Returns the most common value in a data set

List of [Statistical Functions](#)

MCORREL

Macro Sheets Only

Returns a correlation matrix that measures the correlation between two or more data sets that are scaled to be independent of the unit of measurement.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

MCORREL(inprng, outrng, grouped, labels)

MCORREL?(inprng, outrng, grouped, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Grouped is a text character that indicates whether the data in the input range is organized by row or column.

- If grouped is "C" or omitted, then the data is organized by column.
- If grouped is "R", then the data is organized by row.

Labels is a logical value that describes where the labels are located in the input range, as shown in the following table:

Labels	Grouped	Labels are in
TRUE	"C"	First row of the input range.
TRUE	"R"	First column of the input range.
FALSE or omitted	(ignored)	No labels. All cells in the input range are data.

Related Functions

[CORREL](#) Returns the correlation coefficient between two data sets

[COVAR](#) Returns covariance, the average of the products of paired deviations

[MCOVAR](#) Returns the covariance between two or more data sets

List of [Statistical Functions](#)

MCOVAR

Macro Sheets Only

Returns a covariance matrix that measures the covariance between two or more data sets.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

MCOVAR(inprng, outrng, grouped, labels)

MCOVAR?(inprng, outrng, grouped, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Grouped is a text character that indicates whether the data in the input range is organized by row or column.

- If grouped is "C" or omitted, then the data is organized by column.
- If grouped is "R", then the data is organized by row.

Labels is a logical value that describes where the labels are located in the input range, as shown in the following table:

Labels	Grouped	Labels are in
TRUE	"C"	First row of the input range
TRUE	"R"	First column of the input range
FALSE or omitted	(ignored)	No labels. All cells in the input range are data.

Related Functions

[CORREL](#) Returns the correlation coefficient between two data sets

[COVAR](#) Returns covariance, the average of the products of paired deviations

[MCORREL](#) Returns the correlation coefficient of two or more data sets that are scaled to be independent of the unit of measurement

List of [Statistical Functions](#)

MOVEAVG

Macro Sheets Only

Projects values in a forecast period, based on the average value of the variable over a specific number of preceding periods.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

MOVEAVG(inprng, outrng, interval, stderrs, chart, labels)

MOVEAVG?(inprng, outrng, interval, stderrs, chart, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Interval is the number of values to include in the moving average. If omitted, interval is 3.

Stderrs is a logical value.

- If stderrs is TRUE, standard error values are included in the output table.
- If stderrs is FALSE or omitted, standard errors are not included in the output table.

Chart is a logical value.

- If chart is TRUE, then MOVEAVG generates a chart for the actual and forecast values.
- If chart is FALSE or omitted, the chart is not generated.

Labels is a logical value.

- If labels is TRUE, then the first row or column of inprng contains labels.
- If labels is FALSE or omitted, all cells in inprng are considered data. Microsoft Excel generates appropriate data labels for the output table.

Related Functions

[EXPON](#) Predicts a value based on the forecast for the prior period

[FORECAST](#) Returns a value along a linear trend

[GROWTH](#) Returns values along an exponential trend

[TREND](#) Returns values along a linear trend

List of [Statistical Functions](#)

PTTESTM

Macro Sheets Only

Performs a paired two-sample Student's t-Test for means.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

PTTESTM(inprng1, inprng2, outrng, labels, alpha, difference)

PTTESTM?(inprng1, inprng2, outrng, labels, alpha, difference)

Inprng1 is the input range for the first data set.

Inprng2 is the input range for the second data set.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then labels are in the first row or column of the input ranges.
- If labels is FALSE or omitted, all cells in inprng1 and inprng2 are considered data. The output table will include default row or column headings.

Alpha is the confidence level for the test. If omitted, alpha is 0.05.

Difference is the hypothesized mean difference. If omitted, difference is 0.

Related Functions

[PTTESTV](#)

Performs a two-sample Student's t-Test, assuming unequal variances

[TDIST](#)

Returns the Student's t-distribution

[TTEST](#)

Returns the probability associated with a Student's t-Test

[TTESTM](#)

Performs a two-sample Student's t-Test for means, assuming equal variances

List of [Statistical Functions](#)

PTTESTV

Macro Sheets Only

Performs a two-sample Student's t-Test, assuming unequal variances.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

PTTESTV(inprng1, inprng2, outrng, labels, alpha)

PTTESTV?(inprng1, inprng2, outrng, labels, alpha)

Inprng1 is the input range for the first data set.

Inprng2 is the input range for the second data set.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then labels are in the first row or column of the input ranges.
- If labels is FALSE or omitted, all cells in inprng1 and inprng2 are considered data. The output table will include default row or column headings.

Alpha is the confidence level for the test. If omitted, alpha is 0.05.

Related Functions

[PTTESTM](#) Performs a paired two-sample Student's t-Test for means

[TDIST](#) Returns the Student's t-distribution

[TTEST](#) Returns the probability associated with a Student's t-Test

[TTESTM](#) Performs a two-sample Student's t-Test for means, assuming equal variances

List of [Statistical Functions](#)

RANDOM

Macro Sheets Only

Fills a range with independent random or patterned numbers drawn from one of several distributions.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

RANDOM provides six different random distributions and one patterned data option. Because the distributions require different argument lists, there are seven syntax forms for RANDOM.

Syntax 1

Uniform distribution

RANDOM(outrng, variables, points, **distribution**, seed, **from**, **to**)

RANDOM?(outrng, variables, points, distribution, seed, from, to)

Syntax 2

Normal distribution

RANDOM(outrng, variables, points, **distribution**, seed, **mean**, **standard_dev**)

RANDOM?(outrng, variables, points, distribution, seed, mean, standard_dev)

Syntax 3

Bernoulli distribution

RANDOM(outrng, variables, points, **distribution**, seed, **probability**)

RANDOM?(outrng, variables, points, distribution, seed, probability)

Syntax 4

Binomial distribution

RANDOM(outrng, variables, points, **distribution**, seed, **probability**, **trials**)

RANDOM?(outrng, variables, points, distribution, seed, probability, trials)

Syntax 5

Poisson distribution

RANDOM(outrng, variables, points, **distribution**, seed, **lambda**)

RANDOM?(outrng, variables, points, distribution, seed, lambda)

Syntax 6

Patterned distribution

RANDOM(outrng, variables, points, **distribution**, seed, **from**, **to**, **step**, **repeat_num**, **repeat_seq**)

RANDOM?(outrng, variables, points, distribution, seed, from, to, step, repeat_num, repeat_seq)

Syntax 7

Discrete distribution

RANDOM(outrng, variables, points, **distribution**, seed, **inprng**)

RANDOM?(outrng, variables, points, distribution, seed, inprng)

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Variables is the number of random number sets to generate. RANDOM will generate variables columns of random numbers. If omitted, variables is equal to the number of columns in the output range.

Points is the number of data points per random number set. RANDOM will generate points rows of random numbers for each random number set. If omitted, points is equal to the number of rows in the output range. Points is ignored when distribution is 6 (Patterned).

Distribution indicates the type of number distribution.

Distribution	Distribution type
--------------	-------------------

1	Uniform
2	Normal
3	Bernoulli

4	Binomial
5	Poisson
6	Patterned
7	Discrete

Seed is an optional value with which to begin random number generation. Seed is ignored when distribution is 6 (Patterned) or 7 (Discrete).

From is the lower bound.

To is the upper bound.

Mean is the mean.

Standard_dev is the standard deviation.

Probability is the probability of success on each trial.

Trials is the number of trials.

Lambda is the Poisson distribution parameter.

Step is the increment between from and to.

Repeat_num is the number of times to repeat each value.

Repeat_seq is the number of times to repeat each sequence of values.

Inprng is a two-column range of values and their probabilities.

Related Function

[RAND](#) Returns a random number between 0 and 1

List of [Statistical Functions](#)

RANKPERC

Macro Sheets Only

Returns a table that contains the ordinal and percent rank of each value in a data set.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

RANKPERC(inprng, outrng, grouped, labels)

RANKPERC?(inprng, outrng, grouped, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Grouped is a text character that indicates whether the data in the input range is organized by row or column.

- If grouped is "C" or omitted, then the data is organized by column.
- If grouped is "R", then the data is organized by row.

Labels is a logical value that describes where the labels are located in the input range, as shown in the following table:

Labels	Grouped	Labels are in
TRUE	"C"	First row of the input range.
TRUE	"R"	First column of the input range.
FALSE or omitted	(ignored)	No labels. All cells in the input range are data.

Related Functions

PERCENTILE	Returns the k-th percentile of values in a range
PERCENTRANK	Returns the percentage rank of a value in a data set
QUARTILE	Returns the quartile of a data set
SMALL	Returns the k-th smallest value in a data set
List of Statistical Functions	

REGRESS

Macro Sheets Only

Performs multiple linear regression analysis.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

REGRESS(inpyrng, inpxrng, constant, labels, confid, soutrng, residuals, sresiduals, rplots, lplots, routrng, nplots, poutrng)

REGRESS?(inpyrng, inpxrng, constant, labels, confid, soutrng, residuals, sresiduals, rplots, lplots, routrng, nplots, poutrng)

Inpyrng is the input range for the y-values (dependent variable).

Inpxrng is the input range for the x-values (independent variable).

Constant is a logical value. If constant is TRUE, the y-intercept is assumed to be zero (the regression line passes through the origin). If constant is FALSE or omitted, the y-intercept is assumed to be a non-zero number.

Labels is a logical value.

- If labels is TRUE, then the first row or column of the input ranges contain labels.
- If labels is FALSE or omitted, all cells in inpyrng and inpxrng are considered data. Microsoft Excel will then generate the appropriate data labels for the output table.

Confid is an additional confidence level to apply to the regression. If omitted, confid is 95%.

Soutrng is the first cell (the upper-left cell) in the output table or the name, as text, of the new sheet to contain the summary output table. If FALSE, blank, or omitted, places the summary output table in a new workbook. Microsoft Excel version 5.0 uses a single output table for REGRESS; Microsoft Excel version 4.0 used three different output tables for summary, residual, and probability data.

Residuals is a logical value. If residuals is TRUE, REGRESS includes residuals in the output table. If residuals is FALSE or omitted, residuals are not included.

Sresiduals is a logical value. If sresiduals is TRUE, REGRESS includes standardized residuals in the output table. If sresiduals is FALSE or omitted, standardized residuals are not included.

Rplots is a logical value. If rplots is TRUE, REGRESS generates separate charts for each x versus the residual. If rplots is FALSE or omitted, separate charts are not generated.

Lplots is a logical value. If lplots is TRUE, REGRESS generates a chart showing the regression line fitted to the observed values. If lplots is FALSE or omitted, the chart is not generated.

Routrng is the first cell (the upper-left cell) in the residuals output table or the name, as text, of the new sheet to contain the residuals output table. If FALSE, blank, or omitted, places the residuals output table in a new worksheet. This argument is for compatibility with Microsoft Excel 4.0 only and is ignored in Microsoft Excel version 5.0.

Nplots is a logical value. If nplots is TRUE, REGRESS generates a chart of normal probabilities. If nplots is FALSE or omitted, the chart is not generated.

Poutrng is the first cell (the upper-left cell) in the probability data output table or the name, as text, of the new sheet to contain the probability output table. If FALSE, blank, or omitted, places the probability output table in a new worksheet. This argument is for compatibility with Microsoft Excel 4.0 only and is ignored in Microsoft Excel version 5.0.

Related Functions

<u>FORECAST</u>	Returns a value along a linear trend
<u>GROWTH</u>	Returns values along an exponential trend
<u>LINEST</u>	Returns the parameters of a linear trend
<u>LOGEST</u>	Returns the parameters of an exponential trend
<u>TREND</u>	Returns values along a linear trend

List of [Statistical Functions](#)

SAMPLE

Macro Sheets Only

Samples data.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SAMPLE(inprng, outrng, method, rate, labels)

SAMPLE?(inprng, outrng, method, rate, labels)

Inprng is the input range.

Outrng is the first cell (the upper-left cell) in the output column or the name, as text, of a new sheet to contain the output column. If FALSE, blank, or omitted, places the output table in a new workbook.

Method is a text character that indicates the type of sampling.

- If method is "P", then periodic sampling is used. The input range is sampled every nth cell, where $n = \text{rate}$.

- If method is "R", then random sampling is used. The output column will contain rate samples.

Rate is the sampling rate, if method is "P" (periodic sampling). Rate is the number of samples to take if method is "R" (random sampling).

Labels is a logical value.

- If labels is TRUE, then the first row or column of inprng contains labels.

- If labels is FALSE or omitted, all cells in inprng are considered data. Microsoft Excel generates appropriate data labels for the output table.

Related Function

[FOURIER](#) Performs a Fourier transform

List of [Statistical Functions](#)

TTESTM

Macro Sheets Only

Performs a two-sample Student's t-Test for means, assuming equal variances.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

TTESTM(inprng1, inprng2, outrng, labels, alpha, difference)

TTESTM?(inprng1, inprng2, outrng, labels, alpha, difference)

Inprng1 is the input range for the first data set.

Inprng2 is the input range for the second data set.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then labels are in the first row or column of the input ranges.
- If labels is FALSE or omitted, all cells in inprng1 and inprng2 are considered data. The output table will include default row or column headings.

Alpha is the confidence level for the test. If omitted, alpha is 0.05.

Difference is the hypothesized difference in means. If omitted, difference is 0.

Related Functions

[PTTESTM](#)

Performs a paired two-sample Student's t-Test for means

[PTTESTV](#)

Performs a two-sample Student's t-Test, assuming unequal variances

[TDIST](#)

Returns the Student's t-distribution

[TTEST](#)

Returns the probability associated with a Student's t-Test

List of [Statistical Functions](#)

ZTESTM

Macro Sheets Only

Performs a two-sample z-test for means, assuming the two samples have known variances.

If this function is not available, you must install the Analysis ToolPak add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

ZTESTM(inprng1, inprng2, outrng, labels, alpha, difference, var1, var2)

ZTESTM?(inprng1, inprng2, outrng, labels, alpha, difference, var1, var2)

Inprng1 is the input range for the first data set.

Inprng2 is the input range for the second data set.

Outrng is the first cell (the upper-left cell) in the output table or the name, as text, of a new sheet to contain the output table. If FALSE, blank, or omitted, places the output table in a new workbook.

Labels is a logical value.

- If labels is TRUE, then the first row or column of the input ranges contains labels.
- If labels is FALSE or omitted, all cells in inprng1 and inprng2 are considered data. Microsoft Excel will then generate the appropriate data labels for the output table.

Alpha is the confidence level for the test. If omitted, alpha is 0.05.

Difference is the hypothesized difference in means. If omitted, difference is 0.

Var1 is the variance of the first data set.

Var2 is the variance of the second data set.

Related Functions

NORMDIST	Returns the normal cumulative distribution
NORMINV	Returns the inverse of the normal cumulative distribution
NORMSDIST	Returns the standard normal cumulative distribution
NORMSINV	Returns the inverse of the standard normal cumulative distribution
ZTEST	Returns the two-tailed P-value of a z-test
List of Statistical Functions	

CELL.PROTECTION

Macro Sheets Only

Equivalent to choosing the Protection tab in the Format Cells dialog box, which appears when you choose the Cells command from the Format menu. Allows you to control cell protection and display.

Arguments are logical values corresponding to check boxes in the Protection tab. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted and the setting has been previously changed from the defaults, the setting is not changed.

Syntax

CELL.PROTECTION(locked, hidden)

CELL.PROTECTION?(locked, hidden)

Locked corresponds to the Locked check box. The default is TRUE.

Hidden corresponds to the Hidden check box. The default is FALSE.

Remarks

Options selected in the Protection tab of the Format Cells dialog box or with the CELL.PROTECTION function are activated only when the Protect Sheet command is chosen from the Protection submenu on the Tools menu or the PROTECT.DOCUMENT function is used to select protection.

Related Functions

PROTECT.DOCUMENT

Controls protection for the active document

SAVE.AS

Saves a document and allows you to specify the name, file type, password, backup file, and location of the document

List of Command-Equivalent Functions

COLUMN.WIDTH

Macro Sheets Only

Equivalent to choosing the Width command on the Column submenu of the Format menu. Changes the width of the columns in the specified reference.

Syntax

COLUMN.WIDTH(width_num, reference, standard, type_num, standard_num)

COLUMN.WIDTH?(width_num, reference, standard, type_num, standard_num)

Width_num specifies how wide you want the columns to be in units of one character of the font corresponding to the Normal cell style. Width_num is ignored if standard is TRUE or if type_num is provided.

Reference specifies the columns for which you want to change the width.

- If reference is specified, it must be either an external reference to the active worksheet, such as !\$A:\$C or !Database, or an R1C1-style reference in the form of text, such as "C1:C3", "C[-4]:C[-2]", or "Database".
- If reference is a relative R1C1-style reference in the form of text, it is assumed to be relative to the active cell.
- If reference is omitted, it is assumed to be the current selection.

Standard is a logical value corresponding to the Standard Width command from the Column submenu on the Format menu.

- If standard is TRUE, Microsoft Excel sets the column width to the currently defined standard (default) width and ignores width_num.
- If standard is FALSE or omitted, Microsoft Excel sets the width according to width_num or type_num.

Type_num is a number from 1 to 3 corresponding to the Hide, Unhide, or AutoFit Selection commands, respectively, on the Column submenu of the Format menu.

Type_num	Action taken
1	Hides the column selection by setting the column width to 0
2	Unhides the column selection by setting the column width to the value set before the selection was hidden
3	Sets the column selection to a best-fit width, which varies from column to column depending on the length of the longest data string in each column

Standard_num specifies how wide the standard width is, and is measured in points. If standard_num is omitted, the standard width setting remains unchanged.

Remarks

- Changing the value of standard_num changes the width of all columns except those that have been set to a custom value.
- If any of the argument settings conflict, such as when standard is TRUE and type_num is 3, Microsoft Excel uses the type_num argument and ignores any arguments that conflict with type_num.
- If you are recording a macro while using a mouse and you change column widths by dragging the column border, Microsoft Excel records the references of the columns using R1C1-style references in the form of text.

Related Function

ROW.HEIGHT Changes the heights of rows
List of Command-Equivalent Functions

COMBINATION

Macro Sheets Only

Equivalent to choosing the Combination item from the Galleries list box within the AutoFormat dialog box, which appears when you choose the AutoFormat command from the Format menu. This function is available only when a chart is made active. Changes the format of the active chart to the combination format you select from the gallery.

Syntax

COMBINATION(type_num)

COMBINATION?(type_num)

Type_num is a number corresponding to the combination chart you want.

Related Functions

[FORMAT.MAIN](#)

Formats a main chart

[FORMAT.OVERLAY](#)

Formats an overlay chart

List of [Command-Equivalent Functions](#)

CUT

Macro Sheets Only

Equivalent to choosing the Cut command from the Edit menu. Cuts or moves data or objects.

Syntax

CUT(from_reference, to_reference)

From_reference is a reference to the cell or range of cells you want to cut. If from_reference is omitted, it is assumed to be the current selection.

To_reference is a reference to the cell or range of cells where you want to paste what you have cut.

- To_reference should be a single cell or an enlarged multiple of from_reference. For example, if from_reference is a 2 by 4 rectangle, to_reference can be a 4 by 8 rectangle.
- To_reference can be omitted so that you can paste from_reference later using the PASTE or PASTE.SPECIAL functions.

Remarks

The following information may be helpful if you're having problems with CUT updating references in unexpected ways. When you move cells using CUT, formulas that referred to from_reference will refer to to_reference, and formulas that referred to to_reference may return #REF! error values. However, if from_reference or to_reference contains references that are calculated at runtime (for example, CUT(ACTIVE.CELL(), !B1)), then Microsoft Excel does not update those references when the CUT function is run, so no error values are returned.

Related Functions

COPY Copies and pastes data or objects

PASTE Pastes cut or copied data

List of Command-Equivalent Functions

DEMOTE

Macro Sheets Only

Equivalent to clicking the Group tool. Demotes, or groups, the selected rows or columns in an outline. Use DEMOTE to change the configuration of an outline by grouping rows or columns of information.

Syntax

DEMOTE(row_col)

DEMOTE?(row_col)

Row_col specifies whether to group rows or columns.

Row_col	Demotes
1 or omitted	Rows
2	Columns

Remarks

- If the selection consists of an entire row or rows, then rows are demoted even if row_col is 2. Similarly, selection of an entire column overrides row_col 1.
- If the selection is unambiguous (an entire row or column), then DEMOTE? will not display the dialog box.

Related Functions

PROMOTE Promotes the selection in an outline
SHOW.DETAIL Expands or collapses a portion of an outline
SHOW.LEVELS Displays a specific number of levels of an outline
List of Command-Equivalent Functions

FILE.DELETE

Macro Sheets Only

Deletes a file from the disk. Although you will normally delete files manually, you can, for example, use FILE.DELETE in a macro to delete temporary files created by the macro.

Syntax

FILE.DELETE(file_text)

FILE.DELETE?(file_text)

File_text is the name of the file to delete.

Remarks

- If Microsoft Excel can't find file_text, it displays a message saying that it cannot delete the file. To avoid this, include the entire path in file_text. See the following second and fifth examples. You can also use FILES to generate an array of filenames and then check if the file you want to delete is in the array. For an example of how to see if an entry is in an array, see OR.
- If a file is open when you delete it, the file is removed from the disk but remains open in Microsoft Excel.
- In the dialog-box form, FILE.DELETE?, you can use an asterisk (*) to represent any series of characters and a question mark (?) to represent any single character. See the following third and sixth examples.

Examples

In Microsoft Excel for Windows, the following macro formula deletes a file called CHART1.XLS from the current directory:

```
FILE.DELETE("CHART1.XLS")
```

The following macro formula deletes a file called 92INFO.XLS kept in the EXCEL\SALES subdirectory:

```
FILE.DELETE("C:\EXCEL\SALES\92INFO.XLS")
```

The following macro formula displays the Delete dialog box listing all documents whose extensions begin with the letters "XL":

```
FILE.DELETE?("*XL?")
```

In Microsoft Excel for the Macintosh, the following macro formula deletes a file called CHART1 from the current folder:

```
FILE.DELETE("CHART1")
```

The following macro formula deletes a file called 1992 INFO kept in a series of nested folders:

```
FILE.DELETE("HARD DISK:EXCEL 5:SALES WORKSHEETS:1992 INFO")
```

The following macro formula displays the Delete dialog box listing all documents beginning with the word "Clients":

```
FILE.DELETE?("Clients*")
```

Related Functions

FILE.CLOSE Closes the active document

FILES Returns the filenames in the specified directory or folder

List of Command-Equivalent Functions

FULL

Macro Sheets Only

Equivalent to pressing CTRL+F10 (full size) and CTRL+F5 (previous size) or double-clicking the title bar in Microsoft Excel for Windows version 3.0 or earlier. Equivalent to double-clicking the title bar or clicking the zoom box in Microsoft Excel for the Macintosh version 3.0 or earlier. This function is included only for macro compatibility. To perform the equivalent of a FULL(TRUE) function in Microsoft Excel version 4.0, use the WINDOW.MAXIMIZE function. To perform the equivalent of a FULL(FALSE) function in Microsoft Excel version 4.0, use the WINDOW.RESTORE function.

Syntax

FULL(logical)

Related Functions

List of [Command-Equivalent Functions](#)

GROUP

Macro Sheets Only

Equivalent to choosing the Group command from the Placement submenu on the Format menu. Creates a single object from several selected objects and returns the object identifier of the group (for example, "Group 5"). Use GROUP to combine a number of objects so that you can move or resize them together.

If no object is selected, only one object is selected, or a group is already selected, GROUP returns the #VALUE! error value and interrupts the macro.

Syntax

GROUP()

Related Function

UNGROUP Separates a grouped object

List of Command-Equivalent Functions

HPAGE

Macro Sheets Only

Horizontally scrolls through the active window one window at a time. Use HPAGE to change the displayed area of a worksheet or macro sheet.

Syntax

HPAGE(num_windows)

Num_windows specifies the number of windows to scroll through the active window horizontally. A window is defined as the number of visible columns. If three columns are visible in the window, HPAGE scrolls through in increments of three columns.

- If num_windows is positive, HPAGE scrolls to the right.
- If num_windows is negative, HPAGE scrolls to the left.

Related Functions

<u>HLINE</u>	Horizontally scrolls through the active window by columns
<u>HSCROLL</u>	Horizontally scrolls through a document by percentage or by column number
<u>VLIN</u>	Vertically scrolls through the active window by rows
<u>VPAGE</u>	Vertically scrolls through the active window one window at a time
<u>VSCROLL</u>	Vertically scrolls through a document by percentage or by row number
List of <u>Command-Equivalent Functions</u>	

MOVE

Macro Sheets Only

Equivalent to moving a window by dragging its title bar in Microsoft Excel version 3.0 or earlier. MOVE is also equivalent to choosing the Move command from the Control menu in Microsoft Windows. This function is included only for macro compatibility and will be converted to WINDOW.MOVE when you open older macro sheets. For more information, see WINDOW.MOVE.

Syntax

MOVE(x_pos, y_pos, window_text)

MOVE?(x_pos, y_pos, window_text)

Related Functions

WINDOW.MOVE Sizes a window

WINDOW.SIZE Moves a window

List of Command-Equivalent Functions

NOTE

Macro Sheets Only

Equivalent to choosing the Note command from the Insert menu. Creates a note or replaces characters in a note. Use NOTE to create a cell note or to replace a specified number of characters with other text.

Syntax

NOTE(add_text, cell_ref, start_char, num_chars)

NOTE?()

Add_text is text of up to 255 characters you want to add to a note. Add_text must be enclosed in quotation marks.

- If add_text is omitted, it is assumed to be "" (empty text) unless the note includes sound.
- If a note contains sound and you omit add_text, Microsoft Excel deletes the entire note, but if you explicitly specify empty text, Microsoft Excel deletes only the text, not the sound. To erase only the sound, use the SOUND.NOTE function.

Cell_ref is the cell to which you want to add the note text. If cell_ref is omitted, add_text is added to the active cell's note.

Start_char is the number of the character at which you want add_text to be added. If start_char is omitted, it is assumed to be 1. If there is no existing note, start_char is ignored.

Num_chars is the number of characters that you want to replace in the note. If num_chars is omitted, it is assumed to be equal to the length of the note.

Remarks

- NOTE returns the number of the last character entered in the cell note. This is useful if you want to know how many characters are in the text string so it can be manipulated with other functions such as RIGHT, LEFT, and MID.
- The dialog-box form of this function, NOTE?, takes no arguments.
- NOTE () deletes the note attached to the active cell.

To find out if a cell has a note attached to it, use GET.CELL.

Related Functions

GET.NOTE Returns characters from a note

SOUND.NOTE Records or imports sounds into or erases sound from cell notes

List of Command-Equivalent Functions

OUTLINE

Macro Sheets Only

Creates an outline and defines settings for automatically creating outlines.

The first three arguments are logical values corresponding to check boxes in the Outline dialog box, which appears when you choose the Settings command from the Group and Outline submenu on the Data menu. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the check box is left in its current state.

Syntax

OUTLINE(auto_styles, row_dir, col_dir, create_apply)

OUTLINE?(auto_styles, row_dir, col_dir, create_apply)

Auto_styles corresponds to the Automatic Styles check box.

Row_dir corresponds to the Summary Rows Below Detail check box.

Col_dir corresponds to the Summary Columns To Right Of Detail check box.

Create_apply is the number 1 or 2 and corresponds to the Create button and the Apply Styles button.

Create_apply	Result
1	Creates an outline with the current settings
2	Applies outlining styles to the selection based on outline levels
Omitted	Corresponds to choosing the OK button to set the other outline settings

Related Functions

DEMOTE Demotes the selection in an outline

PROMOTE Promotes the selection in an outline

List of Command-Equivalent Functions

PARSE

Macro Sheets Only

Distributes the contents of the current selection to fill several adjacent columns; the selection can be no more than one column wide. Use PARSE to reorganize data, especially data that you've read from files created by another application, such as a database.

Syntax

PARSE(parse_text, destination_ref)

PARSE?(parse_text, destination_ref)

Parse_text is the parse line in the form of text. It is a copy of the first nonblank cell in the selected column, with square brackets indicating where to distribute (or parse) text. If parse_text is omitted, Microsoft Excel guesses where to place the brackets based on the spacing and formatting of data.

Destination_ref is a reference to the upper-left corner of the range of cells where you want to place the parsed data. If destination_ref is omitted, it is assumed to be the current selection, so the parsed data will replace the original data.

Remarks

- In most cases, it's easier to record the Parse command than to enter the PARSE function manually.
- When you use the PARSE function, Microsoft Excel splits the first column into as many columns as you specify with parse_text and replaces any information in those columns.

Related Functions

MID Returns a specific string starting at the position you specify

SEARCH Finds one text value within another (not case-sensitive)

List of Command-Equivalent Functions

PLACEMENT

Macro Sheets Only

Equivalent to choosing the Object Placement command from the Format menu in Microsoft Excel version 3.0. Determines how the selected object or objects are attached to the cells beneath them. This function is included only for macro compatibility and will be converted to OBJECT.PROPERTIES when you load older macro sheets. For more information, see OBJECT.PROPERTIES.

Syntax

PLACEMENT(placement_type)

PLACEMENT?(placement_type)

Related Functions

OBJECT.PROPERTIES Determines an object's relationship to underlying cells

List of Command-Equivalent Functions

PROMOTE

Macro Sheets Only

Equivalent to clicking the Ungroup button. Promotes, or ungroups, the currently selected rows or columns in an outline. Use PROMOTE to change the configuration of an outline by promoting rows or columns of information.

Syntax

PROMOTE(rowcol)

PROMOTE?(rowcol)

Rowcol specifies whether to promote rows or columns.

Rowcol	Demotes
1 or omitted	Rows
2	Columns

Remarks

- If the selection consists of an entire row or rows, then rows are promoted even if rowcol is 2. Similarly, selection of an entire column overrides rowcol 1.
- Also, if the selection is unambiguous (an entire row or column), then PROMOTE? will not display the dialog box.

Related Functions

<u>DEMOTE</u>	Demotes the selection in an outline
<u>SHOW.DETAIL</u>	Expands or collapses a portion of an outline
<u>SHOW.LEVELS</u>	Displays a specific number of levels of an outline

RESUME

Macro Sheets Only

Equivalent to choosing the Resume button on the Macro toolbar. Resumes a paused macro. Returns TRUE if successful or the #VALUE! error value if no macro is paused. A macro can be paused by using the PAUSE function or choosing Pause from the Single Step dialog box, which appears when you choose the Step button from the Macro dialog box.

Syntax

RESUME(type_num)

Type_num is a number from 1 to 4 specifying how to resume.

Type_num	How Microsoft Excel resumes
1 or omitted	If paused by a PAUSE function, continues running the macro. If paused from the Single Step dialog box, returns to that dialog box.
2	Halts the paused macro
3	Continues running the macro
4	Opens the Single Step dialog box

Tip You can use Microsoft Excel's ON functions to resume based on an event. For an example, see [ENTER.DATA](#).

Remarks

- If one macro runs a second macro that pauses, and you need to halt only the paused macro, use RESUME(2) instead of HALT. HALT halts all macros and prevents resuming or returning to any macro.
- If the macro was paused from the Single Step dialog box, RESUME returns to the Single Step dialog box.

Related Functions

HALT	Stops all macros from running
PAUSE	Pauses a macro
RETURN	Ends the currently running macro

List of [Command-Equivalent Functions](#)

ROW.HEIGHT

Macro Sheets Only

Equivalent to choosing the Height command on the Row submenu of the Format menu. Changes the height of the rows in a reference.

Syntax

ROW.HEIGHT(height_num, reference, standard_height, type_num)

ROW.HEIGHT?(height_num, reference, standard_height, type_num)

Height_num specifies how high you want the rows to be in points. If standard_height is TRUE, height_num is ignored.

Reference specifies the rows for which you want to change the height.

- If reference is omitted, the reference is assumed to be the current selection.
- If reference is specified, it must be either an external reference to the active worksheet, such as !\$2:\$4 or !Database, or an R1C1-style reference in the form of text or a name, such as "R1:R3", "R[-4]:R[-2]", or Database.
- If reference is a relative R1C1-style reference in the form of text, it is assumed to be relative to the active cell.

Standard_height is a logical value that sets the row height as determined by the font in each row.

- If standard_height is TRUE, Microsoft Excel sets the row height to a standard height that may vary from row to row depending on the fonts used in each row, ignoring height_num.
- If standard_height is FALSE or omitted, Microsoft Excel sets the row height according to height_num.

Type_num is a number from 1 to 3 corresponding to selecting the Hide, Unhide, or AutoFit commands from the Row submenu.

Type_num	Action taken
1	Hides the row selection by setting the row height to 0
2	Unhides the row selection by setting the row height to the value set before the selection was hidden
3	Sets the row selection to an AutoFit height, which varies from row to row depending on how large the font is in any cell in each row or on how many lines of text are wrapped

Remarks

- If any of the argument settings conflict, such as when standard_height is TRUE and type_num is 3, Microsoft Excel uses the type_num argument and ignores any arguments that conflict with type_num.
- If you are recording a macro while using a mouse, and you change row heights by dragging the row border, Microsoft Excel records the reference of the rows using R1C1-style references in the form of text. If Uses Relative References is selected, Microsoft Excel uses R1C1-style relative references. If Uses Relative References is not selected, Microsoft Excel uses R1C1-style absolute references.

Related Function

COLUMN.WIDTH Sets the widths of the specified columns

List of Command-Equivalent Functions

RUN

Macro Sheets Only

Equivalent to choosing the Run button in the Macro dialog box, which appears when you choose the Macro command from the Tools menu. Runs a macro.

Syntax

RUN(reference, step)

RUN?(reference, step)

Reference is a reference to the macro you want to run or a number from 1 to 4 specifying an Auto macro to run.

If reference is	Specifies
1	All Auto_Open macros on the active workbook
2	All Auto_Close macros
3	All Auto_Activate macros
4	All Auto_Deactivate macros

- If reference is a range of cells, RUN begins with the macro function in the upper-left cell of reference.
- If the macro sheet containing the macro is not the active document, reference can be an external reference to the name of the macro, such as RUN([BOOK1]Macro!Months) or an external R1C1-style reference to the location of the macro, such as RUN("[Book1]Macro!R2C3"). The reference must be in text form.
- If reference is omitted, the macro function in the active cell is carried out, and macro execution continues down that column.

Step is a logical value specifying that the macro is to be run in single-step mode. If step is TRUE, Microsoft Excel runs the macro in single-step mode; if FALSE or omitted, Microsoft Excel runs the macro normally.

Remarks

- RUN is recorded when you choose the Run command from the Macro menu while recording a macro. The reference you enter in the Run dialog box is recorded as text, with A1-style references converted to R1C1-style references.
- To run a macro from a macro sheet, you could alternatively enter the name of the macro as a formula, followed by a set of parentheses. For example, enter =[Book1]Macro!Months() instead of =RUN([Book1]Macro!Months).

Related Functions

GOTO Directs macro execution to another cell
List of Command-Equivalent Functions

SAVE

Macro Sheets Only

Equivalent to choosing the Save command from the File menu. Saves the active document.

Syntax

SAVE()

Remarks

Use the SAVE.AS function instead of SAVE when you want to change the filename or file type, specify a password, create a backup file, or save a file to a different directory or folder.

Related Functions

[SAVE.AS](#)

Saves a document and allows you to specify the name, file type, password, backup file, and location of the document

[SAVE.WORKBOOK](#)

Saves a workbook

List of [Command-Equivalent Functions](#)

SELECT.END

Macro Sheets Only

Selects the cell at the edge of the range in the direction specified. Equivalent to pressing CTRL+ARROW in Microsoft Excel for Windows or COMMAND+ARROW in Microsoft Excel for the Macintosh.

Syntax

SELECT.END(direction_num)

Direction_num is a number from 1 to 4 indicating the direction in which to move.

Direction_num	Direction
1	Left (equivalent to CTRL+LEFT ARROW or COMMAND+LEFT ARROW)
2	Right (equivalent to CTRL+RIGHT ARROW or COMMAND+RIGHT ARROW)
3	Up (equivalent to CTRL+UP ARROW or COMMAND+UP ARROW)
4	Down (equivalent to CTRL+DOWN ARROW or COMMAND+DOWN ARROW)

Related Function

SELECT.LAST.CELL

Selects the last cell on a worksheet or macro sheet that contains a formula, value, or format or that is referred to in a formula or name

List of Command-Equivalent Functions

SELECT.LAST.CELL

Macro Sheets Only

Equivalent to choosing the Special button from the Goto dialog box and selecting the Last Cell Option. The Goto dialog box appears when you choose the Goto command from the Edit menu. Selects the cell at the intersection of the last row and column that contains a formula, value, or format, or that is referred to in a formula or name.

Syntax

SELECT.LAST.CELL()

Related Function

SELECT.END Selects the last cell in a range

List of Command-Equivalent Functions

SHOW.DETAIL

Macro Sheets Only

Expands or collapses the detail under the specified expand or collapse button.

Syntax

SHOW.DETAIL(rowcol, rowcol_num, expand, show_field)

Rowcol is a number that specifies whether to operate on rows or columns of data.

Rowcol	Operates on
1	Rows
2	Columns
3	The current cell's row or column. The second argument, rowcol_num, is then ignored.

Rowcol_num is a number that specifies the row or column to expand or collapse. If you are in A1 mode, you must still give the column as a number. If rowcol_num is not a summary row or column, SHOW.DETAIL returns the #VALUE! error value and interrupts the macro.

Expand is a logical value that specifies whether to expand or collapse the detail under the row or column. If expand is TRUE, Microsoft Excel expands the detail under the row or column; if FALSE, it collapses the detail under the row or column. If expand is omitted, the detail is expanded if it is currently collapsed and collapsed if it is currently expanded.

Show_Field is a string specifying the name of the field to add to a PivotTable, if the selection is inside a PivotTable. The new field is added as the new innermost field. Available for only innermost row or column fields.

Related Function

[SHOW.LEVELS](#) Displays a specific number of levels of an outline

List of [Command-Equivalent Functions](#)

SHOW.LEVELS

Macro Sheets Only

Displays the specified number of row and column levels of an outline.

Syntax

SHOW.LEVELS(row_level, col_level)

Row_level specifies the number of row levels of an outline to display. If the outline has fewer levels than specified by row_level, Microsoft Excel shows all levels. If row_level is zero or omitted, no action is taken on rows.

Col_level specifies the number of column levels of an outline to display. If the outline has fewer levels than specified by col_level, Microsoft Excel shows all levels. If col_level is zero or omitted, no action is taken on columns.

Remarks

If you omit both arguments, SHOW.LEVELS returns the #VALUE! error value.

Related Function

SHOW.DETAIL Expands or collapses a portion of an outline

List of Command-Equivalent Functions

SPLIT

Macro Sheets Only

Equivalent to choosing the Split command from the Window menu or to dragging the split bar in the active window's scroll bar. Splits the active window into panes. Use SPLIT when you want to view different parts of the active document at the same time.

Syntax

SPLIT(col_split, row_split)

Col_split specifies where to split the window vertically and is measured in columns from the left of the window.

Row_split specifies where to split the window horizontally and is measured in rows from the top of the window.

If an argument is 0 and there is a split in that direction, the split is removed. If an argument is omitted, a split in that direction is not changed.

Related Function

[FREEZE.PANES](#) Freezes or unfreezes the panes of a window

List of [Command-Equivalent Functions](#)

STYLE

Macro Sheets Only

Checks the fonts for a bold and/or italic font and applies it to the current selection in Microsoft Excel for the Macintosh version 1.5 or earlier. If no appropriate font is available, Microsoft Excel finds the most similar font available and formats the selection using that font. This function is included only for macro compatibility. If you want to change a font to bold or italic, use the FONT.PROPERTIES function.

Syntax

STYLE(bold,italic)

STYLE?(bold,italic)

Related Functions

FONT.PROPERTIES Applies a font to the selection

List of Command-Equivalent Functions

UNGROUP

Macro Sheets Only

Equivalent to choosing the Ungroup command from the Placement submenu on the Format menu. Separates a grouped object into individual objects. Use UNGROUP to separate a group of objects so that you can format, move, or size one of the objects.

If the selection is not a grouped object, UNGROUP returns FALSE.

Syntax

UNGROUP()

Related Function

GROUP Groups selected objects

List of Command-Equivalent Functions

VLIN

Macro Sheets Only

Scrolls through the active window vertically by the number of rows you specify.

Syntax

VLIN(num_rows)

Num_rows is a number that specifies how many rows to scroll.

- If num_rows is positive, Microsoft Excel scrolls down by the number of rows indicated by num_rows.
- If num_rows is negative, Microsoft Excel scrolls up by the number of rows indicated by num_rows.

Related Functions

<u>HLIN</u>	Horizontally scrolls through the active window by columns
<u>HPAGE</u>	Horizontally scrolls through the active window one window at a time
<u>HSCROLL</u>	Horizontally scrolls through a document by percentage or by column number
<u>VPAGE</u>	Vertically scrolls through the active window one window at a time
<u>VSCROLL</u>	Vertically scrolls through a document by percentage or by row number

List of Command-Equivalent Functions

VPAGE

Macro Sheets Only

Vertically scrolls through the active window one window at a time. Use VPAGE to change the displayed area of a worksheet or macro sheet.

Syntax

VPAGE(num_windows)

Num_windows specifies the number of windows to scroll through the active window vertically. A window is defined as the number of visible rows. If 20 rows are visible in the window, VPAGE scrolls in increments of 20 rows.

- If num_windows is positive, VPAGE scrolls down.
- If num_windows is negative, VPAGE scrolls up.

Related Functions

<u>HPAGE</u>	Horizontally scrolls through the active window one window at a time
<u>HSCROLL</u>	Horizontally scrolls through a document by percentage or by column number
<u>VLIN</u>	Vertically scrolls through the active window by rows
<u>VSCROLL</u>	Vertically scrolls through a document by percentage or by row number
List of <u>Command-Equivalent Functions</u>	

ZOOM

Macro Sheets Only

Equivalent to choosing the Zoom command from the View menu. Enlarges or reduces a document in the active window. Use ZOOM when you need to view more cells than would normally fit in the active windows, or fewer cells at a larger size.

Syntax

ZOOM(magnification)

Magnification is a logical value or a number specifying the size of the document.

- Magnification can be a number from 10 to 400 specifying the percentage of enlargement or reduction.
- If magnification is TRUE or omitted, the current selection is enlarged or reduced to completely fill the active window.
- If magnification is FALSE, the document is restored to normal 100% magnification.

Related Function

PRINT.PREVIEW Previews pages and pagebreaks before printing.

List of Command-Equivalent Functions

ACTIVATE

Macro Sheets Only

Switches to a window if more than one window is open, or a pane of a window if the window is split and its panes are not frozen. Switching to a pane is useful with functions such as VSCROLL, HSCROLL, and GOTO, which operate only on the active pane.

Syntax

ACTIVATE(window_text, pane_num)

ACTIVATE?(window_text, pane_num)

Window_text is text specifying the name of a window to switch to: for example, "Book1" or "Book1:2".

- If a workbook is displayed in more than one window and window_text does not specify which window to switch to, the first window containing that workbook is switched to.
- If window_text is omitted, the active window is not changed.

Pane_num is a number from 1 to 4 specifying which pane to switch to. If pane_num is omitted and the window has more than one pane, the active pane is not changed.

Pane_num	Activates
1	Upper-left pane of the active sheet. If the window is not split, this is the only pane. If the window is split only horizontally, this is the upper pane. If the window is split only vertically, this is the left pane.
2	Upper-right pane of the active sheet. If the window is split only vertically, this is the right pane. If the window is split only horizontally, an error occurs.
3	Lower-left pane of the active sheet. If the window is split only horizontally, this is the lower pane. If the window is split only vertically, an error occurs.
4	Lower-right pane of the active sheet. If the window is split into only two panes either vertically or horizontally, an error occurs.

Related Functions

ACTIVATE.NEXT

Switches to the next window, or switches to the next sheet in a workbook

ACTIVATE.PREV

Switches to the previous window, or switches to the previous sheet in a workbook

DOCUMENTS

Returns the names of the specified open workbooks

FREEZE.PANES

Freezes the panes of a window so that they do not scroll

ON.WINDOW

Runs a macro when you switch to a window

SPLIT

Splits a window

WINDOWS

Returns the names of all open windows

WORKBOOK.SELECT

Select a sheet in a workbook

List of Command-Equivalent Functions

ACTIVATE.NEXT

ACTIVATE.PREV

Macro Sheets Only

Switches to the next or previous window, respectively, or switches to the next or previous sheet in a workbook.

Syntax

ACTIVATE.NEXT(workbook_text)

ACTIVATE.PREV(workbook_text)

Workbook_text is the name of the workbook for which you want to activate a window.

- If workbook_text is specified, ACTIVATE.NEXT and ACTIVATE.PREV are equivalent to pressing CTRL+PAGE DOWN and CTRL+PAGE UP (in Microsoft Excel for Windows) or COMMAND+PAGE DOWN and COMMAND+PAGE UP (in Microsoft Excel for the Macintosh). These functions switch to the next and previous sheets, respectively.
- If workbook_text is omitted, these functions are equivalent to pressing CTRL+TAB or CTRL+SHIFT+TAB (in Microsoft Excel for Windows) or COMMAND+M or COMMAND+SHIFT+M (in Microsoft Excel for the Macintosh). These functions switch to the next and previous windows, respectively.

Related Functions

ACTIVATE

Switches to a window

ON.WINDOW

Runs a macro when you switch to a window

WORKBOOK.NEXT

Switches to the next sheet in a workbook

WORKBOOK.PREV

Switches to the previous sheet in a workbook

WORKBOOK.SELECT

Select a sheet in a workbook

List of Command-Equivalent Functions

ADD.BAR

Macro Sheets Only

Creates a new menu bar and returns the bar ID number. Use the bar ID number to identify the menu in functions that display and add menus and commands to the menu bar. You can also use ADD.BAR to restore a built-in menu bar with its original menus and commands.

Syntax

ADD.BAR(bar_num)

Bar_num is the number of a built-in menu bar that you want to restore. Use ADD.BAR(bar_num) to restore an unaltered version of a built-in menu bar after you have made changes to the menu bar's menus and commands. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

Important Restoring a built-in menu bar will remove menus and commands added by other macros. Use ADD.COMMAND and ADD.MENU to restore individual commands and menus.

Remarks

ADD.BAR just creates a new menu bar; it does not display it. Use SHOW.BAR to display a menu bar. The argument to the SHOW.BAR function should be the number returned by ADD.BAR or a reference to the cell containing ADD.BAR.

You can define up to 15 custom menu bars at one time. If you carry out an ADD.BAR function when more than 15 custom menu bars are already defined, Microsoft Excel returns the #VALUE! error value.

Example

The following formula creates a new menu bar and returns a bar ID number:

ADD.BAR ()

Related Functions

ADD.COMMAND Adds a command to a menu

ADD.MENU Adds a menu to a menu bar

DELETE.BAR Deletes a menu bar

SHOW.BAR Displays a menu bar

List of Customizing Functions

ADD.COMMAND

Macro Sheets Only

Adds a command to a menu. ADD.COMMAND returns the position number on the menu of the added command. Use ADD.COMMAND to add one or more custom menu commands to a menu on a built-in or custom menu bar. You can also use ADD.COMMAND to restore a deleted built-in command to its original menu.

Syntax

ADD.COMMAND(bar_num, menu, command_ref, position1, position2)

Bar_num is the number corresponding to a menu bar or a type of shortcut menu to which you want to add a command.

- Bar_num can be the ID number of a built-in or custom menu bar. The ID number of a custom menu bar is the number returned by the ADD.BAR function.
- Bar_num can also refer to a type of shortcut menu; use menu to identify the specific shortcut menu.

The ID numbers of the built-in menu bars and the types of shortcut menus are listed in the following tables. Short menus are abbreviated versions of the normal Microsoft Excel menus. To turn on short menus, use the SHORT.MENUS function.

Bar_num	Built-in menu bar
1	Worksheet and macro sheet (Excel 4.0)
2	Chart (Excel 4.0)
3	Null (the menu displayed when no workbooks are open)
4	Info
5	Worksheet and macro sheet (short menus, Excel 3.0 and earlier)
6	Chart (short menus, Excel 3.0 and earlier)
7	Cell, toolbar, and workbook (shortcut menus)
8	Object (shortcut menus)
9	Chart (Excel 4.0 shortcut menus)
10	Worksheet and Macro sheet
11	Chart
12	Visual Basic

Menu is the menu to which you want the new command added.

- Menu can be either the name of a menu as text or the number of a menu.
- If bar_num is 1 through 6, menus are numbered starting with 1 from the left of the menu bar.
- If bar_num is 7, 8, or 9, menu refers to a built-in shortcut menu. The combination of bar_num and menu determines which shortcut menu to modify, as shown in the following table.

Bar_num	Menu	Shortcut menu
7	1	Toolbars
7	2	Toolbar buttons
7	3	Workbook paging icons in Excel 4.0
7	4	Cells (worksheet)
7	5	Column selections
7	6	Row selections
7	7	Workbook tabs
7	8	Cells (macro sheet)
7	9	Workbook titlebar
7	10	Desktop (Microsoft Excel for Windows only)
7	11	Module

7	12	Watch pane
7	13	Immediate pane
7	14	Debug code pane
8	1	Drawn or imported objects on worksheets, dialog sheets, and charts.
8	2	Buttons on sheets
8	3	Text boxes
8	4	Dialog sheet
9	1	Chart series
9	2	Chart and Axis titles
9	3	Chart plot area and walls
9	4	Entire chart
9	5	Chart axes
9	6	Chart gridlines
9	7	Chart floor and arrows
9	8	Chart legend

Note Any commands that you add to the toolbar buttons, watch pane, immediate pane or debug code pane shortcut menus will be dimmed.

Command_ref is an array or a reference to an area on the macro sheet that describes the new command or commands.

- **Command_ref** must be at least two columns wide. The first column specifies command names; the second specifies macro names. Optional columns can be specified for shortcut keys (in Microsoft Excel for the Macintosh), status bar messages, and custom Help topics, in that order.
- **Command_ref** is similar to **menu_ref** in **ADD.MENU**. For more information about **command_ref**, see the description of **menu_ref** in **ADD.MENU**.
- **Command_ref** can be the name, as text, of a previously deleted built-in command that you want to restore. You can also use the value returned by the **DELETE.COMMAND** formula that deleted the command.

Position1 specifies the placement of the new command.

- Use a hyphen (-) to represent a line separating commands on a menu. If you want to place a command before the second separator on a menu, use two hyphens (--), three hyphens for the third separator, and so on.
- **Position1** can be a number indicating the position of the command on the menu. Commands are numbered from the top of the menu starting with 1.
- **Position1** can be the name of an existing command, as text, above which you want to add the new command.
- If **position1** is omitted, the command is added to the bottom of the menu.
- For the toolbar shortcut menu (**bar_num** 7, menu 1) and the shortcut menu for workbook paging icons in Microsoft Excel version 4.0(**bar_num** 7, menu 3), you cannot add commands to the middle of the toolbar name list or the middle of the workbook contents list.

Position2 specifies the placement of the new command on a submenu.

- **Position2** can be a number indicating the position of the command on the submenu. Commands are numbered from the top of the menu starting with 1.
- **Position2** can be the name of an existing command, as text, above which you want to add the new command.
- If **position2** is omitted, the command is added to the main menu, not the submenu.
- To add a command to the bottom of a submenu, use 0 for **position2**.

Tip In general, use menu and command names rather than numbers for arguments. The numbers assigned to menus and commands change as you add and delete menus and commands. Using names ensures that your menu and command macro functions always refer to the correct items.

Example

The following macro formula adds the command described in cells G16:J16 to the bottom of the worksheet cells shortcut menu:

```
ADD.COMMAND(7, 4, G16:J16)
```

Related Functions

ADD.BAR

Adds a menu bar

ADD.MENU

Adds a menu to a menu bar

ADD.TOOL

Adds one or more buttons to a toolbar

ADD.TOOLBAR

Creates a toolbar with the specified tools

DELETE.COMMAND

Deletes a command from a menu

ENABLE.COMMAND

Enables or disables a menu or custom command

GET.TOOLBAR

Retrieves information about a toolbar

RENAME.COMMAND

Changes the name of a command or menu

List of Customizing Functions

ADD.MENU

Macro Sheets Only

Adds a menu to a menu bar. Use ADD.MENU to add a custom menu to a built-in or custom menu bar. You can also use ADD.MENU to restore built-in menus you have deleted with DELETE.MENU.

ADD.MENU returns the position number in the menu bar of the new menu.

Syntax

ADD.MENU(bar_num, menu_ref, position1, position2)

Bar_num is the menu bar to which you want a menu added. Bar_num can be the ID number of a built-in or custom menu bar. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

Menu_ref is an array or a reference to an area on the macro sheet that describes the new menu or the name of a deleted built-in menu you want to restore.

- Menu_ref must be made up of at least two rows and two columns of cells. The upper-left cell of menu_ref specifies the menu title, which is displayed in the menu bar. In the following example, the range A3:E10 is a valid menu_ref.

	A	B	C	D	E
1	Menu or Command Name	Macro Name	Shortcut key	Status bar text	Help topic
2	Macintosh only				
3	Reports				
4	Weekly Report	WeeklyRept		Prints weekly report	Help!35
5	Monthly Report	MonthlyRept		Prints monthly report	Help!36
6	Quarterly Report	QuartRept		Prints quarterly report	Help!37
7	-				
8	Custom Report	CustomRept		Create a custom report	Help!38
9	-				
10	Remove Menu	RemoveMenu		Removes Reports menu	Help!39

- The rest of the first column indicates the names of the commands. The corresponding rows in the second column give the names of the macros that run when the commands are chosen.

- You can also specify status-bar text and Help topics in the fourth and fifth columns of menu_ref. In Microsoft Excel for the Macintosh, you can specify shortcut keys in the third column of menu_ref.

Position1 specifies the placement of the new menu. Position can be the name of a menu, as text, or the number of a menu. Menus are numbered from left to right starting with 1. Menus are added to the left of the position specified.

- Use a hyphen (-) to represent a line separating commands on a menu. If you want to place a command before the second separator on a menu, use two hyphens (--), three hyphens for the third separator, and so on.
- If position1 is omitted, the menu is added to the end of the menu bar.
- If there is already a menu at position1, that menu is shifted to the right and the new menu is added in its place.
- If you are using ADD.MENU to restore a deleted built-in menu, you can use the position argument to put it back in its original place on the menu bar. For example, to restore the Data menu on the worksheet and macro sheet menu bar, use position 7. If position1 is omitted, the menu is added to the right of the last menu restored.

Position2 specifies the placement of a submenu.

- Use a hyphen (-) to represent a line separating commands on a menu. If you want to place a command before the second separator on a menu, use two hyphens (--), three hyphens for the third separator, and so on.
- Position2 can be a number indicating the position of the submenu on the menu. Commands are numbered from the top of the menu starting with 1 and include separators.
- Position2 can also be the name, as text, of an existing command above which you want to add the new command.
- If position2 is omitted, the command is added to the main menu, not the submenu.

Example

The following macro formula adds a new menu to the end of the worksheet menu bar, where A10:B15 is the menu_ref describing the menu:

```
ADD.MENU (1, A10:B15)
```

Related Functions

[ADD.BAR](#)

Adds a menu bar

[ADD.COMMAND](#)

Adds a command to a menu

[DELETE.MENU](#)

Deletes a menu

[ENABLE.COMMAND](#)

Enables or disables a menu or custom command

List of [Customizing Functions](#)

ADD.TOOLBAR

Macro Sheets Only

Creates a new toolbar with the specified buttons. For more information about customizing toolbars, see [Overview of customizing toolbars](#).

Syntax

ADD.TOOLBAR(bar_name, tool_ref)

Bar_name is a text string identifying the toolbar you want to create.

Tool_ref is either a number specifying a built-in button or a reference to an area on the macro sheet that defines a custom button or set of buttons (or an array containing this information). For a complete list of built-in buttons and their corresponding numbers, see Appendix D, "Toolbar Buttons in Microsoft Excel," in the Microsoft Excel Visual Basic User's Guide.

For a complete description of tool_ref, see ADD.TOOL.

Remarks

If you create a toolbar without buttons, use ADD.TOOL to add them. Use SHOW.TOOLBAR to display the toolbar.

Example

The following macro formula creates Toolbar9 with one button in it. The cell range B7:I7 contains tool_ref.

```
ADD.TOOLBAR("Toolbar9", B7:I7)
```

Related Functions

[ADD.TOOL](#)

Adds a button to a toolbar

[DELETE.TOOL](#)

Deletes a button from a toolbar

[DELETE.TOOLBAR](#)

Deletes custom toolbars

[RESET.TOOLBAR](#)

Resets a built-in toolbar to its default initial setting

[SHOW.TOOLBAR](#)

Hides or displays a toolbar

List of [Customizing Functions](#)

ALIGNMENT

Macro Sheets Only

Equivalent to choosing the Alignment tab in the Format Cells dialog box, which is displayed when you choose the Cells command from the Format menu. Aligns the contents of the selected cells.

Syntax

ALIGNMENT(horiz_align, wrap, vert_align, orientation, add_indent)

ALIGNMENT?(horiz_align, wrap, vert_align, orientation, add_indent)

Horiz_align is a number from 1 to 7 specifying the type of horizontal alignment, as shown in the following table. If horiz_align is omitted, horizontal alignment does not change.

Horiz_align	Horizontal alignment
1	General
2	Left
3	Center
4	Right
5	Fill
6	Justify
7	Center across selection

Wrap is a logical value corresponding to the Wrap Text check box in the Alignment tab. If wrap is TRUE, Microsoft Excel selects the check box and wraps text in cells; if FALSE, Microsoft Excel clears the check box and does not wrap text. If wrap is omitted, wrapping does not change.

Vert_align is a number from 1 to 4 specifying the vertical alignment of the text. If vert_align is omitted, vertical alignment does not change.

Vert_align	Vertical alignment
1	Top
2	Center
3	Bottom
4	Justify

Orientation is a number from 0 to 4 specifying the orientation of the text. If orientation is omitted, text orientation does not change.

Orientation	Text orientation
0	Horizontal
1	Vertical
2	Upward
3	Downward
4	Automatic (applies to only chart tick labels)

Add_indent This argument is for only Far East versions of Microsoft Excel.

Related Function

[FORMAT.TEXT](#) Formats a worksheet text box or a chart text item

List of [Command-Equivalent Functions](#)

APP.MAXIMIZE

Macro Sheets Only

Equivalent to choosing the Maximize command from the Control menu for the application window.
Maximizes the Microsoft Excel window.

Syntax

APP.MAXIMIZE()

Note This function is only for Microsoft Excel for Windows. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>APP.MINIMIZE</u>	Minimizes the Microsoft Excel application window
<u>APP.MOVE</u>	Moves the Microsoft Excel application window
<u>APP.RESTORE</u>	Restores the Microsoft Excel application window
<u>APP.SIZE</u>	Changes the size of the Microsoft Excel application window
<u>FULL.SCREEN</u>	Controls full screen display
List of <u>Command-Equivalent Functions</u>	

APP.MINIMIZE

Macro Sheets Only

Equivalent to choosing the Minimize command from the Control menu for the application window.
Minimizes the Microsoft Excel window.

Syntax

APP.MINIMIZE()

Note This function is only for Microsoft Excel for Windows. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>APP.MAXIMIZE</u>	Maximizes the Microsoft Excel application window
<u>APP.MOVE</u>	Moves the Microsoft Excel application window
<u>APP.RESTORE</u>	Restores the Microsoft Excel application window
<u>APP.SIZE</u>	Changes the size of the Microsoft Excel application window
List of <u>Command-Equivalent Functions</u>	

APP.MOVE

Macro Sheets Only

Equivalent to choosing the Move command from the Control menu for the application window. Moves the Microsoft Excel window. In Microsoft Excel for Windows, if the application window is already maximized APP.MOVE returns the #VALUE! error value and interrupts the macro.

Syntax

APP.MOVE(x_num, y_num)

APP.MOVE?(x_num, y_num)

Note This function is only for Microsoft Excel for Windows. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

X_num specifies the horizontal position of the Microsoft Excel window measured in points from the left edge of your screen to the left side of the Microsoft Excel window.

Y_num specifies the vertical position of the Microsoft Excel window measured in points from the top edge of your screen to the top of the Microsoft Excel window.

Remarks

- APP.MOVE?, the dialog-box form of the function, doesn't display a dialog box. Instead, it is equivalent to pressing ALT + SPACEBAR, M or to dragging the title bar with the mouse. With APP.MOVE?, you can move the window with the keyboard or mouse.
- If you specify x_num and/or y_num in the dialog-box form of the function, the window is moved according to the specified coordinates, and you are left in move mode.

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>APP.MAXIMIZE</u>	Maximizes the Microsoft Excel application window
<u>APP.MINIMIZE</u>	Minimizes the Microsoft Excel application window
<u>APP.RESTORE</u>	Restores the Microsoft Excel application window
<u>APP.SIZE</u>	Changes the size of the Microsoft Excel application window

List of Command-Equivalent Functions

APP.RESTORE

Macro Sheets Only

Equivalent to choosing the Restore command from the Control menu for the application window.
Restores the Microsoft Excel window to its previous size and location.

Syntax

APP.RESTORE()

Note This function is only for Microsoft Excel for Windows. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>APP.MAXIMIZE</u>	Maximizes the Microsoft Excel application window
<u>APP.MINIMIZE</u>	Minimizes the Microsoft Excel application window
<u>APP.MOVE</u>	Moves the Microsoft Excel application window
<u>APP.SIZE</u>	Changes the size of the Microsoft Excel application window
List of <u>Command-Equivalent Functions</u>	

APP.TITLE

Macro Sheets Only

Changes the title of the Microsoft Excel application workspace to the title you specify. The title appears at the top of the application window and in the Microsoft Windows Task List. Use APP.TITLE to control the application title when you're using Microsoft Excel to create a custom application. This function does not affect Microsoft Excel for the Macintosh.

Syntax

APP.TITLE(text)

Text is the title you want to assign to the Microsoft Excel application workspace. If text is omitted, it is restored to Microsoft Excel.

Remarks

- The custom application title, followed by the individual workbook title, will appear in the application title bar if the workbook is maximized.
- APP.TITLE does not affect DDE communications. You will still refer to the application as "Excel".

Related Function

WINDOW.TITLE Changes the title of the active window

List of Customizing Functions

APPLY.STYLE

Macro Sheets Only

Equivalent to choosing the Style command from the Format menu, selecting a style, and choosing the OK button. Applies a previously defined style to the current selection.

Syntax

APPLY.STYLE(style_text)

APPLY.STYLE?(style_text)

Style_text is the name, as text, of a previously defined style. If style_text is not defined, APPLY.STYLE returns the #VALUE! error value and interrupts the macro. If style_text is omitted, the Normal style is applied to the selection.

Related Functions

DEFINE.STYLE Defines a cell style

DELETE.STYLE Deletes a cell style

MERGE.STYLES Imports styles from another workbook into the active workbook

List of Command-Equivalent Functions

ARRANGE.ALL

Macro Sheets Only

Equivalent to choosing the Arrange command from the Window menu. Rearranges open windows and icons and resizes open windows. Also can be used to synchronize scrolling of windows of the active sheet.

Syntax

ARRANGE.ALL(arrange_num, active_doc, sync_horiz, sync_vert)

ARRANGE.ALL?(arrange_num, active_doc, sync_horiz, sync_vert)

Arrange_num is a number from 1 to 7 specifying how to arrange the windows.

Arrange_num	Result
1 or omitted	Tiled (also used to arrange icons in Microsoft Excel for Windows)
2	Horizontal
3	Vertical
4	None
5	Horizontally arranges and sizes the windows based on the position of the active cell.
6	Vertically arranges and sizes the windows based on the position of the active cell.
7	Arranges windows so that they cascade from the upper left to the bottom right of the application workspace.

If you want to change whether the windows are synchronized for scrolling but not how they are arranged, make sure arrange_num is 4.

Active_doc is a logical value specifying which windows to arrange. If active_doc is TRUE, Microsoft Excel arranges only windows on the active workbook; if FALSE or omitted, all open windows are arranged.

Sync_horiz is a logical value corresponding to the Sync Horizontal check box in Microsoft Excel version 4.0.

- If sync_horiz is TRUE, Microsoft Excel selects the check box and synchronizes horizontal scrolling.
- If sync_horiz is FALSE or omitted, Microsoft Excel clears the check box, and windows will not be synchronized when you scroll horizontally.
- This argument is used only when active_doc is TRUE.

Sync_vert is a logical value corresponding to the Sync Vertical check box in Microsoft Excel version 4.0.

- If sync_vert is TRUE, Microsoft Excel selects the check box and synchronizes vertical scrolling.
- If sync_vert is FALSE or omitted, Microsoft Excel clears the check box, and windows will not be synchronized when you scroll vertically.
- This argument is used only when active_doc is TRUE.

Note If arguments are omitted in the dialog box form of this function, the default values are the previous settings, if any; otherwise the default values are as described above.

Remarks

- After arranging windows, the top or leftmost window is active.
- In Microsoft Excel for Windows, if all windows are minimized, ARRANGE.ALL ignores its arguments, if any, and arranges the corresponding icons horizontally along the bottom of the workspace.

Tip You can use synchronized horizontal or vertical scrolling when you need to scroll while viewing macro formulas in one window and corresponding macro values in another window of the same macro sheet.

Related Function

ACTIVATE

Switches to a window

List of Command-Equivalent Functions

ASSIGN.TO.OBJECT

Macro Sheets Only

Assigns a macro to the currently select object.

Syntax

ASSIGN.TO.OBJECT(macro_ref)

ASSIGN.TO.OBJECT?(macro_ref)

Macro_ref is the name of, or a reference to, the macro you want to run when the object is clicked. If macro_ref is omitted, Microsoft Excel no longer runs the previously specified macro (ASSIGN.TO.OBJECT is turned off).

Remarks

- If an object is not selected, ASSIGN.TO.OBJECT returns the #VALUE! error value and interrupts the macro.
- To change the macro assigned to an object, select the object and use ASSIGN.TO.OBJECT again, using the reference to the new macro as macro_ref. The previous macro is replaced with the new macro.

Related Functions

CREATE.OBJECT Creates an object

RUN Runs a macro

List of Command-Equivalent Functions

ASSIGN.TO.TOOL

Macro Sheets Only

Assigns a macro to be run when a button is clicked with the mouse.

Syntax

ASSIGN.TO.TOOL(**bar_id**, **position**, macro_ref)

Bar_id specifies the number or name of a toolbar to which you want to assign a macro. For more information about bar_id, see ADD.TOOL.

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Macro_ref is the name of, or a reference to, the macro you want to run when the button is clicked. If macro_ref is omitted, Microsoft Excel no longer runs the previously specified macro. After canceling the macro, if the button is a built-in button, Microsoft Excel performs the normal default action when the button is clicked. If the button is a custom button, Microsoft Excel displays the Assign Macro dialog box when the button is clicked.

Related Functions

ADD.TOOL Adds one or more buttons to a toolbar

GET.TOOL Returns information about a button or buttons on a toolbar

List of Command-Equivalent Functions

ATTACH.TEXT

Macro Sheets Only

Equivalent to choosing the Titles command from the Insert menu available when a chart is displayed in the active window. Attaches text to certain parts of the selected chart. Use ATTACH.TEXT to attach text as a title or as a label for an axis or data point.

Syntax

ATTACH.TEXT(attach_to_num, series_num, point_num)

ATTACH.TEXT?(attach_to_num, series_num, point_num)

Attach_to_num specifies which item on a chart to attach text to. Attach_to_num is different for 2-D and 3-D charts. Attach_to_num values for 2-D charts are shown in the following table.

Attach_to_num	Attaches text to
1	Chart title
2	Value (y) axis
3	Category (x) axis
4	Series and data point
5	Secondary value (y) axis
6	Secondary category (x) axis

Attach_to_num values for 3-D charts are shown in the following table.

Attach_to_num	Attaches text to
1	Chart title
2	Value (z) axis
3	Series (y) axis
4	Category (x) axis
5	Series and data point

Series_num specifies the series number if attach_to_num specifies a series or data point. If attach_to_num specifies a series or data point and series_num is omitted, the macro is interrupted.

Point_num specifies the number of the data point, but only if you specify a series number. Point_num is required if series_num is specified, unless the chart is an area chart.

Remarks

When you record adding an axis title or a chart title, Microsoft Excel records both an ATTACH.TEXT function to attach the text and a FONT.PROPERTIES function to make the text bold.

Example

The following macro functions attach the text "Quarterly Sales" to the X (category) axis of the selected chart:

```
ATTACH.TEXT(3)
```

```
FORMULA("Quarterly Sales")
```

Related Functions

DATA.LABEL Assigns text labels to point on a chart

FORMULA Enters values into a cell or range or onto a chart

List of Command-Equivalent Functions

AXES

Macro Sheets Only

Equivalent to choosing the Axes command from the Insert menu available when a chart is displayed in the active window. Controls whether the axes on a chart are visible. There are two syntax forms of this function. Syntax 1 is for 2-D charts; syntax 2 is for 3-D charts.

Syntax 1

For 2-D charts

AXES(x_primary, y_primary, x_secondary, y_secondary)

AXES?(x_primary, y_primary, x_secondary, y_secondary)

Syntax 2

For 3-D charts

AXES(x_primary, y_primary, z_primary)

AXES?(x_primary, y_primary, z_primary)

Arguments are logical values corresponding to the check boxes in the Axes dialog box.

- If an argument is TRUE, Microsoft Excel selects the check box and displays the corresponding axis.
- If an argument is FALSE, Microsoft Excel clears the check box and hides the corresponding axis.
- If an argument is omitted, the display of that axis is unchanged.

X_primary corresponds to the primary category (x) axis.

Y_primary corresponds to the primary value (y) axis.

Z_primary corresponds to the value (z) axis on the primary 3-D chart.

X_secondary corresponds to the secondary category (x) axis on a 2-D chart only.

Y_secondary corresponds to the secondary value (y) axis on a 2-D chart only.

If a 2-D chart has no secondary axis, only the first two arguments are used.

Related Function

GRIDLINES Controls whether chart gridlines are visible

List of Command-Equivalent Functions

BRING.TO.FRONT

Macro Sheets Only

Equivalent to choosing the Bring To Front command from the Placement submenu on the Format menu. Puts the selected object or objects on top of all other objects. For example, if some worksheet objects are covering part of an embedded chart, you can select the chart and use BRING.TO.FRONT to display the chart on top of the worksheet objects.

Syntax

BRING.TO.FRONT()

If the selection is not an object or a group of objects, BRING.TO.FRONT returns the #VALUE! error value.

Related Function

SEND.TO.BACK Sends selected objects behind other objects

List of Command-Equivalent Functions

CALCULATE.NOW

Macro Sheets Only

Equivalent to choosing the Calculation tab from the Options dialog box and then choosing the Calc Now button. Calculates all sheets in all open workbooks. Use CALCULATE.NOW to calculate all open workbooks when calculation is set to manual.

Syntax

CALCULATE.NOW()

Related Functions

CALCULATE.DOCUMENT

Calculates the active sheet only

CALCULATION

Controls calculation settings

List of Command-Equivalent Functions

CALCULATION

Macro Sheets Only

Controls when and how formulas in open workbooks are calculated. This function is included for compatibility with Microsoft Excel version 4.0. For controlling calculation in Microsoft Excel version 5.0, see `OPTIONS.CALCULATION`.

Syntax

CALCULATION(type_num, iter, max_num, max_change, update, precision, date_1904, calc_save, save_values, alt_exp, alt_form)

CALCULATION?(type_num, iter, max_num, max_change, update, precision, date_1904, calc_save, save_values, alt_exp, alt_form)

Arguments correspond to check boxes and options in the Calculation dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Type_num is a number from 1 to 3 indicating the type of calculation.

Type_num	Type of calculation
1	Automatic
2	Automatic except tables
3	Manual

Iter corresponds to the Iteration check box. The default is FALSE.

Max_num is the maximum number of iterations. The default is 100.

Max_change is the maximum change of each iteration. The default is 0.001.

Update corresponds to the Update Remote References check box. The default is TRUE.

Precision corresponds to the Precision As Displayed check box. The default is FALSE.

Date_1904 corresponds to the 1904 Date System check box. The default is FALSE in Microsoft Excel for Windows and TRUE in Microsoft Excel for the Macintosh.

Calc_save corresponds to the Recalculate Before Save check box. If calc_save is FALSE, the workbook is not recalculated before saving when in manual calculation mode. The default is TRUE.

Save_values corresponds to the Save External Link Values check box. The default is TRUE.

Alt_exp corresponds to the Transition Formula Evaluation check box in the Transition tab of the Options dialog box.

- If alt_exp is TRUE, Microsoft Excel uses a set of rules compatible with that of Lotus 1-2-3 when calculating formulas. Text is treated as 0; TRUE and FALSE are treated as 1 and 0; and certain characters in database criteria ranges are interpreted the same way Lotus 1-2-3 interprets them. For more information on how Microsoft Excel calculates formulas, see [Controlling Calculation](#).

- If alt_exp is FALSE or omitted, Microsoft Excel calculates normally.

Alt_form corresponds to the Transition Formula Entry check box in the Transition tab of the Options dialog box.

- This argument is available only in Microsoft Excel for Windows.

- If alt_form is TRUE, Microsoft Excel accepts formulas entered in Lotus 1-2-3 style. For more information on formulas entered in Lotus 1-2-3 style, see [Entering formulas](#).

- If alt_form is FALSE or omitted, Microsoft Excel only accepts formulas entered in Microsoft Excel style.

Note Microsoft Excel for Windows and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see `NOW`.

Remarks

Use `GET.DOCUMENT` to return the current calculation settings for your document. For more information, see `GET.DOCUMENT`.

Related Functions

[CALCULATE.DOCUMENT](#)

Calculates the active document only

CALCULATE.NOW

Calculates all open workbooks immediately

GET.DOCUMENT

Returns information about a document

NOW

Returns the serial number of the current date and time

OPTIONS.CALCULATION

Controls calculation

OPTIONS.TRANSITION

Controls transition options

List of Command-Equivalent Functions

CANCEL.COPY

Macro Sheets Only

Equivalent to pressing ESC in Microsoft Excel for Windows or ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh to cancel the marquee after you copy or cut a selection.

Syntax

CANCEL.COPY(render_logical)

Render_logical is a logical value that, if TRUE, places the contents of the Excel Clipboard on the Clipboard or, if FALSE or omitted, does not place them on the Clipboard. Render_logical is available only in Microsoft Excel for the Macintosh.

Related Functions

List of [Command-Equivalent Functions](#)

CHANGE.LINK

Macro Sheets Only

Equivalent to choosing the Change Source button in the Links dialog box, which appears when you choose the Links command from the Edit menu. Changes a link from one supporting workbook to another.

Syntax

CHANGE.LINK(old_text, new_text, type_of_link)

CHANGE.LINK?(old_text, new_text, type_of_link)

Old_text is the path of the link from the active dependent workbook you want to change.

New_text is the path of the link you want to change to.

Type_of_link is the number 1 or 2 specifying what type of link you want to change.

Type_of_link	Link document type
1 or omitted	Microsoft Excel link
2	DDE/OLE link

Remarks

The workbook whose links you want to change must be active when this function is calculated.

Related Functions

[GET.LINK.INFO](#)

Returns information about a link

[LINKS](#)

Returns the name of all linked documents

[OPEN.LINKS](#)

Opens specified supporting workbooks

[SET.UPDATE.STATUS](#)

Controls the update status of a link

[UPDATE.LINK](#)

Updates a link to another workbook

List of [Command-Equivalent Functions](#)

CHART.WIZARD

Macro Sheets Only

Equivalent to choosing the ChartWizard button on the standard or chart toolbar. Creates a chart. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

CHART.WIZARD(long, ref, gallery_num, type_num, plot_by, categories, ser_titles, legend, title, x_title, y_title, z_title, number_cats, number_titles)

CHART.WIZARD?(long, ref, gallery_num, type_num, plot_by, categories, ser_titles, legend, title, x_title, y_title, z_title, number_cats, number_titles)

Long is a logical value that determines which type of ChartWizard button CHART.WIZARD is equivalent to.

- If long is TRUE, CHART.WIZARD is equivalent to using the five-step ChartWizard button.
- If long is FALSE or omitted, CHART.WIZARD is equivalent to using the two-step ChartWizard button, and gallery_num, type_num, legend, and the title arguments are ignored.

Ref is a reference to the range of cells on the active worksheet that contains the source data for the chart, or the object identifier of the chart if it has already been created.

Gallery_num is a number from 1 to 15 specifying the type of chart you want to create.

Gallery_num	Chart
1	Area
2	Bar
3	Column
4	Line
5	Pie
6	Radar
7	XY (scatter)
8	Combination
9	3-D area
10	3-D bar
11	3-D column
12	3-D line
13	3-D pie
14	3-D surface
15	Doughnut

Type_num is a number identifying a formatting option. The formatting options are shown in the dialog box of the AutoFormat command. The first formatting option in any gallery is 1. For a complete list of galleries, see [What's the best chart type for your data?](#)

Plot_by is the number 1 or 2 and specifies whether the data for each data series is in rows or columns. 1 specifies rows; 2 specifies columns. If plot_by is omitted, Microsoft Excel uses the appropriate value for the chart you're creating.

Categories is the number 1 or 2 and specifies whether the first row or column contains a list of x-axis labels, or data for the first data series. 1 specifies x-axis labels; 2 specifies the first data series. If categories is omitted, Microsoft Excel uses the appropriate value for the chart you're creating. If number_cats is specified, this argument is ignored.

Ser_titles is the number 1 or 2 and specifies whether the first column or row contains series titles, or data for the first data point in each series. 1 specifies series titles; 2 specifies the first data point. If ser_titles is omitted, Microsoft Excel uses the appropriate value for the chart you're creating. If number_titles is specified, this argument is ignored.

Legend is the number 1 or 2 and specifies whether to include a legend. 1 specifies a legend; 2 specifies no legend. If legend is omitted, Microsoft Excel does not include a legend.

For the following arguments, if an argument is omitted or is empty text (""), no title is specified.

Title is text that you want to use as a chart title.

X_title is text that you want to use as an x-axis title.

Y_title is text that you want to use as a y-axis title.

Z_title is text that you want to use as a z-axis title.

Number_cats specifies the number of rows or columns (depending on the value of plot_by) to use for the category labels in the chart. This argument overrides the categories argument.

Number_titles specifies the number of rows or columns (depending on the value of plot_by) to use for the series labels in the chart. This argument overrides the ser_title argument.

Remarks

If you are using the macro recorder, Microsoft Excel records a CREATE.OBJECT and a COPY function when the chart is created and a CHART.WIZARD function when the chart is formatted.

Related Functions

CREATE.OBJECT Creates an object

List of Command-Equivalent Functions

CLEAR

Macro Sheets Only

Equivalent to choosing the Clear command from the Edit menu. Clears contents, formats, notes, or all of these from the active worksheet or macro sheet. Clears series or formats from the active chart.

Syntax

CLEAR(type_num)

CLEAR?(type_num)

Type_num is a number from 1 to 4 specifying what to clear. Only values 1, 2, and 3 are valid if the selected item is a chart.

On a worksheet or macro sheet, or if an entire chart is selected, the following occurs.

Type_num	Clears
1	All
2	Formats (if a chart, clears the chart format or clears pictures)
3	Contents (if a chart, clears all data series)
4	Notes (including sound notes; this does not apply to charts)

On a chart, if a single point, an entire data series, error bars, or a trend line is selected, the following occurs.

Type_num	Clears
1	Selected series, error bars, or trend line
2	Format in the selected point, series, error bars, or trend line

If type_num is omitted, the default values are set as shown in the following table.

Active sheet	Type_num
Worksheet	3
Macro sheet	3
Chart (with no selection)	1
Chart (with item selected)	Deletes the selected item

Related Function

EDIT.DELETE Removes cells from a sheet

List of Command-Equivalent Functions

CLOSE

Macro Sheets Only

Closes the active window. In Microsoft Excel for Windows, CLOSE is equivalent to choosing the Close command from the Document Control menu. In Microsoft Excel for the Macintosh, CLOSE is equivalent to clicking the close box.

Syntax

CLOSE(save_logical, route_logical)

Save_logical is a logical value that specifies whether to save the file before closing the window.

Save_logical	Result
TRUE	Saves the file
FALSE	Does not save the file
Omitted	If you've made changes to the file, displays a dialog box asking if you want to save the file
Route_logical	is a logical value that specifies whether to route the file after closing it. This argument is ignored if there is not a routing slip present.
Route_logical	Result
TRUE	Routes the file
FALSE	Does not route the file
Omitted	If you've specified recipients for routing, displays a dialog box asking if you want to save the file

Remarks

- Users of Microsoft Excel versions earlier than 4.0 should note that if the macro sheet containing the function is the active sheet, CLOSE now closes the workbook.

Note When you use the CLOSE function, Microsoft Excel does not run any Auto_Close macros before closing the workbook.

Related Functions

<u>CLOSE.ALL</u>	Closes all windows
<u>FILE.CLOSE</u>	Closes the active workbook
<u>SAVE</u>	Saves the active workbook
List of <u>Command-Equivalent Functions</u>	

COLOR.PALETTE

Macro Sheets Only

Copies a color palette from an open workbook to the active workbook. Use COLOR.PALETTE to share color palettes between workbooks.

Syntax

COLOR.PALETTE(file_text)

COLOR.PALETTE?(file_text)

File_text is the name of a workbook, as a text string, that you want to copy a color palette from. The workbook specified by file_text must be open, or COLOR.PALETTE returns the #VALUE! error value and interrupts the macro. If file_text is empty text (""), then COLOR.PALETTE sets colors to the default values.

Related Function

EDIT.COLOR Defines a color on the color palette

List of Command-Equivalent Functions

CONSOLIDATE

Macro Sheets Only

Equivalent to choosing the Consolidate command from the Data menu. Consolidates data from multiple ranges on multiple worksheets into a single range on a single worksheet.

Syntax

CONSOLIDATE(source_refs, function_num, top_row, left_col, create_links)

CONSOLIDATE?(source_refs, function_num, top_row, left_col, create_links)

Source_refs are references to areas that contain data to be consolidated on the destination worksheet. Source_refs must be in text form and include the full path of the file and the cell reference or named ranges in the workbook to be consolidated. Source_refs are usually external references and must be given as an array, for example: {"SHEET1!IncomeOne", "SHEET2!IncomeTwo"}.

To add or delete source_refs from an existing consolidation on a worksheet, reuse the CONSOLIDATE function, specifying the new source_refs.

Function_num is a number from 1 to 11 that specifies one of the 11 functions you can use to consolidate data. If function_num is omitted, the SUM function, number 9, is used. The functions and their corresponding numbers are listed in the following table.

Function_num	Function
1	AVERAGE
2	COUNT
3	COUNTA
4	MAX
5	MIN
6	PRODUCT
7	STDEV
8	STDEVP
9	SUM
10	VAR
11	VARP

The following arguments correspond to text boxes and check boxes in the Consolidate dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Top_row corresponds to the Top Row check box. The default is FALSE.

Left_col corresponds to the Left Column check box. The default is FALSE.

If top_row and left_col are both FALSE or omitted, the data is consolidated by position.

Create_links corresponds to the Create Links To Source Data check box.

Remarks

- If you use the CONSOLIDATE function with no arguments and there is a consolidation on the active worksheet, Microsoft Excel reconsolidates, using the sources, function, and position attributes used to create the existing consolidation.
- If you use the CONSOLIDATE function with no arguments and there is no consolidation on the active worksheet, the function returns the #VALUE! error value.

Related Functions

<u>CHANGE.LINK</u>	Changes supporting workbook links
<u>LINKS</u>	Returns the names of all linked workbooks
<u>OPEN.LINKS</u>	Opens specified supporting workbooks
<u>UPDATE.LINK</u>	Updates a link to another workbook

List of [Command-Equivalent Functions](#)

COPY.TOOL

Macro Sheets Only

Equivalent to selecting a button and choosing the Copy Button Image command from the Edit menu.
Copies a button face to the Clipboard.

Syntax

COPY.TOOL(bar_id, position)

Bar_id specifies the number or name of a toolbar from which you want to copy the button face. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Related Functions

ADD.TOOL Adds one or more buttons to a toolbar

GET.TOOL Returns information about a button or buttons on a toolbar

PASTE.TOOL Pastes a button face from the Clipboard to a specified position on a toolbar

List of Command-Equivalent Functions

DELETE.BAR

Macro Sheets Only

Deletes a custom menu bar.

Syntax

DELETE.BAR(bar_num)

Bar_num is the ID number of the custom menu bar you want to delete.

Tip Rather than trying to discover the ID number of the menu bar you want to delete, use a reference to the ADD.BAR function that created the bar. For example, the following macro formula deletes the menu bar created by the ADD.BAR function in the cell named ReportsBar:

`DELETE.BAR(ReportsBar)`

Related Functions

[ADD.BAR](#) Adds a menu bar

[SHOW.BAR](#) Displays a menu bar

List of [Customizing Functions](#)

DELETE.COMMAND

Macro Sheets Only

Deletes a command from a custom or built-in menu. Use DELETE.COMMAND to remove commands you don't want the user to have access to or to remove custom commands that you have added.

Syntax

DELETE.COMMAND(bar_num, menu, command, subcommand)

Bar_num is the menu bar from which you want to delete the command. Bar_num can be the ID number of a built-in or custom menu bar. See ADD.COMMAND for a list of ID numbers for built-in menu bars and shortcut menus.

Menu is the menu from which you want to delete the command. Menu can be the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

Command is the command you want to delete, or the name of a submenu. Command can be the name of the command as text or the number of the command; the first command on a menu is in position 1.

Subcommand is the command you want to delete from a submenu. If you use subcommand, you must use command as the name of the submenu.

Remarks

- If the specified command does not exist, DELETE.COMMAND returns the #VALUE! error value and interrupts the macro.
- After a command is deleted, the command number for all commands below that command is decreased by one.
- When you delete a built-in command, DELETE.COMMAND returns a unique ID number for that command. You can use this ID number with ADD.COMMAND to restore the built-in command to the original menu.

Example

The following macro formula removes the Compile Reports command from the Reports menu on a custom menu bar created by the ADD.BAR function in a cell named Financials.

```
DELETE.COMMAND(Financials, "Reports", "Compile Reports...")
```

Related Functions

[ADD.COMMAND](#)

Adds a command to a menu

[CHECK.COMMAND](#)

Adds or deletes a check mark to or from a command

[ENABLE.COMMAND](#)

Enables or disables a menu or custom command

[RENAME.COMMAND](#)

Changes the name of a command or menu

List of [Customizing Functions](#)

DELETE.FORMAT

Macro Sheets Only

Equivalent to deleting the specified format in the Number tab in the Format Cells dialog box, which appears when you choose the Cells command from the Format menu, or in the Number tab for selected chart objects. Deletes a specified custom number format.

Syntax

DELETE.FORMAT(format_text)

Format_text is the custom format given as a text string, for example, "000-00-0000". If you specify a built-in Microsoft Excel format, DELETE.FORMAT returns the #VALUE! error value.

Remarks

When you delete a custom number format, all numbers formatted with that number format are formatted with the General format.

Related Functions

FORMAT.NUMBER

Applies a number format to the selection

GET.CELL

Returns information about the specified cell

List of Command-Equivalent Functions

DELETE.MENU

Macro Sheets Only

Deletes a menu or submenu. Use DELETE.MENU to delete menus you have added to menu bars when the supporting macro sheet is closed (using an Auto_Close macro), or any time you want to remove a menu.

Syntax

DELETE.MENU(bar_num, menu, submenu)

Bar_num is the menu bar from which you want to delete the menu. Bar_num can be the number of a Microsoft Excel built-in menu bar or the number returned by a previously run ADD.BAR function. For a list of ID numbers for built-in menu bars, see ADD.COMMAND.

Menu is the menu you want to delete. Menu can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen. If the specified menu does not exist, DELETE.MENU returns the #VALUE! error value and interrupts the macro. After a menu is deleted, the menu number for each menu to the right of that menu is decreased by 1.

Submenu is the name of the submenu you want to delete or the number of the menu in the list of commands.

Remarks

You cannot delete a shortcut menu. Instead, use ENABLE.COMMAND to prevent the user from accessing a shortcut menu.

Example

The following macro formula deletes the Reports menu from the custom menu bar created by the ADD.BAR function in a cell named Financials:

```
DELETE.MENU(Financials, "Reports")
```

Related Functions

[ADD.MENU](#)

Adds a menu to a menu bar

[DELETE.BAR](#)

Deletes a menu bar

[DELETE.COMMAND](#)

Deletes a command from a menu

[ENABLE.COMMAND](#)

Enables or disables a menu or custom command

List of [Customizing Functions](#)

DELETE.TOOL

Macro Sheets Only

Equivalent to selecting a button and dragging it to an area other than a toolbar. Deletes a button from a toolbar.

Syntax

DELETE.TOOL(bar_id, position)

Bar_id specifies the name or number of a toolbar from which you want to delete a button. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Related Functions

ADD.TOOL

Adds one or more buttons to a toolbar

ADD.TOOLBAR

Creates a new toolbar with the specified buttons

DELETE.TOOLBAR

Deletes custom toolbars

List of Command-Equivalent Functions

DELETE.TOOLBAR

Macro Sheets Only

Equivalent to choosing the Delete button from the Toolbars dialog box, which appears when you choose the Toolbars command from the View menu. Deletes a custom toolbar.

Syntax

DELETE.TOOLBAR(bar_name)

Bar_name specifies the name of the toolbar that you want to delete. For detailed information about bar_name, see ADD.TOOL.

Remarks

- You cannot delete built-in toolbars.
- If DELETE.TOOLBAR successfully deletes the toolbar, it returns TRUE. If you try to delete a built-in toolbar, DELETE.TOOLBAR returns the #VALUE! error value, interrupts the macro, and takes no other action.

Related Functions

ADD.TOOL Adds or more buttons to a toolbar

ADD.TOOLBAR Creates a new toolbar with the specified buttons

RESET.TOOLBAR Resets a built-in toolbar to its initial default setting

List of Command-Equivalent Functions

DISPLAY

Macro Sheets Only

Controls whether the screen displays formulas, gridlines, row and column headings, and other screen attributes. There are two syntax forms of this function. Use syntax 1 to control screen display. Use syntax 2 to control the display of the Info window.

<u>Syntax 1</u>	Controls screen display
<u>Syntax 2</u>	Controls display of Info window

DISPLAY Syntax 1

Macro Sheets Only

Controls whether the screen displays formulas, gridlines, row and column headings, and other screen attributes. There are two syntax forms of this function. Use syntax 1 to control screen display. This function is provided for compatibility with Microsoft Excel version 4.0. To control screen display in Microsoft Excel version 5.0, see [OPTIONS.VIEW](#).

Arguments for this syntax form correspond to options and check boxes in the Display Options dialog box in Microsoft Excel version 4.0. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, no action is taken.

Syntax

DISPLAY(formulas, gridlines, headings, zeros, color_num, reserved, outline, page_breaks, object_num)

DISPLAY?(formulas, gridlines, headings, zeros, color_num, reserved, outline, page_breaks, object_num)

Formulas corresponds to the Formulas check box. The default is FALSE on worksheets and TRUE on macro sheets.

Gridlines corresponds to the Gridlines check box. The default is TRUE.

Headings corresponds to the Row & Column Headings check box. The default is TRUE.

Zeros corresponds to the Zero Values check box. The default is TRUE.

Color_num is a number from 0 to 56 corresponding to the gridline and heading colors in the Display Options dialog box; 0 corresponds to automatic color and is the default value.

Reserved is reserved for certain international versions of Microsoft Excel.

Outline corresponds to the Outline Symbols check box. The default is TRUE.

Page_breaks corresponds to the Automatic Page Breaks check box. The default is FALSE.

Object_num is a number from 1 to 3 corresponding to the display options in the Object box.

Object_num	Corresponds to
1 or omitted	Show All
2	Show Placeholders
3	Hide

Related Functions

<u>OPTIONS.VIEW</u>	Controls display
<u>WORKSPACE</u>	Changes workspace settings
<u>ZOOM</u>	Enlarges or reduces a sheet in the active window
<u>Syntax 2</u>	Controls display of Info window
List of <u>Command-Equivalent Functions</u>	

DISPLAY Syntax 2

Macro Sheets Only

Equivalent to choosing the commands from the Info menu when the Info Window is active. Controls which commands on the Info window are in effect. There are two syntax forms of this function. Use syntax 2 to control the display of the Info window. The Info window must be active to use this form of DISPLAY.

Arguments in this syntax form correspond to commands on the Info menu with the same names.

For these arguments:

- If the argument is TRUE, Microsoft Excel displays the corresponding Info item.
- If the argument is FALSE, Microsoft Excel does not display the corresponding Info item.
- If the argument is omitted, the status of the item is unchanged.

Syntax

For controlling Info window display

DISPLAY(cell, formula, value, format, protection, names, precedents, dependents, note)

Cell is a logical value that corresponds to the Cell command and controls the display of cell information in the Info Window. If TRUE, cell information will be displayed; if FALSE, cell information will not be displayed.

Formula is a logical value that corresponds to the Formula command and controls the display of formula information in the Info Window. If TRUE, formula information will be displayed; if FALSE, formula information will not be displayed.

Value is a logical value that corresponds to the Value command and controls the display of value information in the Info Window. If TRUE, value information will be displayed; if FALSE, value information will not be displayed.

Format is a logical value that corresponds to the Format command and controls the display of format information in the Info Window. If TRUE, format information will be displayed; if FALSE, format information will not be displayed.

Protection is a logical value that corresponds to the Protection command and controls the display of protection information in the Info Window. If TRUE, protection information will be displayed; if FALSE, protection information will not be displayed.

Names is a logical value that corresponds to the Names command and controls the display of name information in the Info Window. If TRUE, name information will be displayed; if FALSE, name information will not be displayed.

Precedents is a number from 1 to 3 that specifies which precedents to list, according to the following table.

Dependents is a number from 1 to 3 that specifies which dependents to list, according to the following table.

Precedents or dependents	List
--------------------------	------

0	None
1	Direct only
2	All levels

Note is a logical value that corresponds to the Note command and controls the display of note information in the Info Window. If TRUE, note information will be displayed; if FALSE, note information will not be displayed.

Related Functions

<u>SHOW.INFO</u>	Controls the display of the Info window
<u>ZOOM</u>	Enlarges or reduces a sheet in the active window
<u>Syntax 1</u>	Controls screen display
List of <u>Command-Equivalent Functions</u>	

EDIT.DELETE

Macro Sheets Only

Equivalent to choosing the Delete command from the Edit menu. Removes the selected cells from the worksheet and shifts other cells to close up the space.

Syntax

EDIT.DELETE(shift_num)

EDIT.DELETE?(shift_num)

Shift_num is a number from 1 to 4 specifying whether to shift cells left or up after deleting the current selection or else to delete the entire row or column.

Shift_num	Result
1	Shifts cells left
2	Shifts cells up
3	Deletes entire row
4	Deletes entire column

- If shift_num is omitted and if one cell or a horizontal range is selected, EDIT.DELETE shifts cells up.
- If shift_num is omitted and a vertical range is selected, EDIT.DELETE shifts cells left.

Related Function

CLEAR Clears specified information from the selected cells or chart

List of Command-Equivalent Functions

ELSE

Macro Sheets Only

Used with IF, ELSE.IF, and END.IF to control which functions are carried out in a macro. ELSE signals the beginning of a group of formulas in a macro sheet that will be carried out if the results of all preceding ELSE.IF statements and the preceding IF statement are FALSE. Use ELSE with IF, ELSE.IF, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

Syntax

ELSE()

Remarks

ELSE must be entered in a cell by itself. In other words, the cell can contain only "=ELSE()".

For more information about ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function.

Related Functions

<u>ELSE.IF</u>	Specifies an action to take if an IF or another ELSE.IF function returns FALSE
<u>END.IF</u>	Ends a group of macro functions started with an IF statement
<u>IF</u>	Specifies an action to take if a logical test is TRUE
List of <u>Control Functions</u>	

ELSE.IF

Macro Sheets Only

Used with IF, ELSE, and END.IF to control which functions are carried out in a macro. ELSE.IF signals the beginning of a group of formulas in a macro sheet that will be carried out if the preceding IF or ELSE.IF function returns FALSE and if logical_test is TRUE. Use ELSE.IF with IF, ELSE, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

Syntax

ELSE.IF(logical_test)

Logical_test is a logical value that ELSE.IF uses to determine what functions to carry out next that is, where to branch.

- If logical_test is TRUE, Microsoft Excel carries out the functions between the ELSE.IF function and the next ELSE.IF, ELSE, or END.IF function.
- If logical_test is FALSE, Microsoft Excel immediately branches to the next ELSE.IF, ELSE, or END.IF function.

Remarks

- ELSE.IF must be entered in a cell by itself.
- Logical_test will always be evaluated, even if the ELSE.IF section is not reached (due to a previous IF or ELSE.IF logical_test evaluating to TRUE). For this reason, you should not use formulas that carry out actions for logical_test. If you need to base the ELSE.IF condition on the return value of a formula that carries out an action, use the form "ELSE, IF(logical_test), and END.IF" in place of "ELSE.IF(logical_test)."

For more information about ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function.

Related Functions

<u>CHOOSE</u>	Chooses a value from a list of values
<u>ELSE</u>	Specifies an action to take if an IF function returns FALSE
<u>END.IF</u>	Ends a group of macro functions started with an IF statement
<u>IF</u>	Specifies an action to take if a logical test is TRUE
List of <u>Control Functions</u>	

ENABLE.COMMAND

Macro Sheets Only

Enables or disables a custom command or menu. Disabled commands appear dimmed and can't be chosen. Use ENABLE.COMMAND to control which commands the user can choose in a menu bar.

Syntax

ENABLE.COMMAND(bar_num, menu, command, enable, subcommand)

Bar_num is the menu bar in which a command resides. Bar_num can be the number of a built-in menu bar or the number returned by a previously run ADD.BAR function. See ADD.COMMAND for a list of the built-in menu bar numbers.

Menu is the menu on which the command resides. Menu can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

Command is the command you want to enable or disable. Command can be either the name of the command as text or the number of the command. The top command on a menu is command 1. If command is 0, ENABLE.COMMAND enables or disables the entire menu.

Enable is a logical value specifying whether the command should be enabled or disabled. If enable is TRUE, Microsoft Excel enables the command; if FALSE, it disables the command.

Subcommand is the name of the command on a submenu that you want to enable. If you use subcommand, you must use command as the name of the submenu. Use subcommand 0 to enable an entire submenu.

Remarks

- You cannot disable built-in commands. If the specified command is a built-in command or does not exist, ENABLE.COMMAND returns the #VALUE! error value and interrupts the macro.
- You can hide any shortcut menu from users by using ENABLE.COMMAND with command set to 0.

Example

The following macro formula disables a custom command that had been added previously to the View menu on the worksheet and macro sheet menu bar:

```
ENABLE.COMMAND(10, "View", "Audit...", FALSE)
```

Related Functions

[ADD.COMMAND](#)

Adds a command to a menu

[CHECK.COMMAND](#)

Adds or deletes a check mark to or from a command

[DELETE.COMMAND](#)

Deletes a command from a menu

[RENAME.COMMAND](#)

Changes the name of a command or menu

List of [Customizing Functions](#)

ENABLE.TOOL

Macro Sheets Only

Enables or disables a button on a toolbar. An enabled button can be accessed by the user. Disabled buttons may still be visible but cannot be accessed. Use ENABLE.TOOL to control which buttons the user can choose in a particular situation.

Syntax

ENABLE.TOOL(bar_id, position, enable)

Bar_id is the number or name of a toolbar on which the button resides. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button on the toolbar. Position starts with 1 at the left side (if horizontal) or from the top (if vertical).

Enable specifies whether the button can be accessed. If enable is TRUE or omitted, the user can access the button; if FALSE, the user cannot access it.

Remarks

Microsoft Excel sounds a tone if you click a disabled button.

Example

The following macro formula enables the fourth button in Toolbar1:

```
ENABLE.TOOL("Toolbar1", 4, TRUE)
```

Related Function

GET.TOOL Returns information about a button or buttons on a toolbar

List of Customizing Functions

END.IF

Macro Sheets Only

Ends a block of functions associated with the preceding IF function. You must include one and only one END.IF function for each macro-sheets-only syntax form (syntax 2) of the IF function in a macro. Syntax 1 of the IF function, which can be used on both worksheets and macro sheets, does not require an END.IF function. Use END.IF with IF, ELSE, and ELSE.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.

Syntax

END.IF()

Remarks

- If you accidentally omit an END.IF function, your macro will end with an error at the cell containing the first IF function that does not have a corresponding END.IF function.
- END.IF must be entered in a cell by itself.
- For more information about ELSE, ELSE.IF, END.IF, and IF, and for examples of these functions, see form 2 of the IF function.

Related Functions

<u>ELSE</u>	Specifies an action to take if an IF function returns FALSE
<u>ELSE.IF</u>	Specifies an action to take if an IF or another ELSE.IF function returns FALSE
<u>IF</u>	Specifies an action to take if a logical test is TRUE
List of <u>Control Functions</u>	

EXTEND.POLYGON

Macro Sheets Only

Adds vertices to a polygon. This function must immediately follow a CREATE.OBJECT function or another EXTEND.POLYGON function. Use multiple EXTEND.POLYGON functions to create arbitrarily complex polygons. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

EXTEND.POLYGON(array)

Array is an array of values, or a reference to a range of cells containing values, that indicate the position of vertices in the polygon. The position is measured in points and is relative to the upper-left corner of the polygon's bounding rectangle.

- A vertex is a point. Each vertex is defined by a pair of coordinates in one row of array.
- The polygon is defined by the array argument to the CREATE.OBJECT function and to all the immediately following EXTEND.POLYGON functions.
- If the polygon contains many vertices, one array may not be sufficient to define it. If the number of elements in the formula exceeds 1024, you must include additional EXTEND.POLYGON functions. If you're recording a macro, Microsoft Excel automatically records additional EXTEND.POLYGON functions as needed.

Related Functions

CREATE.OBJECT Creates an object

FORMAT.SHAPE Inserts, moves, or deletes vertices of the selected polygon

List of Command-Equivalent Functions

FILL.AUTO

Macro Sheets Only

Equivalent to copying cells or automatically filling a selection by dragging the fill selection handle with the mouse (the AutoFill feature).

Syntax

FILL.AUTO(destination_ref, copy_only)

Destination_ref is the range of cells into which you want to fill data. The top, bottom, left, or right end of destination_ref must include all of the cells in the source reference (the current selection).

Copy_only is a number specifying whether to copy cells or perform an AutoFill operation.

Value	Result
0 or omitted	Normal AutoFill
1 or TRUE	Copy cells
2	Copy formats
3	Fill values
4	Increment
5	Increment by day
6	Increment by weekday
7	Increment by month
8	Increment by year
9	Linear trend
10	Growth trend

Related Functions

COPY Copies and pastes data or objects

DATA.SERIES Fills a range of cells with a series of numbers or dates

List of Command-Equivalent Functions

FILE.CLOSE

Macro Sheets Only

Equivalent to choosing the Close command from the File menu. Closes the active workbook.

Syntax

FILE.CLOSE(save_logical, route_logical)

Save_logical is a logical value specifying whether to save the file before closing it.

Save_logical	Result
TRUE	Saves the workbook
FALSE	Does not save the workbook
Omitted	If you've made changes to the workbook, displays a dialog box asking if you want to save the workbook
Route_logical is a logical value that specifies whether to route the file after closing it. This argument is ignored if there is not a routing slip present.	
Route_logical	Result
TRUE	Routes the file
FALSE	Does not route the file
Omitted	If you've specified recipients for routing, displays a dialog box asking if you want to save the file

Remarks

If you make any changes to the structure of a workbook, such as the name of sheets, their order, and so on, then a message will be displayed reminding you that there are unsaved changes, regardless of the save_logical value.

Note When you use the FILE.CLOSE function, Microsoft Excel does not run any Auto_Close macros before closing the workbook.

Related Functions

<u>CLOSE</u>	Closes the active window
<u>CLOSE.ALL</u>	Closes all unprotected windows
<u>FCLOSE</u>	Closes a text file
List of <u>Command-Equivalent Functions</u>	

FORMAT.AUTO

Macro Sheets Only

Equivalent to choosing the AutoFormat command from the Format menu when a worksheet is active or clicking the AutoFormat button. Formats the selected range of cells from a built-in gallery of formats.

Syntax

FORMAT.AUTO(format_num, number, font, alignment, border, pattern, width)

FORMAT.AUTO?(format_num, number, font, alignment, border, pattern, width)

Format_num is a number from 1 to 17 corresponding to the formats in the Table Format list box in the AutoFormat dialog box.

Format_num	Table Format
0	None
1 or omitted	Classic 1
2	Classic 2
3	Classic 3
4	Accounting 1
5	Accounting 2
6	Accounting 3
7	Colorful 1
8	Colorful 2
9	Colorful 3
10	List 1
11	List 2
12	List 3
13	3D Effects 1
14	3D Effects 2
15	Japan 1 (Far East versions of Microsoft Excel only)
16	Japan 2 (Far East versions of Microsoft Excel only)
17	Accounting 4
18	Simple

The following arguments are logical values corresponding to the Formats To Apply check boxes in the AutoFormat dialog box. If an argument is TRUE or omitted, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Number corresponds to the Number check box.

Font corresponds to the Font check box.

Alignment corresponds to the Alignment check box.

Border corresponds to the Border check box.

Pattern corresponds to the Pattern check box.

Width corresponds to the Column Width/Row Height check box.

Related Functions

[ALIGNMENT](#)

Aligns or wraps text in cells

[BORDER](#)

Adds a border to the selected cell or object

[FONT.PROPERTIES](#)

Applies a font to the selection

[FORMAT.NUMBER](#)

Applies a number format to the selection

[PATTERNS](#)

Changes the appearance of the selected object

List of [Command-Equivalent Functions](#)

FORMAT.MOVE

Macro Sheets Only

Equivalent to moving an object with the mouse. Moves the selected object to the specified position and, if successful, returns TRUE. If the selected object cannot be moved, FORMAT.MOVE returns FALSE.

There are three syntax forms of this function. Use syntax 1 to move worksheet objects. Use syntax 2 to move chart items. Use syntax 3 to move pie-chart and doughnut-chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax 1

Moves worksheet items

Syntax 2

Moves chart items

Syntax 3

Moves pie-chart and doughnut-chart items

FORMAT.MOVE Syntax 1

Macro Sheets Only

Equivalent to moving an object with the mouse. Moves the selected object to the specified position and, if successful, returns TRUE. If the selected object cannot be moved, FORMAT.MOVE returns FALSE.

There are three syntax forms of this function. Use syntax 1 to move worksheet objects. Use syntax 2 to move chart items. Use syntax 3 to move pie-chart and doughnut-chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

FORMAT.MOVE(x_offset, y_offset, reference)

FORMAT.MOVE?(x_offset, y_offset, reference)

X_offset specifies the horizontal position to which you want to move the object and is measured in points from the upper-left corner of the object to the upper-left corner of the cell specified by reference. A point is 1/72nd of an inch.

Y_offset specifies the vertical position to which you want to move the object and is measured in points from the upper-left corner of the object to the upper-left corner of the cell specified by reference.

Reference specifies which cell or range of cells to place the object in relation to.

- If reference is a range of cells, only the upper-left cell is used.
- If reference is omitted, it is assumed to be cell A1.

Remarks

The position of an object is based on its upper-left corner. For ovals and arcs, the position is based on the upper-left corner of the bounding rectangle of the object.

Example

The following macro formula moves an object on the active worksheet so that it is 10 points horizontally offset and 15 points vertically offset from cell D4:

```
FORMAT.MOVE(10, 15, !$D$4)
```

Related Functions

[CREATE.OBJECT](#) Creates an object

[FORMAT.SIZE](#) Sizes an object

[WINDOW.MOVE](#) Moves a window

[Syntax 2](#) Moves chart items

[Syntax 3](#) Moves pie-chart and doughnut-chart items

List of [Command-Equivalent Functions](#)

FORMAT.MOVE Syntax 2

Macro Sheets Only

Equivalent to moving an object with the mouse. Moves the base of the selected object to the specified position and, if successful, returns TRUE. If the selected object cannot be moved, FORMAT.MOVE returns FALSE. There are three syntax forms of this function. Use syntax 3 to move pie-chart and doughnut-chart items. Use syntax 1 to move worksheet objects. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

FORMAT.MOVE(x_pos, y_pos)

FORMAT.MOVE?(x_pos, y_pos)

X_pos specifies the horizontal position to which you want to move the object and is measured in points from the base of the object to the lower-left corner of the window. A point is 1/72nd of an inch.

Y_pos specifies the vertical position to which you want to move the object and is measured in points from the base of the object to the lower-left corner of the window.

Remarks

- The base of a text label on a chart is the lower-left corner of the text rectangle.
- The base of an arrow is the end without the arrowhead.
- The base of a pie slice is the point.

Example

On a chart, the following macro formula moves the base of the selected chart object 10 points to the right of and 20 points above the lower-left corner of the window:

```
FORMAT.MOVE(10, 20)
```

Related Functions

<u>FORMAT.SIZE</u>	Sizes an object
<u>WINDOW.MOVE</u>	Moves a window
<u>Syntax 1</u>	Moves worksheet items
<u>Syntax 3</u>	Moves pie-chart and doughnut-chart items
List of <u>Command-Equivalent Functions</u>	

FORMAT.MOVE Syntax 3

Macro Sheets only

Equivalent to exploding by moving a pie-chart or doughnut-chart slice with the mouse. Sets the percentage of pie-chart or doughnut-chart slice explosion, and, if successful, returns TRUE. If the selected object cannot be exploded, returns FALSE. There are three syntax forms of this function. Use syntax 1 to move worksheet items. Use syntax 2 to move chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

FORMAT.MOVE(explosion_num)

Explosion_num is a number specifying the explosion percentage for the selected pie slice or the entire chart (if the series is selected). Zero is no explosion (the tip of the slice is in the center of the pie).

Related Functions

<u>FORMAT.SIZE</u>	Sizes an object
<u>Syntax 1</u>	Moves worksheet items
<u>Syntax 2</u>	Moves chart items
<u>WINDOW.MOVE</u>	Moves a window
List of <u>Command-Equivalent Functions</u>	

FORMAT.SHAPE

Macro Sheets Only

Equivalent to clicking the reshape button on the Drawing toolbar and then inserting, moving, or deleting vertices of the selected polygon. A vertex is a point defined by a pair of coordinates in one row of the array that defines the polygon. The array is created by CREATE.OBJECT and EXTEND.POLYGON functions.

Syntax

FORMAT.SHAPE(vertex_num, insert, reference, x_offset, y_offset)

Vertex_num is a number corresponding to the vertex you want to insert, move, or delete.

Insert is a logical value specifying whether to insert a vertex, or move or delete a vertex.

- If insert is TRUE, Microsoft Excel inserts a vertex between the vertices vertex_num and vertex_num-1. The number of the new vertex then becomes vertex_num. The number of the vertex previously identified by vertex_num becomes vertex_num+1, and so on.
- If insert is FALSE, Microsoft Excel deletes the vertex (if the remaining arguments are omitted) or moves the vertex to the position specified by the remaining arguments.

Reference is the reference from which the vertex you are inserting or moving is measured; that is, the cell or range of cells to use as the basis for the x and y offsets.

- If reference is a range of cells, only the upper-left cell is used.
- If reference is omitted, the vertex is measured from the upper-left corner of the polygon's bounding rectangle.

X_offset is the horizontal distance from the upper-left corner of reference to the vertex. X_offset is measured in points. A point is 1/72nd of an inch. If reference is omitted, x_offset specifies the horizontal distance from the upper-left corner of the polygon bounding rectangle.

Y_offset is the vertical distance from the upper-left corner of reference to the vertex. Y_offset is measured in points. If reference is omitted, y_offset specifies the vertical distance from the upper-left corner of the polygon bounding rectangle.

Remarks

You cannot delete a vertex if only two vertices remain.

Examples

The following macro formula deletes the second vertex of the selected polygon:

```
FORMAT.SHAPE(2, FALSE)
```

The following macro formula moves the thirteenth vertex 6 points to the right and 4 points below the upper-left corner of cell B5 on the active worksheet:

```
FORMAT.SHAPE(13, FALSE, !$B$5, 6, 4)
```

The following macro formula inserts a new vertex between vertices 2 and 3. The new vertex is 60 points to the right and 75 points below the upper-left corner of the polygon's bounding rectangle:

```
FORMAT.SHAPE(3, TRUE, , 60, 75)
```

Related Functions

[CREATE.OBJECT](#)

Creates an object

[EXTEND.POLYGON](#)

Adds vertices to a polygon

List of [Command-Equivalent Functions](#)

FORMAT.SIZE

Macro Sheets Only

Equivalent to sizing an object with the mouse. Sizes the selected object and returns TRUE. If the selected chart object cannot be sized, FORMAT.SIZE returns FALSE. There are two syntax forms of this function. Use syntax 1 to size worksheet objects and chart items absolutely. Use syntax 2 relative to a cell or range of cells to size only worksheet objects. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax 1

Sizes worksheet objects and chart items

Syntax 2

Sizes worksheet objects relative to a cell or range

FORMAT.SIZE Syntax 1

Macro Sheets Only

Equivalent to sizing an object with the mouse. Sizes the selected object and returns TRUE. If the selected chart object cannot be sized, FORMAT.SIZE returns FALSE. There are two syntax forms of this function. Use syntax 1 to size worksheet objects and chart items absolutely. Use syntax 2 relative to a cell or range of cells to size only worksheet objects. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

FORMAT.SIZE(width, height)

FORMAT.SIZE?(width, height)

Width specifies the width of the selected object, measured in points. A point is 1/72nd of an inch.

Height specifies the height of the selected object, measured in points.

You do not always have to use both arguments. For example, if you specify height and not width, the height changes but the width does not.

Remarks

- The base of a text label on a chart is the lower-left corner of the text rectangle.
- The base of an arrow is the end without the arrowhead.

Related Functions

FORMAT.MOVE

Moves the selected object

SIZE

Changes the size of a window

Syntax 2

Sizes worksheet objects relative to a cell or range

List of Command-Equivalent Functions

FORMAT.SIZE Syntax 2

Macro Sheets Only

Equivalent to sizing an object with the mouse. Sizes the selected worksheet object and returns TRUE. If the selected object cannot be sized, FORMAT.SIZE returns FALSE. There are two syntax forms of this function. Use syntax 2 to size worksheet objects relative to a cell or range of cells. Use syntax 1 to size worksheet objects and chart items. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax

FORMAT.SIZE(x_off, y_off, reference)

FORMAT.SIZE?(x_off, y_off, reference)

X_off specifies the width of the selected object and is measured in points from the lower-right corner of the object to the upper-left corner of reference. A point is 1/72nd of an inch. If omitted, x_off is assumed to be 0. If reference is omitted, x_off specifies the horizontal size.

Y_off specifies the height of the selected object and is measured in points from the lower-right corner of the object to the upper-left corner of reference. If omitted, y_off is assumed to be 0. If reference is omitted, y_off specifies the vertical size.

Reference specifies the cell or range of cells to use as the basis for the offset and for sizing. If reference is a range of cells, only the upper-left cell in the range is used.

Related Functions

FORMAT.MOVE

Moves the selected object

SIZE

Changes the size of a window

Syntax 1

Sizes worksheet objects and chart items

List of Command-Equivalent Functions

FORMULA

Macro Sheets Only

Enters a formula in the active cell or in a reference. There are two syntax forms of this function. Use syntax 1 to enter numbers, text, references, and formulas in a worksheet. Although syntax 1 can also be used to enter values on a macro sheet, you will not generally use FORMULA for this purpose. Use syntax 2 to enter a formula in a chart. For information about setting values on a macro sheet, see "Remarks" later in this topic.

Syntax 1

Enters numbers, text, references, and formulas in a worksheet

Syntax 2

Enters formulas in a chart

FORMULA Syntax 1

Macro Sheets Only

Enters a formula in the active cell or in a reference. If the active sheet is a worksheet, using FORMULA is equivalent to entering formula_text in the cell specified by reference. Formula_text is entered just as if you typed it in the formula bar.

There are two syntax forms of this function. Use syntax 1 to enter numbers, text, references, and formulas in a worksheet. Although syntax 1 can also be used to enter values on a macro sheet, you will not generally use FORMULA for this purpose. Use syntax 2 to enter a formula in a chart. For information about setting values on a macro sheet, see "Remarks" later in this topic.

Syntax

FORMULA(formula_text, reference)

Formula_text can be text, a number, a reference, or a formula in the form of text, or a reference to a cell containing any of the above.

- If formula_text contains references, they must be R1C1-style references, such as "=RC[1]*(1+R1C1)". If you are recording a macro when you enter a formula, Microsoft Excel converts A1-style references to R1C1-style references. For example, if you enter the formula =B2*(1+\$A\$1) in cell C2 while recording, Microsoft Excel records that action as =FORMULA("=RC[-1]*(1+R1C1)").
 - If formula_text is a formula, the formula is entered. Text arguments must be surrounded by double sets of quotation marks. For example, to enter the formula =IF(\$A\$1="Hello World", 1, 0) in the active cell with the FORMULA function, you would use the formula FORMULA("=IF(R1C1=""Hello World"", 1, 0)")
- If formula_text is a number, text, or logical value, the value is entered as a constant.

Reference specifies where formula_text is to be entered. It can be a reference to a cell in the active workbook or an external reference to a workbook. If reference is omitted, formula_text is entered in the active cell.

Remarks

Consider the following guidelines as you choose a function to set values on a worksheet or macro sheet:

- Use FORMULA to enter formulas and change values in a worksheet cell.
- SET.VALUE changes values on the macro sheet. Use SET.VALUE to assign initial values to a reference and to store values during the calculation of the macro.
- SET.NAME creates names on the macro sheet. Use SET.NAME to create a name and immediately assign a value to the name.

Examples

If the active sheet is a worksheet, the following macro formula enters the number constant 523 in the active cell:

```
FORMULA(523)
```

If the active sheet is a worksheet, the following macro formula enters the result of the INPUT function in cell A5:

```
FORMULA(INPUT("Enter a formula:", 0), !$A$5)
```

If you're using R1C1-style references and the active sheet is a worksheet, the following macro formula enters the formula =RC[-1]*(1+R1C1) in the active cell:

```
FORMULA("=RC[-1]*(1+R1C1)")
```

If the active sheet is a worksheet, the following macro formulas enter the number 1000 in the cell two rows down and three columns right from the active cell. The R1C1-style formula is shorter, but the OFFSET method may provide faster performance in larger macro sheets.

```
FORMULA(1000, OFFSET(ACTIVE.CELL(), 2, 3))  
FORMULA(1000, "R[2]C[3]")
```

The following macro formula enters the phrase "Year to Date" in cell B4 on the sheet named SALES 1993:

FORMULA("Year to Date", 'SALES 1993'!B4)

Related Functions

FORMULA.ARRAY

Enters an array

FORMULA.FILL

Enters a formula in the specified range

SET.VALUE

Sets the value of a cell on a macro sheet

Syntax 2

Enters formulas in a chart

List of Command-Equivalent Functions

FORMULA Syntax 2

Macro Sheets Only

Enters a text label or SERIES formula in a chart. To enter formulas on a worksheet or macro sheet, use syntax 1 of this function.

Syntax

FORMULA(formula_text)

Formula_text is the text label or SERIES formula you want to enter into the chart.

If

Formula_text can be treated as a text label and the current selection is a text label

Formula_text can be treated as a text label and there is no current selection or the current selection is not a text label

Formula_text can be treated as a SERIES formula and the current selection is a SERIES formula

Formula_text can be treated as a SERIES formula and the current selection is not a SERIES formula

Then

The selected text label is replaced with formula_text.

Formula_text creates a new unattached text label.

The selected SERIES formula is replaced with formula_text.

Formula_text creates a new SERIES formula.

Remarks

You would normally use the EDIT.SERIES function to create or edit a chart series. For more information, see EDIT.SERIES.

Example

The following macro formula enters a SERIES formula on the chart. If the current selection is a SERIES formula, it is replaced:

```
FORMULA("=SERIES("Title", , {1, 2, 3}, 1)")
```

Related Function

EDIT.SERIES

Creates or changes a chart series

Syntax 1

Enters numbers, text, references, and formulas in a worksheet

List of Command-Equivalent Functions

FORMULA.ARRAY

Macro Sheets Only

Enters a formula as an array formula in the range specified or in the current selection. Equivalent to entering an array formula while pressing CTRL+SHIFT+ENTER in Microsoft Excel for Windows or COMMAND+ENTER in Microsoft Excel for the Macintosh.

Syntax

FORMULA.ARRAY(formula_text, reference)

Formula_text is the text you want to enter in the array. For more information on formula_text, see the first form of FORMULA.

Reference specifies where formula_text is entered. It can be a reference to a cell on the active worksheet or an external reference to a named workbook. Reference must be a R1C1-style reference in text form. If reference is omitted, formula_text is entered in the active cell.

Examples

If the selection is D25:E25, the following macro formula enters the array formula {=D22:E22+D23:E23} in the range D25:E25:

```
FORMULA.ARRAY ("=R[-3]C:R[-3]C[1]+R[-2]C:R[-2]C[1]")
```

Regardless of the selection, the following macro formula enters the array formula {=D22:E22+D23:E23} in the range D25:E25:

```
FORMULA.ARRAY ("=R[-3]C:R[-3]C[1]+R[-2]C:R[-2]C[1]", "R25C4:R25C5")
```

To use FORMULA.ARRAY to put an array in a specific workbook, specify the name of the workbook as an external reference in the reference argument. Using "[SALES.XLS]North!R25C3:R25C4" as the reference argument in the preceding example would enter the array in cells C25:D25 on the worksheet named North in the workbook SALES.XLS. Using "SALES!R25C3:R25C4" as the reference argument would enter the array in the same cells in the worksheet named SALES.

Related Functions

FORMULA Enters values into a cell or range or onto a chart

FORMULA.FILL Enters a formula in the specified range

List of Command-Equivalent Functions

FORMULA.FILL

Macro Sheets Only

Enters a formula in the range specified or in the current selection. Equivalent to entering a formula in a range of cells while pressing CTRL+ENTER in Microsoft Excel for Windows or OPTION+ENTER in Microsoft Excel for the Macintosh.

Syntax

FORMULA.FILL(formula_text, reference)

Formula_text is the text with which you want to fill the range. For more information on formula_text, see FORMULA.

Reference specifies where formula_text is entered. It can be a reference to a range in the active worksheet or an external reference to a named workbook. If omitted, formula_text is entered in the current selection.

Related Functions

DATA.SERIES

Fills a range of cells with a series of numbers or dates

FORMULA

Enters values into a cell or range or onto a chart

FORMULA.ARRAY

Enters an array

List of Command-Equivalent Functions

FORMULA.FIND

Macro Sheets Only

Equivalent to choosing the Find command from the Edit menu. Selects the next or previous cell containing the specified text and returns TRUE. If a matching cell is not found, FORMULA.FIND returns FALSE and displays a message.

Syntax

FORMULA.FIND(text, in_num, at_num, by_num, dir_num, match_case)

FORMULA.FIND?(text, in_num, at_num, by_num, dir_num, match_case)

Text is the text you want to find. Text corresponds to the Find What box in the Find dialog box.

In_num is a number from 1 to 3 specifying where to search.

In_num	Searches
1	Formulas
2	Values
3	Notes

At_num is the number 1 or 2 and specifies whether to find cells containing only text or also cells containing text within a longer string of characters.

At_num	Searches for text as
1	A whole string (the only value in the cell)
2	Either a whole string or part of a longer string

By_num is the number 1 or 2 and specifies whether to search by rows or by columns.

By_num	Searches by
1	Rows
2	Columns

Dir_num is the number 1 or 2 and specifies whether to search for the next or previous occurrence of text.

Dir_num	Searches for
1 or omitted	The next occurrence of text
2	The previous occurrence of text

Match_case is a logical value corresponding to the Match Case check box in the Find dialog box. If match_case is TRUE, Microsoft Excel matches characters exactly, including uppercase and lowercase; if FALSE or omitted, matching is not case-sensitive.

Remarks

- In Microsoft Excel for Windows, the dialog-box form of FORMULA.FIND is equivalent to pressing SHIFT+F5.
- If more than one cell is selected when you use FORMULA.FIND, Microsoft Excel searches only that selection.

Related Functions

List of [Command-Equivalent Functions](#)

FORMULA.FIND.NEXT

FORMULA.FIND.PREV

Macro Sheets Only

Finds the next and previous cells on the worksheet, as specified in the Find dialog box, and returns TRUE. (To see the Find dialog box, choose Find from the Edit menu.) If a matching cell is not found, the functions return FALSE. For more information see FORMULA.FIND.

Syntax

FORMULA.FIND.NEXT()

FORMULA.FIND.PREV()

Related Function

DATA.FIND Selects records in a database that match the specified criteria

FORMULA.FIND Finds text in a workbook

List of Command-Equivalent Functions

FORMULA.GOTO

Macro Sheets Only

Equivalent to choosing the Go To command from the Edit menu or to pressing F5. Scrolls through the worksheet and selects a named area or reference. Use FORMULA.GOTO to select a range on any open workbook; use SELECT to select a range on the active workbook.

Syntax

FORMULA.GOTO(reference, corner)

FORMULA.GOTO?(reference, corner)

Reference specifies where to scroll and what to select.

- Reference should be either an external reference to a workbook, an R1C1-style reference in the form of text (see the second example following), or a name.
- If the Go To command has already been carried out, reference is optional. If reference is omitted, it is assumed to be the reference of the cells you selected before the previous Go To command or FORMULA.GOTO macro function was carried out. This feature distinguishes FORMULA.GOTO from SELECT.

Corner is a logical value that specifies whether to scroll through the window so that the upper-left cell in reference is in the upper-left corner of the active window. If corner is TRUE, Microsoft Excel places reference in the upper-left corner of the window; if FALSE or omitted, Microsoft Excel scrolls through normally.

Tip Microsoft Excel keeps a list of the cells you've selected with previous FORMULA.GOTO functions or Go To commands. When you use FORMULA.GOTO with GET.WORKSPACE(41), which returns a horizontal array of previous Go To selections, you can backtrack through multiple previous selections. See the last example below.

Remarks

- If you are recording a macro when you choose the Go To command, the reference you enter in the Reference box of the Go To dialog box is recorded as text in the R1C1 reference style.
- If you are recording a macro when you double-click a cell that has precedents on another worksheet, Microsoft Excel records a FORMULA.GOTO function.

Examples

Each of the following macro formulas goes to cell A1 on the active worksheet:

```
FORMULA.GOTO(!$A$1)
```

```
FORMULA.GOTO("R1C1")
```

Each of the following macro formulas goes to the cells named Sales on the active worksheet and scrolls through the worksheet so that the upper-left corner of Sales is in the upper-left corner of the window:

```
FORMULA.GOTO(!Sales, TRUE)
```

```
FORMULA.GOTO("Sales", TRUE)
```

The following macro formula goes to the cells that were selected by the third most recent FORMULA.GOTO function or Go To command:

```
FORMULA.GOTO(INDEX(GET.WORKSPACE(41), 1, 3))
```

Related Functions

<u>GOTO</u>	Directs macro execution to another cell
<u>HSCROLL</u>	Horizontally scrolls through a sheet by percentage or by column or row number
<u>SELECT</u>	Selects a cell, worksheet object, or chart item
<u>VSCROLL</u>	Vertically scrolls through a sheet by percentage or by column or row number
List of <u>Command-Equivalent Functions</u>	

FORMULA.REPLACE

Macro Sheets Only

Equivalent to choosing the Replace command from the Edit menu. Finds and replaces characters in cells on your worksheet.

Syntax

FORMULA.REPLACE(find_text, replace_text, look_at, look_by, active_cell, match_case)

FORMULA.REPLACE?(find_text, replace_text, look_at, look_by, active_cell, match_case)

Find_text is the text you want to find. You can use the wildcard characters, question mark (?) and asterisk (*), in find_text. A question mark matches any single character; an asterisk matches any sequence of characters. If you want to find an actual question mark or asterisk, type a tilde (~) before the character.

Replace_text is the text you want to replace find_text with.

Look_at is a number specifying whether you want find_text to match the entire contents of a cell or any string of matching characters.

Look_at	Looks for find_text
---------	---------------------

1 or omitted	As the entire contents of a cell
2	As part of the contents of a cell

Look_by is a number specifying whether to search horizontally (through rows) or vertically (through columns).

Look_by	Looks for find_text
---------	---------------------

1 or omitted	By rows
2	By columns

Active_cell is a logical value specifying the cells in which find_text is to be replaced.

- If active_cell is TRUE, find_text is replaced in the active cell only.
- If active_cell is FALSE, find_text is replaced in the entire selection, or, if the selection is a single cell, in the entire sheet.

Match_case is a logical value corresponding to the Match Case check box in the Replace dialog box. If match_case is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If match_case is omitted, the status of the check box is unchanged.

Remarks

- In FORMULA.REPLACE?, the dialog-box form of the function, omitted arguments are assumed to be the same arguments used in the previous replace operation. If there was no previous replace operation, omitted text arguments are assumed to be "" (empty text).
- The result of FORMULA.REPLACE must be a valid cell entry. For example, you cannot replace "=" with "==" at the beginning of a formula.
- If more than a single cell is selected before you use FORMULA.REPLACE, only the selected cells are searched.

Related Functions

FORMULA.FIND Finds text in a workbook

REPLACE Replaces characters within text

List of Command-Equivalent Functions

GET.BAR

Macro Sheets Only

Returns the number of the active menu bar. There are two syntax forms of GET.BAR. Use syntax 1 to return information that you can use with other functions that manipulate menu bars. Use syntax 2 to return information that you can use with functions that add, delete, or alter menu commands.

Syntax 1

Returns the number of the active menu bar

Syntax 2

Returns the name or position number of a specified command on a menu or of a specified menu on a menu bar

GET.BAR Syntax 1

Macro Sheets Only

Returns the number of the active menu bar. There are two syntax forms of GET.BAR. Use syntax 1 to return information that you can use with other functions that manipulate menu bars. For a list of the ID numbers for Microsoft Excel's built-in menu bars, see ADD.COMMAND.

Syntax

GET.BAR()

Example

The following macro formula assigns the name OldBar to the number of the active menu bar. This is useful if you will need to restore the current menu bar after displaying another custom menu bar.

```
SET.NAME("OldBar", GET.BAR())
```

Related Functions

<u>ADD.BAR</u>	Adds a menu bar
<u>SHOW.BAR</u>	Displays a menu bar
<u>Syntax 2</u>	Returns the name or position number of a specified command on a menu or of a specified menu on a menu bar

List of Information Functions

GET.BAR Syntax 2

Macro Sheets Only

Returns the name or position number of a specified command on a menu or of a specified menu on a menu bar. There are two syntax forms of GET.BAR. Use syntax 2 to return information that you can use with functions that add, delete, or alter menu commands.

Syntax

GET.BAR(bar_num, menu, command, subcommand)

Bar_num is the number of a menu bar containing the menu or command about which you want information. Bar_num can be the number of a built-in menu bar or the number returned by a previously run ADD.BAR function. For a list of the ID numbers for Microsoft Excel's built-in menu bars, see ADD.COMMAND.

Menu is the menu on which the command resides or the menu whose name or position you want. Menu can be the name of the menu as text or the number of the menu. Menus are numbered starting with 1 from the left of the menu bar.

Command is the command or submenu whose name or number you want returned. Command can be the name of the command from the menu as text, in which case the number is returned, or the number of the command from the menu, in which case the name is returned. Commands are numbered starting with 1 from the top of the menu. If command is 0, the name or position number of the menu is returned. If an ellipsis (...) follows a command name, such as the Open... command on the File menu, then you must include the ellipsis when referring to that command. See the following examples.

Subcommand returns the name (if number is used for subcommand) or position (if name is used for subcommand) of a command on a submenu. If the command argument refers to an empty submenu, or is a command instead of a submenu, then using subcommand returns #N/A.

Remarks

- If an ampersand is used to indicate the access key in the name of a custom command, the ampersand is included in the name returned by GET.BAR. All built-in commands have an ampersand before the letter used as the access key.
- If the command name or position specified does not exist, GET.BAR returns the #N/A error value.

Examples

In the default worksheet and macro sheet menu bar:

GET.BAR(10, "File", "Print...") equals 14

GET.BAR(10, "File", 14) equals "&Print...^tCTRL+P" (where ^t is a tab character)

GET.BAR(10, 1, "Open") equals #N/A

GET.BAR(10, 1, "Open...") equals 2

Related Functions

[ADD.COMMAND](#)

Adds a command to a menu

[DELETE.COMMAND](#)

Deletes a command from a menu

[GET.TOOLBAR](#)

Retrieves information about a toolbar

[RENAME.COMMAND](#)

Changes the name of a command or menu

[Syntax 1](#)

Returns the number of the active menu bar

List of [Information Functions](#)

GET.TOOL

Macro Sheets Only

Returns information about a button or buttons on a toolbar. Use GET.TOOL to get information about a button to use with functions that add, delete, or alter buttons.

Syntax

GET.TOOL(type_num, bar_id, position)

Type_num specifies what type of information you want GET.TOOL to return.

Type_num	Returns
----------	---------

1	The button's ID number. Gaps are represented by zeros.
2	The reference of the macro assigned to the button. If no macro is assigned, GET.TOOL returns the #N/A error value.
3	If the button is down, returns TRUE. If the button is up, returns FALSE.
4	If the button is enabled, returns TRUE. If the button is disabled, returns FALSE.
5	A logical value indicating the type of the face on the button: TRUE = bitmap FALSE = a default button face
6	The help_text reference associated with the custom button. If the button is built-in, returns #N/A.
7	The balloon_text reference associated with the custom button. If the button is built-in, returns the #N/A error value.
8	The Help context string associated with the custom button.
9	The Tip_text associated with the custom button.

Bar_id specifies the number or name of the toolbar for which you want information. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button on the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical). A position can be occupied by a button or a gap.

Example

The following macro formula requests the help text associated with the third button in Toolbar2:

```
GET.TOOL(6, "Toolbar2", 3)
```

Related Functions

<u>ADD.TOOL</u>	Adds one or more buttons to a toolbar
<u>DELETE.TOOL</u>	Deletes a button from a toolbar
<u>ENABLE.TOOL</u>	Enables or disables a button on a toolbar
<u>GET.TOOLBAR</u>	Retrieves information about a toolbar

List of Information Functions

GOAL.SEEK

Macro Sheets Only

Equivalent to choosing the Goal Seek command from the Tools menu. Calculates the values necessary to achieve a specific goal. If the goal is an amount returned by a formula, the GOAL.SEEK function calculates values that, when supplied to your formula, cause your formula to return the amount you want.

Syntax

GOAL.SEEK(target_cell, target_value, variable_cell)

GOAL.SEEK?(target_cell, target_value, variable_cell)

Target_cell corresponds to the Set Cell box in the Goal Seek dialog box and is a reference to the cell containing the formula. If target_cell does not contain a formula, Microsoft Excel displays an error message.

Target_value corresponds to the To Value box in the Goal Seek dialog box and is the value you want the formula in target_cell to return. This value is called a goal.

Variable_cell corresponds to the By Changing Cell box in the Goal Seek dialog box and is the single cell that you want Microsoft Excel to change so that the formula in target_cell returns target_value.

Target_cell must depend on variable_cell; if it does not, Microsoft Excel will not be able to find a solution.

Remarks

The max_num and max_change values set with the CALCULATION function can be used to change the solution process. Max_num sets the number of iterations; max_change determines the precision of the solution.

Tip You can also use Microsoft Excel Solver to help solve your math equations for optimal values. For more information, see [Overview of Working with Microsoft Excel Solver](#).

Related Functions

Related functions include the SOLVER functions, such as [SOLVER.OPTIONS](#), [SOLVER.SOLVE](#), and so on.

List of [Command-Equivalent Functions](#)

HIDE

Macro Sheets Only

Equivalent to choosing the Hide command from the Window menu. Hides the active window.

Syntax

HIDE()

Tip Hiding windows can speed up your macros. You can switch to hidden windows with the **ACTIVATE** function. You can continue to use functions that refer to specific sheets, such as **FORMULA** and the **GET** functions, even when those sheets are hidden.

Related Function

UNHIDE Displays a hidden window

List of Command-Equivalent Functions

HSCROLL

Macro Sheets Only

Horizontally scrolls through the active sheet by percentage or by column number.

Syntax

HSCROLL(position, col_logical)

Position specifies the column you want to scroll to. Position can be an integer representing the column number or a fraction or percentage representing the horizontal position of the column in the sheet. If position is 0, HSCROLL scrolls through your sheet to its leftmost edge. If position is 1, HSCROLL scrolls through your sheet to its rightmost edge. For charts that do not size with the window, use a fraction or percentage.

Col_logical is a logical value specifying how the function scrolls.

- If col_logical is TRUE, HSCROLL scrolls through the sheet to column position.
- If col_logical is FALSE or omitted, then HSCROLL scrolls through the sheet to the horizontal position represented by the fraction position.

Remarks

- To scroll to a specific column n, either use HSCROLL(n, TRUE) or use HSCROLL(n/256). To scroll to column 38, for example, use HSCROLL(38, TRUE) or HSCROLL(38/256).
- If you are recording a macro and move the scroll box several times in a row, the recorder only records the final location of the scroll box, omitting any intermediate steps. Remember that scrolling does not change the active cell or the selection.

Related Functions

<u>HLINE</u>	Horizontally scrolls through the active window by columns
<u>HPAGE</u>	Horizontally scrolls through the active window one window at a time
<u>VLIN</u>	Vertically scrolls through the active window by rows
<u>VPAGE</u>	Vertically scrolls through the active window one window at a time
<u>VSCROLL</u>	Vertically scrolls through a sheet by percentage or row number

List of Command-Equivalent Functions

IF

Macro Sheets Only

Used with ELSE, ELSE.IF, and END.IF to control which formulas in a macro are executed. There are two syntax forms of the IF function. The following syntax can be used on macro sheets only; use it when you want your macro to branch to a particular set of functions based on the outcome of a logical test. The worksheet form of this function can be used on worksheets and macro sheets.

Syntax

IF(logical_test)

Logical_test is a logical value that IF uses to determine which functions to carry out next that is, where to branch.

- If logical_test is TRUE, Microsoft Excel carries out the functions between the IF function and the next ELSE, ELSE.IF, or END.IF function. Instructions between ELSE.IF or ELSE and END.IF are not carried out.
- If logical_test is FALSE, Microsoft Excel immediately branches to the next ELSE.IF, ELSE, or END.IF function.
- If logical_test produces an error, the macro halts.

Tips

- Use IF with ELSE, ELSE.IF, and END.IF when you want to perform multiple actions based on a condition. This method is preferable to using GOTO because it makes your macros more structured.
- If your macro ends with an error at a cell containing this form of the IF function, make sure there is a corresponding END.IF function.

Example

The following macro runs the macro CompleteEntry if the user chooses OK:

```
IF(ALERT("Are you done with this entry?", 1), CompleteEntry(), )
```

Tip You can indent formulas in a macro. To indent a formula, type as many spaces as you want between the equal sign and the first letter of the formula.

Related Functions

<u>CHOOSE</u>	Chooses a value from a list of values
<u>ELSE</u>	Specifies an action to take if an IF function returns FALSE
<u>ELSE.IF</u>	Specifies an action to take if an IF or another ELSE.IF function returns FALSE
<u>END.IF</u>	Ends a group of macro functions started with an IF statement
<u>ERROR</u>	Specifies what action to take if an error occurs while a macro is running
<u>IF</u>	Specifies a logical test to perform

List of Control Functions

LAST.ERROR

Macro Sheets Only

Returns the reference to the cell where the last macro sheet error occurred. If no error has occurred, LAST.ERROR returns the #N/A error value. Use LAST.ERROR in conjunction with the REFTXT function to quickly locate errors.

Syntax

LAST.ERROR()

Related Function

ERROR Specifies what action to take if an error is encountered while a macro is running

List of Information Functions

LINKS

Macro Sheets Only

Returns, as a horizontal array of text values, the names of all workbooks referred to by external references in the workbook specified. Use LINKS with OPEN.LINKS to open supporting workbooks.

Syntax

LINKS(document_text, type_num)

Document_text is the name of a workbook, including its path. If document_text is omitted, LINKS operates on the active workbook. If the workbook specified by document_text is not open, LINKS returns the #N/A error value.

Type_num is a number from 1 to 6 specifying the type of linked workbooks to return.

Type_num	Returns
1 or omitted	Microsoft Excel link
2	DDE/OLE link (Microsoft Excel for Windows)
3	Reserved
4	Not applicable
5	Publisher (Microsoft Excel for the Macintosh)
6	Subscriber (Microsoft Excel for the Macintosh)

Remarks

- If the active workbook contains no external references, LINKS returns the #N/A error value.
- With the INDEX function, you can select individual workbook names from the array for use in other functions that take workbook names as arguments.
- The names of the workbook are always returned in alphabetic order. If supporting workbooks are open, LINKS returns the names of the workbooks; if supporting workbooks are closed, LINKS includes the full path of each workbook.
- If type_num is 5 or 6, LINKS returns a two-row array in which the first row contains the edition name and the second row contains the reference.

Examples

If a chart named Chart1 is open and contains links to workbook Data1 and Data2, and the LINKS function shown below is entered as an array into a two-cell horizontal range:

LINKS ("Chart1") equals "Data1" in the first cell of the range and "Data2" in the second cell.

In Microsoft Excel for Windows, if the chart named VARIANCE.XLS is open and contains data series that refer to workbook named BUDGET.XLS and ACTUAL.XLS, then:

OPEN.LINKS (LINKS ("VARIANCE.XLS")) opens BUDGET.XLS and ACTUAL.XLS.

In Microsoft Excel for the Macintosh, if the workbook named SALES 1991 is open and contains references to the workbook WEST SALES, SOUTH SALES, and EAST SALES, then:

OPEN.LINKS (LINKS ("SALES 1991")) opens WEST SALES, SOUTH SALES, and EAST SALES.

Related Functions

<u>CHANGE.LINK</u>	Changes supporting workbook links
<u>GET.LINK.INFO</u>	Returns information about a link
<u>OPEN.LINKS</u>	Opens specified supporting workbook
<u>UPDATE.LINK</u>	Updates a link to another workbook

List of Information Functions

MERGE.STYLES

Macro Sheets Only

Equivalent to choosing the Merge button from the Style dialog box, which appears when you choose the Style command from the Format menu. Merges all the styles from another workbook into the active workbook. Use MERGE.STYLES when you want to import styles from another sheet in another workbook.

Syntax

MERGE.STYLES(document_text)

Document_text is the name of a sheet in a workbook from which you want to merge styles into the active workbook.

Remarks

- If any styles from the workbook being merged have the same name as styles in the active workbook, a dialog box appears asking if you want to replace the existing definitions of the styles with the "merged" definitions of the styles. If you choose the Yes button, all the definitions are replaced; if you choose the No button, all the original definitions in the active workbook are retained.
- When you move a sheet with styles to another workbook with styles, any styles with identical names but conflicting definitions have the sheet name added to the style name.

Related Functions

DEFINE.STYLE Creates or changes a cell style

DELETE.STYLE Deletes a cell style

List of Command-Equivalent Functions

MOVE.TOOL

Macro Sheets Only

Moves or copies a button from one toolbar to another.

Syntax

MOVE.TOOL(from_bar_id, from_bar_position, to_bar_id, to_bar_position, copy, width)

From_bar_id specifies the number or name of a toolbar from which you want to move or copy the button. For detailed information, see the description of bar_id in ADD.TOOL.

From_bar_position specifies the current position of the button within the toolbar. From_bar_position starts with 1 at the left side (if horizontal) or at the top (if vertical).

To_bar_id specifies the number or name of a toolbar to which you want to move or paste the button. For detailed information, see the description of bar_id in ADD.TOOL. To_bar_id is optional if you are moving a button within the same toolbar.

To_bar_position specifies where you want to move or paste the button within the toolbar.

To_bar_position starts with 1 at the left side (if horizontal) or at the top (if vertical). To_bar_position is optional if you are only adjusting the width of a drop-down list.

Copy is a logical value specifying whether to copy the button. If copy is TRUE, the button is copied; if FALSE or omitted, the button is moved.

Width is the width, measured in points, of a drop-down list. If the button you are moving is not a drop-down list, width is ignored.

Related Functions

ADD.TOOL Adds one or more buttons to a toolbar

COPY.TOOL Copies a button face to the Clipboard

GET.TOOL Returns information about a button or buttons on a toolbar

List of Command-Equivalent Functions

NAMES

Macro Sheets Only

Returns, as a horizontal array of text, the specified names defined in the specified workbook. The returned array lists the names in alphabetic order. Use NAMES instead of LIST.NAMES when you want to return the names to the macro sheet instead of to the active worksheet.

Syntax

NAMES(document_text, type_num, match_text)

Document_text is text that specifies the workbook whose names you want returned. If document_text is omitted, it is assumed to be the active workbook.

Type_num is a number from 1 to 3 that specifies whether to include hidden names in the returned array.

If type_num is	NAMES returns
1 or omitted	Normal names only
2	Hidden names only
3	All names

Match_text is text that specifies the names you want returned and can include wildcard characters. If match_text is omitted, all names are returned.

Remarks

- Hidden names are defined using the DEFINE.NAME macro function and do not appear in the Paste Name, Define Name, or Go To dialog boxes.
- NAMES returns a horizontal array, so you will normally enter this function as an array in several horizontal cells or define a name to refer to the array that NAMES returns. If you want the names in a vertical array instead, use the TRANSPOSE function.
- You can use the COLUMNS function to count the number of entries in the horizontal array.

Example

The following macro formula returns all names on the active workbook starting with the letter P.

```
NAMES(, 3, "P*")
```

Related Functions

<u>DEFINE.NAME</u>	Defines a name on the active worksheet or macro sheet
<u>DELETE.NAME</u>	Deletes a name
<u>GET.DEF</u>	Returns a name matching a definition
<u>GET.NAME</u>	Returns the definition of a name
<u>LIST.NAMES</u>	Lists names and their associated information
<u>SET.NAME</u>	Defines a name as a value

List of Information Functions

NEW

Macro Sheets Only

Equivalent to choosing the New command from the File menu. Creates a new Microsoft Excel workbook or opens a template.

Syntax

NEW(type_num, xy_series, add_logical)

NEW?(type_num, xy_series, add_logical)

Type_num specifies the type of workbook to create, as shown in the following table. Type_num is most often 5 or quoted text; other values are mainly for compatibility with Microsoft Excel version 4.

Type_num	Workbook
1	New workbook with one worksheet
2	New workbook with one chart based on the current selection
3	New workbook with one macro sheet
4	New workbook with one international macro sheet
5	New workbook with 16 worksheets or based on the default workbook
6	New workbook with one Visual Basic module
7	New workbook with one dialog sheet
Quoted text	Template.

Xy_series is a number from 0 to 3 that specifies how data is arranged in a chart.

Xy_series	Result
0	Displays a dialog box if the selection is ambiguous.
1 or omitted	The first row/column is the first data series.
2	The first row/column contains the category (x) axis labels.
3	The first row/column contains the x-values; the created chart is an xy (scatter) chart.

Add_logical specifies whether or not to add the sheet type to the open workbook. If add_logical is TRUE, the sheet type is inserted before the current sheet; if FALSE or omitted, it is not inserted. This argument is for compatibility with Microsoft Excel version 4.0.

Add_logical is ignored if type_num is 5.

Remarks

You can also use NEW to create new sheets from templates that exist in the startup directory or folder, using for type_num the text that appears in the File New list box. To create new sheets from any template that is not in the start-up directory, use the OPEN function. For more information about templates, see [Overview of creating and using a workbook template](#).

Related Functions

[NEW.WINDOW](#) Creates a new window for an existing worksheet or macro sheet

[OPEN](#) Opens a workbook

List of [Command-Equivalent Functions](#)

PASTE

Macro Sheets Only

Equivalent to choosing the Paste command from the Edit menu. Pastes a selection or object that you copied or cut using the COPY or CUT function. Use PASTE when you want to paste all components of the selection. To paste only specific components of the selection, use PASTE.SPECIAL.

Syntax

PASTE(to_reference)

To_reference is a reference to the cell or range of cells where you want to paste what you have copied. If to_reference is omitted, Microsoft Excel pastes to the current selection. If there is nothing to paste, the macro halts.

Related Functions

<u>COPY</u>	Copies and pastes data or objects
<u>CUT</u>	Cuts or moves data or objects
<u>FORMULA</u>	Enters values into a cell or range or onto a chart
<u>INSERT</u>	Inserts cells
<u>PASTE.LINK</u>	Pastes copied data and establishes a link to its source
<u>PASTE.SPECIAL</u>	Pastes specific components of copied data
List of <u>Command-Equivalent Functions</u>	

PASTE.TOOL

Macro Sheets Only

Equivalent to selecting a button and choosing the Paste Button Image command from the Edit menu.
Pastes a button face from the Clipboard to a specified position on a toolbar.

Syntax

PASTE.TOOL(bar_id, position)

Bar_id specifies the number or name of the toolbar into which you want to paste the button face. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position within the toolbar of the button on which you want to paste the button face. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Related Function

COPY.TOOL Copies a button face

List of Command-Equivalent Functions

PRESS.TOOL

Macro Sheets Only

Formats a button so that it appears either normal or depressed into the screen.

Syntax

PRESS.TOOL(bar_id, position, down)

Bar_id specifies the number or name of the toolbar in which you want to change the button appearance. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Down is a logical value specifying the appearance of the button. If down is TRUE, the button appears depressed into the screen; if FALSE or omitted, it appears normal (up).

Remarks

This function applies only to custom buttons to which macros have already been assigned. An error occurs if you try to process any other type of button.

Example

The following macro formula sets the third button image on Toolbar4 to normal (up).

```
PRESS.TOOL("Toolbar4", 3, FALSE)
```

Related Functions

ADD.TOOL Adds one or more buttons to a toolbar

DELETE.TOOL Deletes a button from a toolbar

List of Customizing Functions

PROTECT.DOCUMENT

Macro Sheets Only

Adds protection to or removes protection from the active sheet, macro sheet, chart, dialog sheet, module, or scenario. Use PROTECT.DOCUMENT to prevent yourself or others from changing cell contents, or objects in a workbook. To protect workbooks in Microsoft Excel version 5.0, see WORKBOOK.PROTECT.

Syntax

PROTECT.DOCUMENT(contents, windows, password, objects, scenarios)

PROTECT.DOCUMENT?(contents, windows, password, objects, scenarios)

Contents is a logical value corresponding to the Contents check box in the Protect Sheet dialog box.

- If contents is TRUE or omitted, Microsoft Excel selects the check box and protects cells on the sheet or macro sheet.
- If contents is FALSE, Microsoft Excel clears the check box (and removes protection if the correct password is supplied).

Windows is provided for compatibility with Microsoft Excel version 4.0. To protect the window placement and structure of workbooks in Microsoft Excel version 5.0, see WORKBOOK.PROTECT.

- If windows is TRUE, Microsoft Excel prevents a document's windows from being moved or sized.
- If windows is FALSE or omitted, Microsoft Excel removes protection if the correct password is supplied.

Password is the password you specify in the form of text to protect or unprotect the file. Password is case-sensitive.

- If password is omitted when you protect a document, then you will be able to remove protection without a password. This is useful if you want only to protect the document from accidental changes.
- If password is omitted when you try to remove protection from a document that was protected with a password, the normal password dialog box is displayed.
- Passwords are not recorded into the PROTECT.DOCUMENT function when you use the macro recorder.

Objects is a logical value. This argument applies only to worksheets and macro sheets. Objects corresponds to the Objects check box in the Protect Sheet dialog box.

- If objects is TRUE or omitted, Microsoft Excel selects the check box and protects all locked objects on the worksheet or macro sheet.
- If objects is FALSE, Microsoft Excel clears the check box.

Scenarios is a logical value that corresponds to the Scenarios check box on the Protect Sheet dialog box. If TRUE, Microsoft Excel protects all the scenarios. If FALSE, the scenarios are not protected.

Remarks

- If contents and objects are FALSE, PROTECT.DOCUMENT carries out the Unprotect Sheet command. If contents, or objects is TRUE, it carries out the Protect Sheet command.
- Make sure that you hide macro sheets that protect or unprotect worksheets. If you type a password directly into the function on an unhidden macro sheet, then someone could see the password needed to unprotect the worksheet. For example, PROTECT.DOCUMENT(TRUE, TRUE, "XD1411C", TRUE) .

Warning If you forget the password of a document that was previously protected with a password, you cannot unprotect the document.

Related Functions

[CELL.PROTECTION](#)

Controls protection for the selected cells

[ENTER.DATA](#)

Turns Data Entry mode on and off

[OBJECT.PROTECTION](#)

Controls how an object is protected

[SAVE.AS](#)

Saves a workbook and allows you to specify the name, file type, password, backup file, and location of the workbook

[WORKBOOK.PROTECT](#)

Protects a workbook

List of Command-Equivalent Functions

RENAME.COMMAND

Macro Sheets Only

Changes the name of a built-in or custom menu command or the name of a menu. Use RENAME.COMMAND to change the name of a command on a menu, for example, when you create two custom commands that toggle on the menu. Examples of two built-in commands that toggle are the Page Break and Remove Page Break commands on the Insert menu.

Syntax

RENAME.COMMAND(bar_num, menu, command, name_text, position)

Bar_num can be either the number of one of the Microsoft Excel built-in menu bars or the number returned by a previously run ADD.BAR function. See ADD.COMMAND for a list of ID numbers for built-in menu bars.

Menu can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

Command can be either the name of the command as text or the number of the command to be renamed (the first command on a menu is command 1). If command is 0, RENAME.COMMAND renames the menu instead of the command. Because other macros can change the position of custom menu commands, you should use the name of the command rather than a number whenever possible.

If the specified menu bar, menu, or command does not exist, RENAME.COMMAND returns the #VALUE! error value and interrupts the macro.

Name_text is the new name for the command.

Position is the name of the command on a submenu that you want to rename. If you use position, you must use command as the name of the submenu.

Tip To specify an access key for the new name, precede the character you want to use with an ampersand (&). The access key is indicated by an underline under one letter of a menu or command name. In Microsoft Excel for the Macintosh, you can use the General tab in the Options dialog box to turn command underlining on or off. To see the Options dialog box, choose Options from the Tools menu.

Example

To rename the Save All command as Global Save, and to make the letter "G" in Global Save an access key, use the following macro formula:

```
RENAME.COMMAND(10, "File", "Close All", "&Global Close")
```

Related Functions

[ADD.COMMAND](#)

Adds a command to a menu

[CHECK.COMMAND](#)

Adds or deletes a check mark to or from a command

[DELETE.COMMAND](#)

Deletes a command from a menu

[ENABLE.COMMAND](#)

Enables or disables a menu or custom command

List of [Customizing Functions](#)

RESET.TOOL

Macro Sheets Only

Equivalent to choosing the Reset Button Image command from the Tool shortcut menu. Resets a button to its original button face.

Syntax

RESET.TOOL(bar_id, position)

Bar_id is the number or name of the toolbar containing the button you want to reset. For detailed information about bar_id, see ADD.TOOL.

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Related Functions

ADD.TOOL Adds one or more buttons to a toolbar

DELETE.TOOL Deletes a button from a toolbar

RESET.TOOLBAR Resets a button to its original button face

List of Customizing Functions

RESET.TOOLBAR

Macro Sheets Only

Resets built-in toolbars to the default Microsoft Excel set.

Syntax

RESET.TOOLBAR(bar_id)

Bar_id specifies the number or name of the toolbar that you want to reset. For detailed information about bar_id, see [ADD.TOOL](#).

Remarks

If RESET.TOOLBAR successfully resets the toolbar, it returns TRUE. If you try to reset a custom toolbar, RESET.TOOLBAR returns FALSE and takes no other action.

Related Function

[ADD.TOOL](#) Adds one or more tools to a toolbar

[DELETE.TOOLBAR](#) Deletes custom toolbars

List of [Customizing Functions](#)

SAVE.AS

Macro Sheets Only

Equivalent to choosing the Save As command from the File menu. Use SAVE.AS to specify a new filename, file type, protection password, or write-reservation password, or to create a backup file.

Syntax

SAVE.AS(document_text, type_num, prot_pwd, backup, write_res_pwd, read_only_rec)

SAVE.AS?(document_text, type_num, prot_pwd, backup, write_res_pwd, read_only_rec)

Document_text specifies the name of a workbook to save, such as SALES.XLS (in Microsoft Excel for Windows) or SALES (in Microsoft Excel for the Macintosh). You can include a full path in document_text, such as C:\EXCEL\ANALYZE.XLS (in Microsoft Excel for Windows) or HARDDISK:FINANCIALS:ANALYZE (in Microsoft Excel for the Macintosh).

Type_num is a number specifying the file format in which to save the workbook. For information about file formats, see [File Formats](#) in online Help. The following table lists the file format corresponding to each type_num.

Type_num	File format
1 or omitted	Normal
2	SYLK
3	Text
4	WKS
5	WK1
6	CSV
7	DBF2
8	DBF3
9	DIF
10	Reserved
11	DBF4
12	Reserved
13	Reserved
14	Reserved
15	WK3
16	Microsoft Excel 2.x
17	Template
18	Add-in macro (For compatibility only. In Microsoft Excel 5.0, this saves as normal.)
19	Text (Macintosh)
20	Text (Windows)
21	Text (MS-DOS)
22	CSV (Macintosh)
23	CSV (Windows)
24	CSV (MS-DOS)
25	International macro
26	International add-in macro
27	Reserved
28	Reserved
29	Microsoft Excel 3.0
30	WK1 / FMT
31	WK1 / Allways
32	WK3 / FM3
33	Microsoft Excel 4.0

34	WQ1
35	Microsoft Excel 4.0 workbook
36	Formatted text (space delimited)

The following table shows which values of type_num apply to the six Microsoft Excel document types.

Document Type	Type_num
Worksheet	All except 10, 12-14, 18, 25-28, 36
Chart sheet	All except 10, 12-14, 18, 25-28
Visual Basic module	1, 3, 17
Dialog	1, 17
Macro sheet	1-3, 6, 9, 16-29, 33
Workbook	1, 15, 35

Prot_pwd corresponds to the Protection Password box in the Save Options dialog box.

- Prot_pwd is a password given as text or as a reference to a cell containing text. Prot_pwd should be no more than 15 characters.
- If a file is saved with a password, the password must be supplied for the file to be opened.

Backup is a logical value corresponding to the Always Create Backup File check box in the Save Options dialog box and specifies whether to make a backup workbook. If backup is TRUE, Microsoft Excel creates a backup file; if FALSE, no backup file is created; if omitted, the status is unchanged.

Write_res_pwd corresponds to the Write Reservation Password box in the Save Options dialog box and allows the user to write to a file. If a file is saved with a password and the password is not supplied when the file is opened, the file is opened read-only.

Read_only_rec is a logical value corresponding to the Read-Only Recommended check box in the Save Options dialog box.

- If read_only_rec is TRUE, Microsoft Excel saves the workbook as a read-only recommended workbook; if FALSE, Microsoft Excel saves the workbook normally; if omitted, Microsoft Excel uses the current settings.
- When you open a workbook that was saved as read-only recommended, Microsoft Excel displays a message recommending that you open the workbook as read-only.

Related Functions

<u>CLOSE</u>	Closes the active window
<u>GET.DOCUMENT</u>	Returns information about a workbook
<u>SAVE</u>	Saves the active workbook
<u>SAVE.WORKBOOK</u>	Saves a workbook
List of <u>Command-Equivalent Functions</u>	

SAVE.TOOLBAR

Macro Sheets Only

Saves one or more toolbar definitions to a specified file.

Syntax

SAVE.TOOLBAR(bar_id, filename)

Bar_id is either the name or number of a toolbar whose definition you want to save or an array of toolbar names or numbers whose definitions you want to save. Use an array to save several toolbar definitions at the same time. For detailed information about bar_id, see ADD.TOOL. If bar_id is omitted, all toolbar definitions are saved.

Filename is text specifying the name of the destination file. If filename does not exist, Microsoft Excel creates a new file. If filename exists, Microsoft Excel overwrites the file. If filename is omitted, Microsoft Excel saves the toolbar or toolbars in EXCEL.XLB (in Microsoft Excel for Windows) or EXCEL TOOLBARS (in Microsoft Excel for the Macintosh).

Examples

In Microsoft Excel for Windows, the following macro formula saves Toolbar6 as \EXCDT\TOOLFILE.XLB.

```
SAVE.TOOLBAR("Toolbar6", "\EXCDT\TOOLFILE.XLB")
```

In Microsoft Excel for the Macintosh, the following macro formula saves Toolbar6 as TOOLFILE.

```
SAVE.TOOLBAR("Toolbar6", "TOOLFILE")
```

Related Functions

<u>ADD.TOOL</u>	Adds one or more tools to a toolbar
<u>ADD.TOOLBAR</u>	Creates a new toolbar with the specified tools
<u>OPEN</u>	Opens a workbook

List of Customizing Functions

SAVE.WORKBOOK

Macro Sheets Only

Equivalent to choosing the Save Workbook command from the File menu in Microsoft Excel version 4.0. Provided for compatibility with Microsoft Excel version 4.0. Saves the workbook to which the active document belongs. To save Microsoft Excel version 5.0 workbooks, see SAVE.AS. For more information about workbooks, see [Overview of creating a new workbook](#).

Syntax

SAVE.WORKBOOK(document_text, type_num, prot_pwd, backup, write_res_pwd, read_only_rec)

SAVE.WORKBOOK?(document_text, type_num, prot_pwd, backup, write_res_pwd, read_only_rec)

For a description of the arguments, see SAVE.AS.

Related Functions

<u>CLOSE</u>	Closes the active window
<u>GET.DOCUMENT</u>	Returns information about a document
<u>SAVE</u>	Saves the active workbook
<u>SAVE.AS</u>	Saves a workbook and allows you to specify the name, file type, password, backup file, and location of the workbook

List of [Command-Equivalent Functions](#)

SELECT

Macro Sheets Only

Equivalent to selecting cells or changing the active cell. There are three syntax forms of SELECT. Use syntax 1 to select a cell on a worksheet or macro sheet; use one of the other syntax forms to select worksheet or macro sheet objects or chart items.

<u>Syntax1</u>	Selects cells
<u>Syntax2</u>	Selects objects on worksheets
<u>Syntax3</u>	Selects chart objects

SELECT Syntax 1

Macro Sheets Only

Equivalent to selecting cells or changing the active cell. There are three syntax forms of SELECT. Use syntax 1 to select a cell on a worksheet or macro sheet; use one of the other syntax forms to select worksheet or macro sheet objects or chart items.

Syntax

SELECT(selection, active_cell)

Selection is the cell or range of cells you want to select. Selection can be a reference to the active worksheet, such as !\$A\$1:\$A\$3 or !Sales, or an R1C1-style reference to a cell or range relative to the active cell in the current selection, such as "R[-1]C[-1]:R[1]C[1]". The reference must be in text form. If selection is omitted, the current selection is used.

Active_cell is the cell in selection you want to make the active cell. Active_cell can be a reference to a single cell on the active worksheet, such as !\$A\$1, or an R1C1-style reference relative to the active cell, such as "R[-1]C[-1]". The reference must be in text form. If active_cell is omitted, SELECT makes the cell in the upper-left corner of selection the active cell.

Remarks

- Active_cell must be within selection. If it is not, an error message is displayed and SELECT returns the #VALUE! error value.
- If you are recording a macro using relative references, Microsoft Excel records the action using R1C1-style relative references in the form of text.
- If you are recording using absolute references, Microsoft Excel records the action using R1C1-style absolute references in the form of text.
- You cannot give an external reference to a specific sheet as the selection argument. The sheet on which you want to make a selection must be active when you use SELECT. Use FORMULA.GOTO to make a selection on a sheet or macro sheet in an external workbook.

Tip You can enter data in a cell without selecting the cell by using the reference arguments to the CUT, COPY, or FORMULA functions.

Examples

The following macro formula selects cells C3:E5 on the active worksheet and makes C5 the active cell:

```
SELECT(!C$3:$E$5, !C$5)
```

If the active cell is C3, the following macro formula selects cells E5:G7 and makes cell F6 the active cell in the selection:

```
SELECT("R[2]C[2]:R[4]C[4]", "R[1]C[1]")
```

You can also make multiple nonadjacent selections with SELECT. The following macro formula selects a number of nonadjacent ranges:

```
SELECT("R1C1, R3C2:R4C3, R8C4:R10C5")
```

The following sequence of macro formulas moves the active cell right, left, down, and up within the selection, just as TAB, SHIFT+TAB, ENTER, and SHIFT+ENTER do:

```
SELECT(, "RC[1]")
SELECT(, "RC[-1]")
SELECT(, "R[1]C")
SELECT(, "R[-1]C")
```

Use SELECT with the OFFSET function to select a new range a specified distance away from the current range. For example, the following macro formula selects a range that is the same size as the current range, one column over:

```
SELECT(OFFSET(SELECTION(), 0, 1))
```


Related Functions

<u>ACTIVE.CELL</u>	Returns the reference of the active cell
<u>SELECT.SPECIAL</u>	Selects a group of cells belonging to a category
<u>SELECTION</u>	Returns the reference of the selection
<u>Syntax2</u>	Selects objects on worksheets
<u>Syntax3</u>	Selects chart objects
List of <u>Command-Equivalent Functions</u>	

SELECT Syntax 2

Macro Sheets Only

Equivalent to selecting objects on a worksheet or macro sheet. There are three syntax forms of SELECT. Use syntax 2 to select an object on which to perform an action; use one of the other syntax forms to select cells on a worksheet or macro sheet or items on a chart.

Syntax

SELECT(object_id_text, replace)

Object_id_text is text that identifies the object to select. Object_id_text can be the name of more than one object. To give the name of more than one object, use the following format:

```
SELECT("Oval 3, Arc 2, Line 4")
```

The last item in the object_id_text list will be the active object. The active object is important when moving and sizing a group of objects. A multiple selection of objects is moved and sized relative to the upper-left corner of the active object.

Replace is a logical value that specifies whether previously selected objects are included in the selection. If replace is TRUE or omitted, Microsoft Excel only selects the objects specified by object_id_text; if FALSE, it includes any objects that were previously selected. For example, if a button is selected and a SELECT formula selects an arc and an oval, TRUE leaves only the arc and oval selected, and FALSE includes the button with the arc and oval.

Remarks

Objects can be identified by their object type and number as described in CREATE.OBJECT, or by the unique number that specifies the order of their creation. For example, if the third object you create is an oval, you could use either "oval 3" or "3" as object_id_text.

Examples

The following macro formulas each select a number of objects and specify Arc 2 as the active object:

```
SELECT("Oval 3, Arc 1, Line 4, Arc 2")
```

```
SELECT("3, 1, 4, 2")
```

Related Functions

<u>FORMAT.MOVE</u>	Moves the selected object
<u>FORMAT.SIZE</u>	Changes the size of the selected objects
<u>GET.OBJECT</u>	Returns information about an object
<u>SELECTION</u>	Returns the reference of the selection
<u>Syntax1</u>	Selects cells
<u>Syntax3</u>	Selects chart objects
List of <u>Command-Equivalent Functions</u>	

SELECT Syntax 3

Macro Sheets Only

Selects a chart object as specified by the selection code `item_text`. There are three syntax forms of `SELECT`. Use syntax 3 to select a chart item to which you want to apply formatting; use one of the other syntax forms to select cells or objects on a worksheet or macro sheet.

Syntax

SELECT(`item_text`, `single_point`)

`Item_text` is a selection code from the following table which specifies which chart object to select.

To select	Item_text
Entire chart	"Chart"
Plot area	"Plot"
Legend	"Legend"
Primary chart value axis	"Axis 1"
Primary chart category axis	"Axis 2"
Secondary chart value axis or 3-D series axis	"Axis 3"
Secondary chart category axis	"Axis 4"
Chart title	"Title"
Label for the primary chart value axis	"Text Axis 1"
Label for the primary chart category axis	"Text Axis 2"
Label for the primary chart series axis	"Text Axis 3"
nth floating text item	"Text n"
nth arrow	"Arrow n"
Major gridlines of value axis	"Gridline 1"
Minor gridlines of value axis	"Gridline 2"
Major gridlines of category axis	"Gridline 3"
Minor gridlines of category axis	"Gridline 4"
Major gridlines of series axis	"Gridline 5"
Minor gridlines of series axis	"Gridline 6"
Primary chart droplines	"Dropline 1"
Secondary chart droplines	"Dropline 2"
Primary chart hi-lo lines	"Hiloline 1"
Secondary chart hi-lo lines	"Hiloline 2"
Primary chart up bar	"UpBar1"
Secondary chart up bar	"UpBar2"
Primary chart down bar	"DownBar1"
Secondary chart down bar	"DownBar2"
Primary chart series line	"Seriesline1"
Secondary chart series line	"Seriesline2"
Entire series	"Sn"
Data associated with point m in series n if <code>single_point</code> is TRUE	"SnPm"
Text attached to point m of series n	"Text SnPm"
Series title text of series n of an area chart	"Text Sn"
Base of a 3-D chart	"Floor"
Back of a 3-D chart	"Walls"
Corners of a 3-D chart	"Corners"
Trend line	"SnTm"
Error bars	"SnEm"
Legend Marker	"Legend Marker n"

Legend Entry

"Legend Entry n"

For trend lines and error bars, the value m can be X or Y, depending on which point you want to select. If m is blank, selects both.

Single_point is a logical value that determines whether to select a single point. Single_point is available only when item_text is "SnPm".

- If single_point is TRUE, Microsoft Excel selects a single point.
- If single_point is FALSE or omitted, Microsoft Excel selects a single point if there is only one series in the chart or selects the entire series if there is more than one series in the chart.
- If you specify single_point when item_text is any value other than "SnPm", SELECT returns an error value.

Examples

SELECT("Chart") selects the entire chart.

SELECT("Dropline 2") selects the droplines of an overlay chart.

SELECT("S1P3", TRUE) selects the third point in the first series.

SELECT("Text S1") selects the series title text of the first series in an area chart.

Related Function

SELECTION Returns the reference of the selection

Syntax1 Selects cells

Syntax2 Selects objects on worksheets

List of Command-Equivalent Functions

SELECT.SPECIAL

Macro Sheets Only

Equivalent to choosing the Go To command from the Edit menu and then selecting the Special button. Use SELECT.SPECIAL to select groups of similar cells in one of a variety of categories.

Syntax

SELECT.SPECIAL(type_num, value_type, levels)

SELECT.SPECIAL?(type_num, value_type, levels)

Type_num is a number from 1 to 13 corresponding to options in the Select Special dialog box and describes what to select.

Type_num	Description
1	Notes
2	Constants
3	Formulas
4	Blanks
5	Current region
6	Current array
7	Row differences
8	Column differences
9	Precedents
10	Dependents
11	Last cell
12	Visible cells only (outlining)
13	All objects

Value_type is a number specifying which types of constants or formulas you want to select.

Value_type is available only when type_num is 2 or 3.

Value_type	Selects
1	Numbers
2	Text
4	Logical values
16	Error values

These values can be added to select more than one type. The default for value_type is 23, which select all value types.

Levels is a number specifying how precedents and dependents are selected. Levels is available only when type_num is 9 or 10. The default is 1.

Levels	Selects
1	Direct only
2	All levels

Related Function

List of [Command-Equivalent Functions](#)

SEND.TO.BACK

Macro Sheets Only

Equivalent to choosing the Send To Back command from the Placement submenu on the Format menu. Sends the selected object or objects to the back. Use SEND.TO.BACK to position selected objects behind other objects.

If the selection is not an object or a group of objects, SEND.TO.BACK returns the #VALUE! error value and interrupts the macro.

Syntax

SEND.TO.BACK()

Related Function

BRING.TO.FRONT Brings selected objects to the front

List of Command-Equivalent Functions

SET.PRINT.AREA

Macro Sheets Only

Defines the print area for the workbookthe area that prints when you choose the Print command from the File menu. Equivalent to entering a range in the Print Area edit box on the Sheet tab in the Page Setup dialog box, which appears when you choose the Page Setup command from the File menu.

Syntax

SET.PRINT.AREA(range)

Range is the reference to the range that you want to be printed. If you specify no range by using a set of empty quotemarks (""), deletes the print area.

Remarks

- If you use SET.PRINT.AREA with a multiple selection and then use the PRINT function, the individual selections are printed one after the other in the order they were selected.
- To resume printing the entire worksheet, choose the Page Setup command from the File menu and choose the Sheet tab. Then delete the range in the Print Area edit box.

Related Functions

[PRINT](#)

Prints the active workbook

[SET.PRINT.TITLES](#)

Identifies text to print as titles

List of [Command-Equivalent Functions](#)

SET.PRINT.TITLES

Macro Sheets Only

Defines the print titles for the sheet. Use SET.PRINT.TITLES if you want Microsoft Excel to print the titles whenever it prints any cells in a row or column that intersect the print titles area; a cell need only share the row or column with a print title for the title to be printed above or to the left of that cell.

Syntax

SET.PRINT.TITLES(titles_for_cols_ref, titles_for_rows_ref)

SET.PRINT.TITLES?(titles_for_cols_ref, titles_for_rows_ref)

Titles_for_cols_ref is a reference to the row to be used as a title for columns.

- If you specify part of a row, Microsoft Excel expands the title to a full row.
- If you omit titles_for_cols_ref, Microsoft Excel uses the existing row of column titles, if any.
- If you specify empty text (""), Microsoft Excel removes the row from the print titles definition.

Titles_for_rows_ref is a reference to the column to be used as a title for rows.

- If you specify part of a column, Microsoft Excel expands the title to a full column.
- If you omit titles_for_rows_ref, Microsoft Excel uses the existing column of row titles, if any.
- If you specify empty text (""), Microsoft Excel removes the column from the print titles definition.

Remarks

- SET.PRINT.TITLES operates on the current sheet. If you specify a range that is invalid for the current sheet, Microsoft Excel returns the #VALUE error value.
- The print titles selection can be a multiple selection. Microsoft Excel names this selection Print_Titles when SET.PRINT.TITLES is run.

Related Functions

DEFINE.NAME Defines a name on the active worksheet or macro sheet

PRINT Prints the active sheet

SET.PRINT.AREA Defines the print area

List of Command-Equivalent Functions

SET.VALUE

Macro Sheets Only

Changes the value of a cell or cells on the macro sheet (not the worksheet) without changing any formulas entered in those cells. Use SET.VALUE to assign initial values and to store values during the calculation of a macro. SET.VALUE is especially useful for initializing a dialog box and the conditional test in a WHILE loop. SET.VALUE assigns values to a specific reference or to the name of a reference that has already been defined. For information about creating a new name or entering data on a worksheet, see "Remarks" later in this topic.

Syntax

SET.VALUE(reference, values)

Reference specifies the cell or cells on the macro sheet to which you want to assign a new value or values. If the cell is empty, enters the value in the cell.

- If a cell in reference previously contained a formula, the formula is not changed, but the value of the cell might change. See the second example following.
- If reference is a reference to a range of cells, rather than to a single cell, then values should be an array of the same size. If not, Microsoft Excel expands it into multiple values using the normal rules for expanding arrays. See the third example following.

Values is the value or set of values to which you want to assign the cell or cells in reference.

Remarks

Consider the following guidelines as you choose a function to set values on a worksheet or macro sheet:

- Use SET.VALUE to assign initial values to a reference (including names that have already been defined) on a macro sheet, and to store values during the calculation of a macro.
- Use FORMULA to enter values in a worksheet cell.
- Use SET.NAME to change the value of a name on a macro sheet (the name is created if it does not already exist). For more information, see SET.NAME.
- Use DEFINE.NAME to create or change the value of a name on a worksheet.

Examples

The following macro formula changes the value of cell A1 on the macro sheet to 1:

```
SET.VALUE($A$1, 1)
```

Suppose the name TempAverage refers to a cell containing the formula AVERAGE(Temp1, Temp2, Temp3). The following formula assigns the value 99 to this cell, even if the average of the arguments is not 99, without changing the formula in TempAverage:

```
SET.VALUE(TempAverage, 99)
```

The preceding formula is useful if a WHILE loop or some other conditional test depends on TempAverage and you want to force the conditional test to have a particular result. Of course, TempAverage is restored to its correct value as soon as it is recalculated. (Recall that unlike formulas in a worksheet, formulas in a macro sheet are not recalculated until the macro actually uses them.)

The following macro formula stores the values 1, 2, 3, and 4 in cells A1:B2:

```
SET.VALUE($A$1:$B$2, {1, 2;3, 4})
```

Related Functions

<u>DEFINE.NAME</u>	Defines a name on the active worksheet or macro sheet
<u>FORMULA</u>	Enters values into a cell or range or onto a chart
<u>SET.NAME</u>	Defines a name as a value
List of <u>Control Functions</u>	

SHORT.MENUS

Macro Sheets Only

Equivalent to choosing the Short Menus command from the Options menu or the Chart menu in Microsoft Excel version 3.0 or earlier.

Syntax

SHORT.MENUS(logical)

Related Functions

List of [Command-Equivalent Functions](#)

SHOW.CLIPBOARD

Macro Sheets Only

Displays the contents of the Clipboard in a new window.

Syntax

SHOW.CLIPBOARD()

Remarks

- In Microsoft Excel for Windows, the Clipboard must already be running if you want to display its contents in a new window. If it is not already running, you must run the SHOW.CLIPBOARD function twice, once to start the Clipboard application and again to display it in a new window.
- If the Clipboard contains cells, the window shows the size of the Clipboard contents in rows and columns. If the Clipboard contains text cut from the formula bar, the window displays the text.

Related Functions

List of [Command-Equivalent Functions](#)

SHOW.INFO

Macro Sheets Only

Controls the display of the Info window.

Syntax

SHOW.INFO(logical)

Logical controls the display of the Info window.

- If logical is TRUE, Microsoft Excel switches to the Info window.
- If the current window is the Info window and logical is FALSE, Microsoft Excel switches to the workbook linked to the Info window.

Tip Before using SHOW.INFO, use ACTIVATE to switch to the workbook you want information about and SELECT or FORMULA.GOTO to select the cell or range you want information about.

Remarks

The Info window is not available when a chart is active. If SHOW.INFO is carried out when a chart is the active sheet, an Info window will be displayed for the worksheet or macro sheet that was most recently active.

Related Function

FORMULA.GOTO Selects a named area or reference on any open workbook

GET.CELL Returns information about the specified cell

SELECT Selects a cell, worksheet object, or chart item

List of Command-Equivalent Functions

STEP

Macro Sheets Only

Stops the normal flow of a macro and calculates it one cell at a time. Running a macro one cell at a time is called single-stepping and is very useful when you are debugging a macro. Use the STEP function, instead of choosing the Step button in the Macro dialog box when you want to start single-stepping at a specific line in a macro. The Macro dialog box appears when you choose the Macro command on the Tools menu.

Syntax

STEP()

Remarks

- When Microsoft Excel encounters a STEP function, it stops running the macro and displays a dialog box. The dialog box tells you which cell in the macro Microsoft Excel is about to calculate, and what formula is in that cell. You can choose Step to carry out the next instruction; choose Evaluate to calculate part of the formula; choose Halt to interrupt the macro; or choose Continue to continue the macro without single-stepping.

You can evaluate the formula by holding down the SHIFT key while you click the Step button. You can also choose Step Over to carry out, but not step through, a user-defined function call; choose Pause, to suspend the macro so you can perform other tasks; and choose Goto, to stop the macro and select the cell being evaluated.

- When placed at the beginning of a macro, STEP is equivalent to choosing the Macro command from the Tool menu and selecting the Step button in the Macro dialog box.

- You can start single-stepping while a macro is running by pressing ESC in Microsoft Excel for Windows or by pressing ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh.

- The Single Step dialog box is initially displayed in the lower-right corner of the screen. You can move the dialog box if it's in your way. If you move it, it will remain in the new location until you stop single-stepping.

- To step through the calculation of a custom function, place the STEP function at the start of the custom function.

Related Functions

HALT Stops all macros from running

RUN Runs a macro

List of Control Functions

TABLE

Macro Sheets Only

Equivalent to choosing the Table command from the Data menu. Creates a table based on the input values and formulas you define on a worksheet. Use data tables to perform a "what-if" analysis by changing certain constant values in your workbook to see how values in other cells are affected.

For information about data tables, see [Projecting figures using a data table](#).

Syntax

TABLE(row_ref, column_ref)

TABLE?(row_ref, column_ref)

Row_ref specifies the one cell to use as the row input for your table.

- Row_ref should be either an external reference to a single cell on the active worksheet, such as !\$A\$1 or !Price, or an R1C1-style reference to a single cell in the form of text, such as "R1C1", "R[-1]C[-1]", or "Price".
- If row_ref is an R1C1-style reference, it is assumed to be relative to the active cell in the selection.

Column_ref specifies the one cell to use as the column input for your table. Column_ref is subject to the same restrictions as row_ref.

Related Functions

List of [Command-Equivalent Functions](#)

VSCROLL

Macro Sheets Only

Vertically scrolls through the active sheet by percentage or by row number.

Syntax

VSCROLL(position, row_logical)

Position specifies the row you want to scroll to. Position can be an integer representing the row number or a fraction or percentage representing the vertical position of the row in the sheet. If position is 0, VSCROLL scrolls through your sheet to its top edge, which is row 1. If position is 1, VSCROLL scrolls through your sheet to its bottom edge, which is row 16,384. For charts that do not size with the window, use a fraction or percentage.

Row_logical is a logical value specifying how the function scrolls.

- If row_logical is TRUE, VSCROLL scrolls through the sheet to row position.
- If row_logical is FALSE or omitted, VSCROLL scrolls through the sheet to the vertical position represented by the fraction position.

Remarks

- To scroll to a specific row n, either use VSCROLL(n, TRUE) or VSCROLL(n/16384). To scroll to row 138, for example, enter VSCROLL(138, TRUE) or VSCROLL(138/16384).
- If you are recording a macro and move the scroll box several times in a row, the recorder only records the final location of the scroll box, omitting any intermediate steps. Remember that scrolling does not change the active cell or the selection.

Related Functions

<u>FORMULA.GOTO</u>	Selects a named area or reference on any open workbook
<u>HLINE</u>	Horizontally scrolls through the active window by columns
<u>HPAGE</u>	Horizontally scrolls through the active window one window at a time
<u>HSCROLL</u>	Horizontally scrolls through a sheet by percentage or by column number
<u>SELECT</u>	Selects a cell, worksheet object, or chart item
<u>VLINE</u>	Vertically scrolls through the active window by rows
<u>VPAGE</u>	Vertically scrolls through the active window one window at a time
List of <u>Command-Equivalent Functions</u>	

WINDOW.MAXIMIZE

Macro Sheets Only

Changes the active window from its normal size to full size. In Microsoft Excel for Windows, using WINDOW.MAXIMIZE is equivalent to pressing CTRL+F10 or double-clicking the title bar. In Microsoft Excel for the Macintosh, using WINDOW.MAXIMIZE is equivalent to double-clicking the title bar or clicking the zoom box.

Syntax

WINDOW.MAXIMIZE(window_text)

Window_text specifies which window to switch to and maximize. Window_text is text enclosed in quotation marks or a reference to a cell containing text. If window_text is omitted, the active window is maximized.

Remarks

WINDOW.MAXIMIZE replaces FULL(TRUE) in earlier versions of Microsoft Excel.

Related Functions

<u>WINDOW.MINIMIZE</u>	Minimizes a window
<u>WINDOW.MOVE</u>	Moves a window
<u>WINDOW.RESTORE</u>	Restores a window to its previous size
<u>WINDOW.SIZE</u>	Changes the size of a window
List of <u>Command-Equivalent Functions</u>	

WINDOW.MINIMIZE

Macro Sheets Only

Shrinks a window to an icon. In Microsoft Excel for Windows, using WINDOW.MINIMIZE is equivalent to clicking the minimize button on a workbook window. In Microsoft Excel for the Macintosh, the minimize feature is not supported.

Syntax

WINDOW.MINIMIZE(window_text)

Window_text specifies which window to minimize.

- Window_text is text enclosed in quotation marks or a reference to a cell containing text.
- If window_text is omitted, Microsoft Excel minimizes the active window.

Remarks

If a window is already minimized, WINDOW.MINIMIZE has no effect.

Related Functions

WINDOW.MAXIMIZE

Maximizes a window

WINDOW.MOVE

Moves a window

WINDOW.RESTORE

Restores a window to its previous size

WINDOW.SIZE

Changes the size of a window

List of Command-Equivalent Functions

WINDOW.MOVE

Macro Sheets Only

Equivalent to choosing the Move command from the Control menu in Microsoft Excel for Windows or moving a window by dragging its title bar or its icon. Moves the active window so that its upper-left corner is at the specified horizontal and vertical positions. The dialog-box form, WINDOW.MOVE?, is supported only in Microsoft Excel for Windows.

Syntax

WINDOW.MOVE(x_pos, y_pos, window_text)

WINDOW.MOVE?(x_pos, y_pos, window_text)

X_pos is the horizontal position to which you want to move the window. X_pos is measured in points. A point is 1/72nd of an inch.

- In Microsoft Excel for Windows, x_pos is measured from the left edge of your workspace to the left edge of the window.
- In Microsoft Excel for the Macintosh, x_pos is measured from the left edge of your screen to the left edge of the window.
- If x_pos is omitted, the window does not move horizontally.

Y_pos is the vertical position to which you want to move the window. Y_pos is measured in points from the bottom edge of the formula bar to the top edge of the window. If y_pos is omitted, the window does not move vertically.

Window_text specifies which window to restore.

- Window_text is text enclosed in quotation marks or a reference to a cell containing text.
- If window_text is omitted, it is assumed to be the name of the active window.

Remarks

- If the window is minimized, WINDOW.MOVE moves the icon on the workspace. Measurements are relative to the upper-left corner of the workspace and the icon.
- WINDOW.MOVE does not change the size of the window or affect whether the specified window is active or inactive.
- In Microsoft Excel for the Macintosh, if window_text is "Clipboard", WINDOW.MOVE moves the Clipboard. The Clipboard must already be available; if it is not available, use the SHOW.CLIPBOARD function before using the WINDOW.MOVE function.
- WINDOW.MOVE replaces MOVE in earlier versions of Microsoft Excel.

Related Functions

<u>FORMAT.MOVE</u>	Moves the selected object
<u>WINDOW.MAXIMIZE</u>	Maximizes a window
<u>WINDOW.MINIMIZE</u>	Minimizes a window
<u>WINDOW.RESTORE</u>	Restores a window to its previous size
<u>WINDOW.SIZE</u>	Changes the size of a window

List of Command-Equivalent Functions

WINDOW.RESTORE

Macro Sheets Only

Changes the active window from maximized or minimized size to its previous size. In Microsoft Excel for Windows, using WINDOW.RESTORE is equivalent to pressing CTRL+F5 or double-clicking the title bar (or double-clicking the icon if it is minimized). In Microsoft Excel for the Macintosh, using WINDOW.RESTORE is equivalent to double-clicking the title bar or clicking the zoom box.

Syntax

WINDOW.RESTORE(window_text)

Window_text specifies which window to switch to and restore.

- Window_text is text enclosed in quotation marks or a reference to a cell containing text.
- If window_text is omitted, Microsoft Excel restores the active window.

Remarks

- If the window is minimized, WINDOW.RESTORE restores the icon to its previous size. This operation is equivalent to double-clicking the icon.
- WINDOW.RESTORE replaces FULL(FALSE) in earlier versions of Microsoft Excel.

Related Functions

WINDOW.MAXIMIZE

Maximizes a window

WINDOW.MINIMIZE

Minimizes a window

WINDOW.MOVE

Moves a window

WINDOW.SIZE

Changes the size of a window

List of Command-Equivalent Functions

WINDOW.SIZE

Macro Sheets Only

Equivalent to choosing the Size command from the Control menu or to adjusting the sizing borders (in Microsoft Excel for Windows) or the sizing box (in Microsoft Excel for the Macintosh) of the window with the mouse. Changes the size of the active window by moving its lower-right corner so that the window has the width and height you specify. WINDOW.SIZE does not change the position of the upper-left corner of the window, nor does it affect whether the specified window is active or inactive.

Syntax

WINDOW.SIZE(width, height, window_text)

WINDOW.SIZE?(width, height, window_text)

Width specifies the width of the window and is measured in points. A point is 1/72nd of an inch.

Height specifies the height of the window and is measured in points.

Window_text specifies which window to size.

- Window_text is text enclosed in quotation marks or a reference to a cell containing text.
- If window_text is omitted, it is assumed to be the name of the active window.

Remarks

- In Microsoft Excel for Windows, an error occurs if you try to resize a window that has already been minimized to an icon or enlarged to its maximum size. You must first restore the window to its original size using the WINDOW.RESTORE formula. For more information, see WINDOW.RESTORE.
- WINDOW.SIZE replaces SIZE in earlier versions of Microsoft Excel.

Related Functions

FORMAT.SIZE

Sizes an object

WINDOW.MAXIMIZE

Maximizes a window

WINDOW.MINIMIZE

Minimizes a window

WINDOW.MOVE

Moves a window

WINDOW.RESTORE

Restores a window to its previous size

List of Command-Equivalent Functions

ABSREF

Macro Sheets Only

Returns the absolute reference of the cells that are offset from a reference by a specified amount. You should generally use OFFSET instead of ABSREF. This function is provided for users who prefer to supply a absolute reference in text form.

Syntax

ABSREF(ref_text, reference)

Ref_text specifies a position relative to reference. Think of ref_text as "directions" from one range of cells to another.

- Ref_text must be an R1C1-style relative reference in the form of text, such as "R[1]C[1]".
- Ref_text is considered relative to the cell in the upper-left corner of reference.

Reference is a cell or range of cells specifying a starting point that ref_text uses to locate another range of cells. Reference can be an external reference.

Remarks

- If you use ABSREF in a function or operation, you will usually get the values contained in the reference instead of the reference itself because the reference is automatically converted to the contents of the reference.
- If you use ABSREF in a function that requires a reference argument, then Microsoft Excel does not convert the reference to a value.
- If you want to work with the actual reference, use the REFTTEXT function to convert the active-cell reference to text, which you can then store or manipulate (or convert back to a reference with TEXTREF). See the third example following.

Examples

ABSREF("R[-2]C[-2]", C3) equals \$A\$1

ABSREF(RELREF(A1, C3), D4) equals \$B\$2

REFTTEXT(ABSREF("R[-2]C[-2]:R[2]C[2]", C3:G7), TRUE) is equivalent to

REFTTEXT(ABSREF("R[-2]C[-2]:R[2]C[2]", C3), TRUE), which equals "\$A\$1:\$E\$5"

In Microsoft Excel for Windows ABSREF("R[-2]C[-2]", [FINANCE.XLS]Sheet1!C3) equals [FINANCE.XLS]Sheet1!\$A\$1.

In Microsoft Excel for the Macintosh ABSREF("R[-2]C[-2]", [FINANCE]Sheet1!C3) equals [FINANCE]Sheet1!\$A\$1

Related Functions

OFFSET Returns a reference offset from a given reference

RELREF Returns a relative reference

List of Lookup & Reference Functions

ADD.ARROW

Macro Sheets Only

Equivalent to clicking the Arrow button on the Drawing toolbar. Adds an arrow to the active chart. If a chart is not the active window, displays an error value.

Syntax

ADD.ARROW()

Remarks

After you create an arrow with ADD.ARROW, the arrow remains selected, so you can use the arrow form of the PATTERNS function to format the arrow and the FORMAT.MOVE and FORMAT.SIZE functions to change the position and size of the arrow.

Related Functions

<u>CREATE.OBJECT</u>	Creates an object
<u>DELETE.ARROW</u>	Deletes the selected arrow
<u>FORMAT.MOVE</u>	Moves the selected object
<u>FORMAT.SIZE</u>	Changes the size of the selected object
<u>PATTERNS</u>	Changes the appearance of the selected object
List of <u>Command-Equivalent Functions</u>	

ADD.CHART.AUTOFORMAT

Adds the format of the currently active chart in the current window to the list of custom formats in the AutoFormat dialog box for charts.

Syntax

ADD.CHART.AUTOFORMAT(name_text, desc_text)

Name_text is the name you want to appear in the list of custom formats.

Desc_text is the description you want to appear when the custom format is selected.

Related Functions

DELETE.CHART.AUTOFORMAT Deletes a custom template

List of Command-Equivalent Functions

ADD.OVERLAY

Macro Sheets Only

Equivalent to choosing the Add Overlay command from the Chart menu in Microsoft Excel 4.0. Adds an overlay to a 2-D chart. If the active chart already has an overlay, ADD.OVERLAY takes no action and returns TRUE. In Microsoft Excel 5.0, ADD.OVERLAY works with charts that have only one chart type.

Syntax

ADD.OVERLAY()

Related Functions

ADD.ARROW Adds an arrow to a chart

LEGEND Adds a legend to a chart

List of Command-Equivalent Functions

APPLY.NAMES

Macro Sheets Only

Equivalent to choosing the Apply command from the Name submenu on the Insert menu. Replaces definitions with their respective names. If no names are defined in the current selection, APPLY.NAMES returns the #VALUE! error value. Use APPLY.NAMES to replace references and values in formulas with names.

Syntax

APPLY.NAMES(name_array, ignore, use_rowcol, omit_col, omit_row, order_num, append_last)

APPLY.NAMES?(name_array, ignore, use_rowcol, omit_col, omit_row, order_num, append_last)

Name_array is the name or names to apply as text elements in an array.

- To give more than one name as the argument, you must use an array. For example:
APPLY.NAMES({"DataRange", "CriteriaRange"})
- If the names indicated by the argument name_array have already replaced all of the appropriate references or values, the #VALUE! error value is returned.

The next four arguments correspond to check boxes and options in the Apply Names dialog box.

Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Ignore corresponds to the Ignore Relative/Absolute check box.

Use_rowcol corresponds to the Use Row And Column Names check box. If use_rowcol is FALSE, the next three arguments are ignored.

Omit_col corresponds to the Omit Column Name If Same Column check box.

Omit_row corresponds to the Omit Row Name If Same Row check box.

Order_num determines which range name is listed first when a cell reference is replaced by a row-oriented and a column-oriented range name, as shown in the following table.

Order_num	Order of range names
1	Row Column
2	Column Row

Append_last determines whether the names most recently defined are also replaced.

- If append_last is TRUE, Microsoft Excel replaces the definitions of the names in name_array and also replaces the definitions of the last names defined.
- If append_last is FALSE or omitted, Microsoft Excel replaces the definitions of the names in name_array only.

Related Functions

CREATE.NAMES Creates names automatically from text labels on a sheet

DEFINE.NAME Defines a name on the active sheet or macro sheet

LIST.NAMES Lists names and their associated information

List of Command-Equivalent Functions

BORDER

Macro Sheets Only

Equivalent to choosing the Border tab in the Format Cells dialog box, which appears when you choose the Cells command from the Format menu. Adds a border to the selected cell or range of cells.

Syntax

BORDER(outline, left, right, top, bottom, shade, outline_color, left_color, right_color, top_color, bottom_color)

BORDER?(outline, left, right, top, bottom, shade, outline_color, left_color, right_color, top_color, bottom_color)

Outline, left, right, top, and bottom are numbers from 0 to 7 corresponding to the line styles in the Border dialog box, as shown in the following table.

Argument	Line type
0	No border
1	Thin line
2	Medium line
3	Dashed line
4	Dotted line
5	Thick line
6	Double line
7	Hairline

Note For compatibility with earlier versions of Microsoft Excel, TRUE and FALSE values for the above arguments create a thin border or no border, respectively.

Shade corresponds to the Shade check box in the Border dialog box of Microsoft Excel version 4.0. This argument is included for compatibility only.

Outline_color, left_color, right_color, top_color, and bottom_color are numbers from 1 to 56 corresponding to the Color box in the Border dialog box. Zero corresponds to automatic color.

Related Function

List of [Command-Equivalent Functions](#)

CALLER

Macro Sheets Only

Returns information about the cell, range of cells, command on a menu, tool on a toolbar, or object that called the macro that is currently running. Use CALLER in a subroutine or custom function whose behavior depends on the location, size, name, or other attribute of the caller.

Syntax

CALLER()

Remarks

- If the custom function is entered in a single cell, CALLER returns the reference of that cell.
- If the custom function was part of an array formula entered in a range of cells, CALLER returns the reference of the range.
- If CALLER appears in a macro called by an Auto_Open, Auto_Close, Auto_Activate, or Auto_Deactivate macro, it returns the name of the calling sheet.
- If CALLER appears in a macro called by a command on a menu, it returns a horizontal array of three elements including the command's position number, the menu number, and the menu bar number.
- If CALLER appears in a macro called by an assigned-to-object macro, it returns the object identifier.
- If CALLER appears in a macro called by a tool on a toolbar, it returns a horizontal array containing the position number and the toolbar name.
- If CALLER appears in a macro called by an ON.DOUBLECLICK or ON.ENTRY function, CALLER returns the name of the chart object identifier or cell reference, if applicable, to which the ON.DOUBLECLICK or ON.ENTRY macro applies.
- If CALLER appears in a macro that was run manually, or for any reason not described above, it returns the #REF! error value.

Examples

If the custom function MACROS!VALUEONE is entered in cell B3 on a sheet named SALES, the nested CALLER function returns the following values.

Nested function	Returns
COLUMN(CALLER())	2
COLUMNS(CALLER())	1
GET.CELL(1, CALLER())	SALES!\$B\$3
ROW(CALLER())	3
ROWS(CALLER())	1

If the same custom function was entered into an array in cells B2:C3, the following values would be returned.

Nested function	Returns
COLUMN(CALLER())	2
COLUMNS(CALLER())	2
ROW(CALLER())	2
ROWS(CALLER())	2

Related Functions

<u>COLUMN</u>	Returns the column number of a reference
<u>COLUMNS</u>	Returns the number of columns in a reference
<u>GET.BAR</u>	Returns the name or position number of menu bars, menus, and commands
<u>GET.CELL</u>	Returns information about the specified cell
<u>ROW</u>	Returns the row number of a reference
<u>ROWS</u>	Returns the number of rows in a reference

List of Information Functions

CANCEL.KEY

Macro Sheets Only

Disables macro interruption, or specifies a macro to run when a macro is interrupted. Use CANCEL.KEY to control what happens when a macro is interrupted.

Syntax

CANCEL.KEY(enable, macro_ref)

Enable specifies whether the macro can be interrupted by pressing ESC in Microsoft Excel for Windows or ESC or COMMAND+PERIOD in Microsoft Excel for the Macintosh.

If enable is	Then
FALSE	Pressing ESC or COMMAND+PERIOD does not interrupt a macro
TRUE and macro_ref is omitted	Pressing ESC or COMMAND+PERIOD interrupts a macro
TRUE and macro_ref is specified	Macro_ref runs when ESC or COMMAND+PERIOD is pressed

Macro_ref is a reference to a macro, as a cell reference or a name, that runs when enable is TRUE and ESC or COMMAND+PERIOD is pressed.

Remarks

- CANCEL.KEY affects only the macro that is currently running. Once the macro is stopped by a RETURN or HALT function, ESC or COMMAND+PERIOD is reactivated.
- When CANCEL.KEY is in effect, users can still cancel a dialog box displayed while the macro is running.

Examples

The following macro formula prevents the macro from being interrupted by pressing ESC or COMMAND+PERIOD:

```
CANCEL.KEY (FALSE)
```

The following macro formula reactivates ESC or COMMAND+PERIOD to cancel macro execution:

```
CANCEL.KEY (TRUE)
```

The following line in a macro runs CheckCancel when ESC or COMMAND+PERIOD is pressed:

```
CANCEL.KEY (TRUE, CheckCancel)
```

Related Functions

- ERROR Specifies an action to take if an error occurs while a macro is running
 - ON.KEY Runs a macro when a specified key is pressed
 - ON.TIME Runs a macro at a specified time
- List of Customizing Functions

CHECK.COMMAND

Macro Sheets Only

Adds or removes a check mark to or from a command name on a menu. A check mark beside a command indicates that the command has been chosen.

Syntax

CHECK.COMMAND(bar_num, menu, command, check, position)

Bar_num is the menu bar containing the command. Bar_num can be the ID number of a built-in or custom menu bar.

Menu is the menu containing the command. Menu can be either the name of a menu as text or the number of a menu. Menus are numbered starting with 1 from the left of the screen.

Command is the command you want to check or the submenu containing the command you want to check. Command can be the name of the command as text or the number of the command; the first command on a menu is in position 1.

Check is a logical value corresponding to the check mark. If check is TRUE, Microsoft Excel adds a check mark to the command; if FALSE, Microsoft Excel removes the check mark.

position is the name of a command on a submenu that you want to check.

Remarks

- The check mark doesn't affect execution of the command. Microsoft Excel automatically adds and deletes check marks to some commands, such as the name of the active workbook in the Window menu. If you have assigned a check mark to a built-in command that Microsoft Excel automatically changes in response to the user's actions, the check mark will be added or removed as appropriate, and any check marks you have added or deleted with CHECK.COMMAND will be ignored.
- If you use CHECK.COMMAND with a command on a Microsoft Excel 4.0 menu bar, the corresponding command on the Microsoft Excel 5.0 menu bar will not be effected.

Example

The following macro formula adds a check mark to the Sales command on the Weekly menu on a custom menu bar created by the ADD.BAR function in a cell named Reports:

```
CHECK.COMMAND(Reports, "Weekly", "Sales", TRUE)
```

Related Functions

[ADD.COMMAND](#)

Adds a command to a menu

[DELETE.COMMAND](#)

Deletes a command from a menu

[ENABLE.COMMAND](#)

Enables or disables a menu or custom command

[RENAME.COMMAND](#)

Changes the name of a command or menu

List of [Customizing Functions](#)

CLOSE.ALL

Macro Sheets Only

Equivalent to choosing the Close All command from the File menu. The Close All command appears when you hold down SHIFT while selecting the File menu. Closes all protected and unprotected windows and all hidden windows. If unsaved changes have been made to the workbook in one or more windows, a message is displayed asking if you want to save each workbook.

Syntax

CLOSE.ALL()

Related Functions

<u>CLOSE</u>	Closes the active window
<u>FILE.CLOSE</u>	Closes the active workbook
<u>QUIT</u>	Ends a Microsoft Excel session
<u>SAVE</u>	Saves the active workbook
List of <u>Command-Equivalent Functions</u>	

CREATE.NAMES

Macro Sheets Only

Equivalent to choosing the Create command from the Name submenu on the Insert menu. Use CREATE.NAMES to quickly create names from text labels on a sheet.

Arguments are logical values corresponding to check boxes in the Create Names dialog box. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE or omitted, Microsoft Excel clears the check box.

Syntax

CREATE.NAMES(top, left, bottom, right)

CREATE.NAMES?(top, left, bottom, right)

Top corresponds to the Top Row check box.

Left corresponds to the Left Column check box.

Bottom corresponds to the Bottom Row check box.

Right corresponds to the Right Column check box.

Remarks

The cell containing the label text that Microsoft Excel uses to create the names is not included in the resulting named range.

Related Functions

APPLY.NAMES Replaces references and values with their corresponding names

DEFINE.NAME Defines a name on the active sheet or macro sheet

DELETE.NAME Deletes a name

FORMULA.GOTO Selects a named area or reference on any open workbook

List of Command-Equivalent Functions

CREATE.OBJECT

Macro Sheets Only

Draws an object on a sheet or macro sheet and returns a value identifying the object created. It is generally easier to use the macro recorder to enter this function on your macro sheet.

Syntax 1

Lines, rectangles, ovals, arcs, pictures, text boxes, and buttons

CREATE.OBJECT(obj_type, ref1, x_offset1, y_offset1, ref2, x_offset2, y_offset2, text, fill, editable)

Syntax 2

Polygons

CREATE.OBJECT(obj_type, ref1, x_offset1, y_offset1, ref2, x_offset2, y_offset2, array, fill)

Syntax 3

Embedded charts

CREATE.OBJECT(obj_type, ref1, x_offset1, y_offset1, ref2, x_offset2, y_offset2, xy_series, fill, gallery_num, type_num, plot_visible)

Obj_type is a number specifying the type of object to create.

Obj_type	Object
1	Line
2	Rectangle
3	Oval
4	Arc
5	Embedded chart
6	Text box
7	Button
8	Picture (created with the camera tool)
9	Closed polygon
10	Open polygon
11	Check box
12	Option button
13	Edit box
14	Label
15	Dialog frame
16	Spinner
17	Scroll bar
18	List box
19	Group box
20	Drop down list box

Ref1 is a reference to the cell from which the upper-left corner of the object is drawn, or from which the upper-left corner of the object's bounding rectangle is defined.

X_offset1 is the horizontal distance from the upper-left corner of ref1 to the upper-left corner of the object or to the upper-left corner of the object's bounding rectangle. X_offset1 is measured in points. A point is 1/72nd of an inch. If x_offset1 is omitted, it is assumed to be 0.

Y_offset1 is the vertical distance from the upper-left corner of ref1 to the upper-left corner of the object or to the upper-left corner of the object's bounding rectangle. Y_offset1 is measured in points. If y_offset1 is omitted, it is assumed to be 0.

Ref2 is a reference to the cell from which the lower-right corner of the object is drawn, or from which the lower-right corner of the object's bounding rectangle is defined.

X_offset2 is the horizontal distance from the upper-left corner of ref2 to the lower-right corner of the object or to the lower-right corner of the object's bounding rectangle. X_offset2 is measured in points. If x_offset2 is omitted, it is assumed to be 0.

Y_offset2 is the vertical distance from the upper-left corner of ref2 to the lower-right corner of the object or to the lower-right corner of the object's bounding rectangle. Y_offset2 is measured in points. If y_offset2 is omitted, it is assumed to be 0.

Text specifies the text that appears in a text box or button. If text is omitted for a button, the button is named "Button n", where n is a number. If obj_type is not 6 or 7, text is ignored.

Fill is a logical value specifying whether the object is filled or transparent. If fill is TRUE, the object is filled; if FALSE, the object is transparent; if omitted, the object is filled with an applicable pattern for the object being created.

Array is an n by 2 array of values, or a reference to a range of cells containing values, that indicate the position of each vertex in a polygon, relative to the upper-left corner of the polygon's bounding rectangle.

- A vertex is a point that is defined by a pair of coordinates in one row of array.
- If the polygon contains many vertices, one array may not be sufficient to define it. If the number of characters in the formula exceeds 1024, you must include one or more EXTEND.POLYGON functions. If you're recording a macro, Microsoft Excel automatically records EXTEND.POLYGON functions as needed. For more information, see EXTEND.POLYGON.

Xy_series is a number from 0 to 3 that specifies how data is arranged in a chart and corresponds to options in the Paste Special dialog box.

Xy_series	Result
0	Displays a dialog box if the selection is ambiguous
1 or omitted	First row/column is the first data series
2	First row/column contains the category (x) axis labels
3	First row/column contains the x-values; the created chart is an xy (scatter) chart

- Xy_series is ignored unless obj_type is 5 (chart).
- If you want more control over how the data is arranged, use the plot_by, categories, and ser_titles arguments to the CHART.WIZARD function. For more information, see CHART.WIZARD.

Gallery_num is a number from 1 to 15 specifying the type of embedded chart you want to create.

Gallery_num	Chart
1	Area
2	Bar
3	Column
4	Line
5	Pie
6	Radar
7	XY (scatter)
8	Combination
9	3-D area
10	3-D bar
11	3-D column
12	3-D line
13	3-D pie
14	3-D surface
15	Doughnut

Type_num is a number identifying a formatting option for a chart. The formatting options are shown in the dialog box of the AutoFormat command that corresponds to the type of chart you're creating. The first formatting option in any gallery is 1.

Plot_visible is a logical value that corresponds to the Plot Visible Cells Only checkbox in the Chart

tab of the Options dialog box. If FALSE or omitted, all values are plotted.

Editable is a logical value that determines whether the drop down list box is editable or not. If TRUE, the drop down list box is editable. If FALSE, the drop down list box is not editable. If obj_type is not 20, this argument is ignored.

Remarks

- CREATE.OBJECT returns the object identifier of the object it created. Object identifiers include text describing the object, such as "Text" or "Oval", and a number indicating the order in which the object was created. For example, CREATE.OBJECT returns "Oval 3" after creating an oval that is the third object in the workbook.
- If the offsets are not specified, the object is drawn from the upper-left corner of ref1 to the upper-left corner of ref2.
- If the object is not a picture and either ref1 or ref2 is omitted, CREATE.OBJECT returns the #VALUE! error value and does not create the object.
- CREATE.OBJECT also selects the object.
- You must use the COPY function before the CREATE.OBJECT function to create a chart or a picture.

Tip To assign a macro to an object, use the ASSIGN.TO.OBJECT function immediately after creating the object.

Related Functions

<u>ASSIGN.TO.OBJECT</u>	Assigns a macro to an object
<u>EXTEND.POLYGON</u>	Adds vertices to a polygon
<u>FORMAT.MOVE</u>	Moves the selected object
<u>FORMAT.SHAPE</u>	Inserts, moves, or deletes vertices of the selected polygon
<u>FORMAT.SIZE</u>	Sizes an object
<u>GET.OBJECT</u>	Returns information about an object
<u>OBJECT.PROPERTIES</u>	Determines an object's relationship to underlying cells
<u>TEXT.BOX</u>	Replaces text in a text box
List of <u>Command-Equivalent Functions</u>	

CREATE.PUBLISHER

Macro Sheets Only

Equivalent to choosing the Create Publisher command from the Publishing submenu on the Edit menu. Publishes the selected range or chart to an edition file for use by other Macintosh applications.

Important This function is only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

Syntax

CREATE.PUBLISHER(file_text, appearance, size, formats)

CREATE.PUBLISHER?(file_text, appearance, size, formats)

File_text is a text string to be used as the name of the new file that will contain the selected data. If file_text is omitted, Microsoft Excel uses the format "<WorkbookName> Edition #n", where WorkbookName is the name of the document from which the publisher is being created, Edition indicates that the file is an edition file, and n is a unique integer.

For example, if you omit file_text and are publishing a selection from a workbook named Seasonal, and it is your third publisher from that document in the current work session, the default name of the publisher would be "Seasonal Edition #3".

Appearance specifies whether the selection is to be published as shown on screen or as shown when printed. The default value for appearance is 1 if the selection is a sheet and 2 if the selection is a chart.

Appearance	Selection is published
1	As shown on screen
2	As shown when printed

Size specifies the size at which to publish a chart. Size is only available if a chart is to be published.

Size	Chart is published
1 or omitted	As shown on screen
2	As shown when printed

Formats is number specifying what file format or formats CREATE.PUBLISHER should use when it creates the Edition file.

Formats	File format
1	PICT
2	BIFF
4	RTF
8	VALU

- You can also use the sum of the allowable file formats for formats. For example, a value of 6 specifies BIFF and RTF.

- If formats is omitted and the document is a sheet, formats is assumed to be 15 (all formats); if the document is a chart, formats is assumed to be 1 (PICT).

Related Functions

[EDITION.OPTIONS](#)

Sets publisher and subscriber options

[GET.LINK.INFO](#)

Returns information about a link

[SUBSCRIBE.TO](#)

Inserts contents of an edition into the active workbook

[UPDATE.LINK](#)

Updates a link to another workbook

List of [Command-Equivalent Functions](#)

CUSTOM.REPEAT

Macro Sheets Only

Allows custom commands to be repeated using the Repeat tool or the Repeat command on the Edit menu. Also allows custom commands to be recorded using the macro recorder.

Syntax

CUSTOM.REPEAT(macro_text, repeat_text, record_text)

Macro_text is the name of, or a reference to, the macro you want to run when the Repeat command is chosen. If macro_text is omitted, no repeat macro is run, but the custom command can still be recorded.

Repeat_text is the text you want to use as the repeat command on the Edit menu (for example, "Repeat Reports"). You can omit repeat_text and macro_text if you only want to record the formula specified by record_text when using the macro recorder.

Record_text is the formula you want to record. For example, if the user chooses a command named Run Reports in Macro 1, the record_text argument would be "=Macro1!RunReports()", where RunReports is the name of the macro called by the Run Reports command.

- References in record_text must be in R1C1 format.
- If record_text is omitted, the macro recorder records normally (a RUN function with the first cell of the macro as its argument).
- If you are not recording a macro, record_text is ignored.

Tip Place CUSTOM.REPEAT at the end of the macro you will want to repeat. If you place it before the end, then the macro formulas that follow CUSTOM.REPEAT may interfere with the desired effects of CUSTOM.REPEAT. The Repeat tool and the Repeat command continue to change as you choose subsequent commands that can be repeated.

Example

The following macro formula specifies that the macro RepeatReport on the MenuMacros macro sheet in the current workbook will be run when the Repeat Report command is chosen:

```
CUSTOM.REPEAT("MenuMacros!RepeatReport", "Repeat Report")
```

Related Function

CUSTOM.UNDO Specifies a macro to run to undo a custom command

List of Customizing Functions

CUSTOM.UNDO

Macro Sheets Only

Creates a customized Undo tool and Undo or Redo command on the Edit menu for custom commands.

Syntax

CUSTOM.UNDO(macro_text, undo_text)

Macro_text is the name of, or an R1C1-style reference to, the macro you want to run when the Undo command is chosen. Macro_text can be the name or cell reference of a macro.

Undo_text is the text you want to use as the Undo command.

Example

The following macro function runs the UndoMult macro when the user chooses the Undo Times100 command, a custom command that multiplies the current cell by 100.

```
=CUSTOM.UNDO("UndoMult", "&Undo Times100")
```

Tip Use CUSTOM.UNDO directly after the macro functions you want to be able to repeat, because other macro functions following CUSTOM.UNDO might reset the Undo command.

Related Function

CUSTOM.REPEAT Specifies a macro to run to repeat a custom command

List of Customizing Functions

DATA.SERIES

Macro Sheets Only

Equivalent to choosing the Series command from the Fill submenu on the Edit menu. Use DATA.SERIES to enter an interpolated or incrementally increasing or decreasing series of numbers or dates on a sheet or macro sheet.

Syntax

DATA.SERIES(rowcol, type_num, date_num, step_value, stop_value, trend)

DATA.SERIES?(rowcol, type_num, date_num, step_value, stop_value, trend)

Rowcol is a number that specifies where the series should be entered. If rowcol is omitted, the default value is based on the size and shape of the current selection.

Rowcol	Enter series in
1	Rows
2	Columns
Type_num	is a number from 1 to 4 that specifies the type of series.

Type_num	Type of series
1 or omitted	Linear
2	Growth
3	Date
4	AutoFill

Date_num is a number from 1 to 4 that specifies the date unit of the series, as shown in the following table. To use the date_num argument, the type_num argument must be 3.

Date_num	Date unit
1 or omitted	Day
2	Weekday
3	Month
4	Year

Step_value is a number that specifies the step value for the series. If step_value is omitted, it is assumed to be 1.

Stop_value is a number that specifies the stop value for the series. If stop_value is omitted, DATA.SERIES continues filling the series until the end of the selected range.

Trend is a logical value corresponding to the Trend check box. If trend is TRUE, Microsoft Excel generates a linear or exponential trend; if FALSE or omitted, Microsoft Excel generates a standard data series.

Remarks

- If you specify a positive value for stop_value that is lower than the value in the active cell of the selection, DATA.SERIES takes no action.
- If type_num is 4 (AutoFill), Microsoft Excel performs an AutoFill operation just as if you had filled the selection by dragging the fill selection handle or had used the FILL.AUTO macro function.

Related Function

FILL.AUTO Copies cells or automatically fills a selection

List of Command-Equivalent Functions

DEFINE.NAME

Macro Sheets Only

Equivalent to choosing the Define command from the Name submenu on the Insert menu. Defines a name on the active sheet or macro sheet. Use DEFINE.NAME instead of SET.NAME when you want to define a name on the active sheet.

Syntax

DEFINE.NAME(name_text, refers_to, macro_type, shortcut_text, hidden, category, local)

DEFINE.NAME?(name_text, refers_to, macro_type, shortcut_text, hidden, category, local)

Name_text is the text you want to use as the name. Names cannot include spaces, and cannot look like cell references.

Refers_to describes what name_text should refer to, and can be any of the following values.

If refers_to is	Then name_text is
A number, text, or logical value	Defined to refer to that value
An external reference, such as !\$A\$1 or SALES!\$A\$1:\$C\$3	Defined to refer to those cells
A formula in the form of text, such as " $=2*PI()/360$ " (if the formula contains references, they must be R1C1-style references, such as " $=R2C2*(1+RC[-1])$ ")	Defined to refer to that formula
Omitted	Defined to refer to the current selection

The next two arguments, macro_type and shortcut_text, apply only if the sheet in the active window is a macro sheet.

Macro_type is a number from 1 to 3 that indicates the type of macro.

Macro_type	Type of macro
1	Custom function (also known as a function macro)
2	Command macro.
3 or omitted	None (that is, name_text does not refer to a macro)

Shortcut_text is a text value that specifies the macro shortcut key. Shortcut_text must be a single letter, such as "z" or "Z".

Hidden is a logical value specifying whether to define the name as a hidden name. If hidden is TRUE, Microsoft Excel defines the name as a hidden name; if FALSE or omitted, Microsoft Excel defines the name normally.

Category is a number or text identifying the category of a custom function and corresponds to categories in the Function Category list box.

- Categories are numbered starting with 1, the first category in the list.
- If category is text but is not one of the existing function types, Microsoft Excel creates a new category and assigns your custom function to it.

Local is a logical value which, if TRUE, defines the name on just the current sheet or macro sheet. If FALSE or omitted, defines the name for all sheets in the workbook.

Remarks

- You can use hidden names to define values that you want to prevent the user from seeing or changing; they do not appear in the Define Name, Paste Name, or Goto dialog boxes. Hidden names can only be created with the DEFINE.NAME macro function.
- If you are recording a macro and you define a name to refer to a formula, Microsoft Excel converts A1-style references to R1C1-style references. For example, if the active cell is C2, and you define the name Previous to refer to =B2, Microsoft Excel records that command as **DEFINE.NAME**("Previous","=RC[-1]").
- In **DEFINE.NAME?**, the dialog-box form of the function, if refers_to is not specified, the current

selection is proposed in the Refers To box. Also, if a name is not specified, text in the active cell is proposed as the name.

Related Functions

<u>DELETE.NAME</u>	Deletes a name
<u>GET.DEF</u>	Returns a name matching a definition
<u>GET.NAME</u>	Returns the definition of a name
<u>NAMES</u>	Returns the names defined in a workbook
<u>SET.NAME</u>	Defines a name as a value

List of Command-Equivalent Functions

DELETE.ARROW

Macro Sheets Only

Deletes the selected arrow, either drawn as an arrow with the arrow tool or as a line that is later formatted as an arrow. In Microsoft Excel 5.0, arrows are named lines.

Syntax

DELETE.ARROW()

If the selection is not an arrow or a line formatted as an arrow, or if the active window is not a chart, DELETE.ARROW interrupts the macro.

Tip Use the SELECT function (chart syntax), with the number of the arrow (or line) you want to delete in order to select the arrow before using the DELETE.ARROW function. For example, SELECT ("Line 1"). You can also use the CLEAR function to delete the arrow.

Related Functions

[CLEAR](#)

Clears specified information from the selected cells or chart

[DELETE.OVERLAY](#)

Deletes the overlay on a chart

List of [Command-Equivalent Functions](#)

DELETE.CHART.AUTOFORMAT

Deletes a custom format from the list of formats shown in the AutoFormat dialog box for charts.

Syntax

DELETE.CHART.AUTOFORMAT(name_text)

Name_text is the template name you want to delete from the list of custom templates.

Related Functions

[ADD.CHART.AUTOFORMAT](#) Adds a custom template

List of [Command-Equivalent Functions](#)

DELETE.OVERLAY

Macro Sheets Only

Equivalent to choosing the Delete Overlay command from the Chart menu in Microsoft Excel 4.0. Deletes all overlays from a chart. If the chart has no overlay, DELETE.OVERLAY takes no action and returns TRUE.

Syntax

DELETE.OVERLAY()

Related Functions

List of [Command-Equivalent Functions](#)

DEREF

Macro Sheets Only

Returns the value of the cells in a reference.

Syntax

DEREF(reference)

Reference is the cell or cells from which you want to obtain a value. If reference is the reference of a single cell, DEREf returns the value of that cell. If reference is the reference of a range of cells, DEREf returns the array of values in those cells. If reference refers to the active sheet, it must be an absolute reference. Relative references are converted to absolute references.

Remarks

In most formulas, there is no difference between using a value and using the reference of a cell containing that value. The reference is automatically converted to the value, as necessary. For example, if cell A1 contains the value 2, then the formula =A1+1, like the formula =2+1, returns the result 3, because the reference A1 is converted to the value 2. However, in a few functions, such as the SET.NAME function, references are not automatically converted to values. Instead, those functions behave differently depending on whether an argument is a reference or a value.

Example

See the sixth example for SET.NAME.

Related Functions

<u>INDIRECT</u>	Returns a reference indicated by a text value
<u>OFFSET</u>	Returns a reference offset from a given reference
<u>SET.NAME</u>	Defines a names as a value

List of Lookup & Reference Functions

DIALOG.BOX

Macro Sheets Only

Displays the dialog box described in a dialog box definition table.

Syntax

DIALOG.BOX(dialog_ref)

Dialog_ref is a reference to a dialog box definition table on sheet, or an array containing the definition table.

- If an OK button in the dialog box is chosen, DIALOG.BOX enters values in fields as specified in the dialog_ref area and returns the position number of the button chosen. The position numbers start with 1 in the second row of the dialog box definition table.
- If the Cancel button in the dialog box is chosen, DIALOG.BOX returns FALSE.

The dialog box definition table must be at least seven columns wide and two rows high. The definitions of each column in a dialog box definition table are listed in the following table.

Column type	Column number
Item number	1
Horizontal position	2
Vertical position	3
Item width	4
Item height	5
Text	6
Initial value or result	7

The first row of dialog_ref defines the position, size, and name of the dialog box. It can also specify the default selected item and the reference for the Help button. The position is specified in columns 2 and 3, the size in columns 4 and 5, and the name in column 6. To specify a default item, place the item's position number in column 7. You can place the reference for the Help button in row 1, column 1 of the table, but the preferred location is column 7 in the row where the Help button is defined. Row 1, column 1 is usually left blank.

The following table lists the numbers for the items you can display in a dialog box.

Dialog-box item	Item number
Default OK button	1
Cancel button	2
OK button	3
Default Cancel button	4
Static text	5
Text edit box	6
Integer edit box	7
Number edit box	8
Formula edit box	9
Reference edit box	10
Option button group	11
Option button	12
Check box	13
Group box	14
List box	15
Linked list box	16
Icons	17
Linked file list box (Microsoft Excel for Windows only)	18
Linked drive and directory box (Microsoft Excel for Windows only)	19

Directory text box	20
Drop-down list box	21
Drop-down combination edit/list box	22
Picture button	23
Help button	24

Remarks

- Add 100 to an item number in the above table to define the item as a trigger. A trigger is a dialog box item that, when chosen, returns to your macro (as choosing OK would) but continues to display the dialog box, allowing your macro to change the dialog box definition or display an alert message or another dialog box. The Help button, edit boxes, group boxes, static text, and icons cannot be triggers.
- Add 200 to an item number to define it as dimmed. A dimmed (gray) item cannot be chosen or selected. For example, 203 is a dimmed OK button. You can use item 223 to include a picture in your dialog box that does not behave like a button.
- If a trigger has been chosen and you still want to clear a dynamic dialog box from the screen, use `DIALOG.BOX(FALSE)`. This is useful if you want to confirm that the dialog box has been filled out correctly before dismissing it.
- The dialog box definition table can be an array. If `dialog_ref` is an array instead of a reference, `DIALOG.BOX` returns a modified copy of that array, along with the results of the dialog box in the seventh column. (The first item in the seventh column is the position number of the chosen button or of a triggered item.) This is useful if you want to preserve the original dialog box definition table since `DIALOG.BOX` does not modify the original array argument. If you cancel the dialog box, or if a dialog box error occurs, `DIALOG.BOX` returns `FALSE` instead of an array.

Related Functions

- [ALERT](#) Displays a dialog box and a message
- [INPUT](#) Displays a dialog box for user input
- List of [Customizing Functions](#)

DIRECTORY

Macro Sheets Only

Sets the current drive and directory or folder to the specified path and returns the name of the new directory or folder as text. Use DIRECTORY to get the name of the current directory or folder for use with the OPEN and SAVE.AS functions or to specify a directory or folder from which to return a list of files with the FILES function.

Syntax

DIRECTORY(path_text)

Path_text is the drive and directory or folder you want to change to.

- If path_text is not specified, DIRECTORY returns the name of the current directory or folder as text.
- If path_text does not include a drive specifier, the current drive is assumed.

Examples

In Microsoft Excel for Windows, the following macro formula sets the directory to \EXCEL\MODELS on the current drive and returns the value "drive:\EXCEL\MODELS":

```
DIRECTORY (" \EXCEL\MODELS")
```

The following macro formula sets the current drive to E and sets the directory to \EXCEL\MODELS on E. It returns the value "E:\EXCEL\MODELS":

```
DIRECTORY ("E: \EXCEL\MODELS")
```

In Microsoft Excel for the Macintosh, the following macro formula sets the folder to HARD DISK: APPS:EXCEL:FINANCIALS and returns the value "HARD DISK:APPS:EXCEL:FINANCIALS":

```
DIRECTORY ("HARD DISK:APPS:EXCEL:FINANCIALS")
```

Related Functions

[FILES](#)

Returns the filenames in the specified directory or folder

List of [Information Functions](#)

DISABLE.INPUT

Macro Sheets Only

Blocks all input from the keyboard and mouse to Microsoft Excel (except input to displayed dialog boxes). Use DISABLE.INPUT to prevent input from the user or from other applications.

Syntax

DISABLE.INPUT(logical)

Logical is a logical value specifying whether input is currently disabled. If logical is TRUE, input is disabled; if FALSE, input is reenabled.

Remarks

Disabling input can be useful if you are using dynamic data exchange (DDE) to communicate with Microsoft Excel from another application.

Important Be sure to end any macro that uses DISABLE.INPUT(TRUE) with a DISABLE.INPUT(FALSE) function. If you do not include DISABLE.INPUT(FALSE) to allow non-dialog-box input, you will not be able to take any actions on your computer after the macro has finished.

Related Functions

<u>CANCEL.KEY</u>	Disables macro interruption
<u>ENTER.DATA</u>	Turns Data Entry mode on and off
<u>WORKSPACE</u>	Changes workspace settings
List of <u>Customizing Functions</u>	

EDIT.COLOR

Macro Sheets Only

Equivalent to choosing the Modify button from the Color tab, which appears when you choose the Options command from the Tools menu. Defines the color for one of the 56 color palette boxes.

Use EDIT.COLOR if you want to use a color that is not currently on the palette and if your system hardware has more than 56 colors available. After you set the color for the color box, any items previously formatted with that color are displayed in the new color.

Syntax

EDIT.COLOR(color_num, red_value, green_value, blue_value)

EDIT.COLOR?(color_num, red_value, green_value, blue_value)

Color_num is a number from 1 to 56 specifying one of the 56 color palette boxes for which you want to set the color.

Red_value, green_value, and blue_value are numbers that specify how much red, green, and blue are in each color.

- In Microsoft Excel for Windows, red_value, green_value, and blue_value are numbers from 0 to 255.
- In Microsoft Excel for the Macintosh, red_value, green_value, and blue_value are also numbers from 0 to 255. However, the color editing dialog box displays numbers from 0 to 65, 535. Microsoft Excel automatically converts the numbers between the two ranges. This allows you to display similar colors in all operating environments without modifying your macros.
- If red_value, green_value, and blue_value are all set to 255, the resulting color is white. If they are all set to zero, the resulting color is black.
- If red_value, green_value, or blue_value is omitted, Microsoft Excel assumes it to be the appropriate value for that color_num.

Remarks

- Your system hardware determines the number of unique colors that you can choose from and the number of colors that can be displayed on the screen at the same time.
- EDIT.COLOR does not use hue, saturation, or brightness values. If you are using the macro recorder and set the color of a color palette box using hue, saturation, and luminance, Microsoft Excel records the corresponding red, green, and blue values instead.
- The dialog-box form of this function, EDIT.COLOR?(color_num), displays your system's color editing dialog box. The default red_value, green_value, and blue_value are determined by the current settings for the color_num you specify. Color_num is a required argument for the dialog-box form of this function.

Related Function

[COLOR.PALETTE](#) Copies a color palette from one workbook to another

List of [Command-Equivalent Functions](#)

EDIT.REPEAT

Macro Sheets Only

Equivalent to choosing the Repeat command from the Edit menu. Repeats certain actions and commands. EDIT.REPEAT is available in the same situations as the Repeat command.

Syntax

EDIT.REPEAT()

Related Functions

List of [Command-Equivalent Functions](#)

EDIT.SERIES

Macro Sheets Only

Equivalent to choosing the Edit Series command from the Chart menu in Microsoft Excel version 4.0. Creates or changes chart series by adding a new SERIES formula or modifying an existing SERIES formula in the topmost chart type. Chart types are displayed in the following order from top to bottom: XY (Scatter), Line, Column, Bar, Area.

Syntax

EDIT.SERIES(series_num, name_ref, x_ref, y_ref, z_ref, plot_order)

EDIT.SERIES?(series_num, name_ref, x_ref, y_ref, z_ref, plot_order)

Series_num is the number of the series you want to change. If series_num is 0 or omitted, Microsoft Excel creates a new data series.

Name_ref is the name of the data series. It can be an external reference to a single cell, a name defined as a single cell, or a name defined as a sequence of characters. Name_ref can also be text (for example, "Projected Sales").

X_ref is an external reference to the name of the sheet and the cells that contain one of the following sets of data:

- Category labels for all charts except xy (scatter) charts
- X-coordinate data for xy (scatter) charts

Y_ref is an external reference to the name of the sheet and the cells that contain values (or y-coordinate data in xy (scatter) charts) for all 2-D charts. Y_ref is required in 2-D charts but does not apply to 3-D charts.

Z_ref is an external reference to the name of the sheet and the cells that contain values for all 3-D charts. Z_ref is required in 3-D charts but does not apply to 2-D charts.

Plot_order is a number specifying whether the data series is plotted first, second, and so on, in the chart type.

- If you assign a plot_order to a series, Microsoft Excel plots that series in the order you specify, and the series that previously had that plot order (and any series following it) has its plot order increased by one.
- If you add a series to a chart with an overlay, the number of series in the main chart does not change, so if the series is added to the main chart, then the series that was plotted last in the main chart will be plotted first in the overlay chart. To change which series is plotted first in the overlay chart, use the (chart type) Group command from the Format menu, and then select the Series Order tab in the Format (chart type) Group dialog box. You can also use the FORMAT.OVERLAY function.
- If you omit plot_order when you add a new series, then Microsoft Excel plots that series last and assigns it the correct plot_order value.
- The maximum value for plot_order is 255.

Remarks

To change where a series is plotted within a chart, you can change the chart type, using the FORMAT.CHART function, or the plot order. Plot order affects where the series appears within the chart type only.

X_ref, y_ref, and z_ref can be arrays or references to a nonadjacent selection, although they cannot be names that refer to a nonadjacent selection. If you specify a nonadjacent selection for any of these arguments, make sure to enclose the reference to the selection in parentheses so that Microsoft Excel does not treat the components of the references as separate arguments.

Tip To delete a data series, use the SELECT("Sn") macro function, where n is the series number, followed by the FORMULA("") macro function. You can also use the CLEAR function instead of FORMULA.

Related Functions

[FORMAT.CHART](#)

List of [Command-Equivalent Functions](#)

ENTER.DATA

Macro Sheets Only

Turns on Data Entry mode and allows you to select and to enter data into the unlocked cells in the current selection only (the data entry area). Use ENTER.DATA when you want to enter data only in a specific part of your sheet. You can then use that part of the sheet as a simple data form.

Syntax

ENTER.DATA(logical)

Logical is a logical value that turns Data Entry mode on or off.

- If logical is TRUE, Data Entry mode is turned on; if FALSE, Data Entry mode is turned off and data entry, cell movement, and cell selection return to normal. If logical is omitted, ENTER.DATA toggles Data Entry mode.
- Logical can also be the number 2. This setting turns on Data Entry mode and prevents the ESC key from turning it off.
- Logical can also be a reference. Using a reference for this argument turns on Data Entry mode for the supplied reference.

Remarks

- In Data Entry mode, you can move the active cell and select cell ranges only in the data entry area. The arrow keys and the TAB and SHIFT+TAB keys move from one unlocked cell to the next, wrapping to the first or last unlocked cell in the next or previous column when the end of a column is reached. The HOME and END keys move to the first and last cell in the data entry area, respectively. You cannot select entire rows or columns, and clicking a cell outside the data entry area does not select it.
 - The only commands available in Data Entry mode are commands normally available to protected workbooks.
 - To turn off Data Entry mode, press ESC (unless logical is 2), activate another window, or use another ENTER.DATA function. If you use another ENTER.DATA function, you will usually design your macros in one of two ways:
 - The macro turns on Data Entry mode, pauses while you enter data, resumes, and then turns off Data Entry mode.
 - The macro turns on Data Entry mode and ends. After entering data, another macro turns off Data Entry mode; this latter macro could be assigned to a "Finished" button, for example.
- With either method, you can use Microsoft Excel's ON functions to resume or run other macros based on an event, such as pressing the CONTROL+D keys.

Tips

- Normally you use Data Entry mode to enter data, but you can also prevent someone from entering data or moving the active cell by locking all the cells in the current selection before turning on Data Entry mode. This is useful if you want a user to view a range of cells but not change it or move the active cell. Similarly, if you unlock certain cells, you can restrict the user's movement to the Data Entry area only.
 - To prevent someone from activating another workbook, which would turn off Data Entry mode, use the ON.WINDOW function or an Auto_Deactivate macro.
-

Related Function

DISABLE.INPUT Blocks all input to Microsoft Excel
FORMULA Enters values into a cell or range or onto a chart
List of Command-Equivalent Functions

EVALUATE

Macro Sheets Only

Evaluates a formula or expression that is in the form of text and returns the result. To run a macro or subroutine, use the RUN function.

Syntax

EVALUATE(formula_text)

Formula_text is the expression in the form of text that you want to evaluate.

Remarks

Using EVALUATE is similar to selecting an expression within a formula in the formula bar and pressing the Recalculate key (F9 in Microsoft Excel for Windows and COMMAND+= in Microsoft Excel for the Macintosh). EVALUATE replaces an expression with a value.

Example

Suppose you want to know the value of a cell named LabResult1, LabResult2, or LabResult3, where the 1, 2, or 3 is specified by the name TrialNum whose value may change as the macro runs. You can use the following formula to calculate the value:

```
EVALUATE("LabResult"&TrialNum)
```

Related Functions

RUN Runs a macro

List of Command-Equivalent Functions

FILES

Macro Sheets Only

Returns a horizontal text array of the names of all files in the specified directory or folder. Use FILES to build a list of filenames upon which you want your macro to operate.

Syntax

FILES(directory_text)

Directory_text specifies which directories or folders to return filenames from.

- Directory_text accepts an asterisk (*) to represent a series of characters and a question mark (?) to represent a single character in filenames.
- If directory_text is not specified, FILES returns filenames from the current directory.

Remarks

If you enter FILES in a single cell, only one filename is returned. You will normally use FILES with SET.NAME to assign the returned array to a name. See the last example below.

Tips You can use COLUMNS to count the number of entries in the returned array. You can use TRANSPOSE to change a horizontal array to a vertical one.

Examples

In Microsoft Excel for Windows, the following macro formula returns the names of all files starting with the letter F in the current directory or folder:

```
FILES("F*. *")
```

When entered as an array formula in several cells, the following macro formula returns the filenames in the current directory to those cells. If the directory contains fewer files than can fit in the selected cells, the #N/A error value appears in the extra cells.

```
FILES()
```

In Microsoft Excel for Windows, the following macro formula returns all files starting with "SALE" and ending with the .XLC extension in the \EXCEL\CHARTS subdirectory:

```
FILES("C:\EXCEL\CHARTS\SALE*.XLC")
```

In Microsoft Excel for the Macintosh, the following macro formula returns all files starting with "SALE" in the nested CHART folder:

```
FILES("DISK:EXCEL:CHART:SALE*")
```

The following macro stores the names of the files in the current directory in the named array FileArray

```
SET.NAME("FileArray",FILES())
```

Related Functions

<u>COLUMNS</u>	Returns the number of columns of a reference
<u>DOCUMENTS</u>	Returns the names of the specified open workbooks
<u>FILE.DELETE</u>	Deletes a file
<u>OPEN</u>	Opens a workbook
<u>SET.NAME</u>	Defines a name as a value
<u>TRANSPOSE</u>	Returns the transpose of an array

List of Information Functions

FILL.DOWN

FILL.LEFT

FILL.RIGHT

FILL.UP

Macro Sheets Only

Equivalent to choosing the Down, Left, Right, and Up commands, respectively, from the Fill submenu on the Edit menu.

Syntax

FILL.DOWN()

FILL.LEFT()

FILL.RIGHT()

FILL.UP()

FILL.DOWN copies the contents and formats of the cells in the top row of a selection into the rest of the rows in the selection.

FILL.LEFT copies the contents and formats of the cells in the right column of a selection into the rest of the columns in the selection.

FILL.RIGHT copies the contents and formats of the cells in the left column of a selection into the rest of the columns in the selection.

FILL.UP copies the contents and formats of the cells in the bottom row of a selection into the rest of the rows in the selection.

Remarks

If you have a multiple selection, each range in the selection is filled separately with the contents of the source range.

Related Functions

<u>COPY</u>	Copies and pastes data or objects
<u>DATA.SERIES</u>	Fills a range of cells with a series of numbers or dates
<u>FILL.AUTO</u>	Copies cells or automatically fills a selection
<u>FORMULA.FILL</u>	Enters a formula in the specified range
List of <u>Command-Equivalent Functions</u>	

FORMAT.LEGEND

Macro Sheets Only

Equivalent to choosing the Selected Legend command from the Format menu when a chart is active. Determines the position and orientation of the legend on a chart and returns TRUE; returns an error message if the legend is not already selected.

Syntax

FORMAT.LEGEND(position_num)

FORMAT.LEGEND?(position_num)

Position_num is a number from 1 to 5 specifying the position of the legend.

Position_num	Position of legend
1	Bottom
2	Corner
3	Top
4	Right
5	Left

Related Functions

FORMAT.MOVE Moves the selected object

FORMAT.SIZE Sizes an object

LEGEND Adds or deletes a chart legend

List of Command-Equivalent Functions

FORMAT.MAIN

Macro Sheets Only

Equivalent to choosing the Main Chart command from the Format menu in Microsoft Excel version 4.0. Formats a chart according to the arguments you specify. This function is included for compatibility with Microsoft Excel 5.0. In Microsoft Excel 5.0, this is equivalent to choosing the Chart Type command from the Format menu. You can also use the FORMAT.CHART function.

Syntax

FORMAT.MAIN(type_num, view, overlap, gap_width, vary, drop, hilo, angle, gap_depth, chart_depth, up_down, series_line, labels, doughnut_size)

FORMAT.MAIN?(type_num, view, overlap, gap_width, vary, drop, hilo, angle, gap_depth, chart_depth, up_down, series_line, labels, doughnut_size)

Type_num is a number specifying the type of chart.

Type_num	Chart
1	Area
2	Bar
3	Column
4	Line
5	Pie
6	XY (Scatter)
7	3-D Area
8	3-D Column
9	3-D Line
10	3-D Pie
11	Radar
12	3-D Bar
13	3-D Surface
14	Doughnut

View is a number specifying one of the views in the Data View box in the Main Chart dialog box. The view varies depending on the type of chart.

Overlap is a number from -100 to 100 specifying how you want bars or columns to be positioned. It corresponds to the Overlap box in the Main Chart dialog box. Overlap is ignored if type_num is not 2 or 3 (bar or column chart).

- If overlap is positive, it specifies the percentage of overlap you want for bars or columns. For example, 50 would cause one-half of a bar or column to be covered by an adjacent bar or column. A value of zero prevents bars or columns from overlapping.
- If overlap is negative, then bars or columns are separated by the specified percentage of the maximum available distance between any two bars or columns.
- If overlap is omitted, it is assumed to be 0 (bars or columns do not overlap), or it is unchanged if a value was previously set.

Gap_width is a number from 0 to 500 specifying the space between bar or column clusters as a percentage of the width of a bar or column. It corresponds to the Gap Width box in the Main Chart dialog box.

- Gap_width is ignored if type_num is not 2, 3, 8, or 12 (bar or column chart).
- If gap_width is omitted, it is assumed to be 50, or it is unchanged if a value was previously set.

Several of the following arguments are logical values corresponding to check boxes in the Main Chart dialog box. If an argument is TRUE, Microsoft Excel selects the corresponding check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the setting is unchanged.

Vary corresponds to the Vary By Categories check box. Vary applies only to charts with one data series and is not available for area charts.

Drop corresponds to the Drop Lines check box. Drop is available only for area and line charts.

Hilo corresponds to the Hi-Lo Lines check box. Hilo is available only for line charts.

Angle is a number from 0 to 360 specifying the angle of the first pie slice (in degrees) if the chart is a pie chart. If angle is omitted, it is assumed to be 0, or it is unchanged if a value was previously set.

The next two arguments are for 3-D charts only.

Gap_depth is a number from 0 to 500 specifying the depth of the gap in front of and behind a bar, column, area, or line as a percentage of the depth of the bar, column, area, or line.

- Gap_depth is ignored if the chart is a pie chart or if it is not a 3-D chart.
- If gap_depth is omitted and the chart is a 3-D chart, gap_depth is assumed to be 50, or it is unchanged if a value was previously set. If gap_depth is omitted and the view is side-by-side, stacked, or stacked 100%, gap_depth is assumed to be 0, or it is unchanged if a value was previously set.

Chart_depth is a number from 20 to 2000 specifying the visual depth of the chart as a percentage of the width of the chart. Chart_depth corresponds to the Chart Depth box in the Main Chart dialog box.

- Chart_depth is ignored if the chart is not a 3-D chart.
- If chart_depth is omitted, it is assumed to be 100, or it is unchanged if a value was previously set.

The next three arguments are logical values corresponding to check boxes in the Main Chart dialog box. If an argument is TRUE, Microsoft Excel selects the corresponding check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the setting is unchanged. The final argument is for compatibility with Microsoft Excel 5.0.

Up_down corresponds to the Up/Down Bars check box. Up_down is available only for line charts.

Series_line corresponds to the Series Lines check box. Series_line is available only for stacked bar and column charts.

Labels corresponds to the Radar Axis Labels check box. Labels is available only for radar charts.

Doughnut_size specifies the size of the hole in a doughnut chart. Can be a value from 10% - 90%. Default is 50%

Related Function

FORMAT.CHART

Formats a chart

FORMAT.OVERLAY

Formats an overlay chart

List of Command-Equivalent Functions

FORMAT.OVERLAY

Macro Sheets Only

Equivalent to choosing the Overlay command from the Format menu in Microsoft Excel version 4.0. This function is included for compatibility with Microsoft Excel 5.0. Formats the overlay chart according to the arguments you specify.

Syntax

FORMAT.OVERLAY(type_num, view, overlap, gap_width, vary, drop, hilo, angle, series_dist, series_num, up_down, series_line, labels)

FORMAT.OVERLAY?(type_num, view, overlap, gap_width, vary, drop, hilo, angle, series_dist, series_num, up_down, series_line, labels)

Type_num is a number specifying the type of chart.

Type_num	Chart
1	Area
2	Bar
3	Column
4	Line
5	Pie
6	XY (Scatter)
11	Radar
14	Doughnut

View is a number specifying one of the views in the Data View box in the Overlay dialog box. The view varies depending on the type of chart.

Overlap is a number from -100 to 100 specifying how you want bars or columns to be positioned. It corresponds to the Overlap box in the Overlay dialog box. Overlap is ignored if type_num is not 2 or 3 (bar or column chart).

- If overlap is positive, it specifies the percentage of overlap you want for bars or columns. For example, 50 would cause one-half of a bar or column to be covered by an adjacent bar or column.
- If overlap is negative, then bars or columns are separated by the specified percentage of the maximum available distance between any two bars or columns.
- If overlap is omitted, it is assumed to be 0 (bars or columns do not overlap), or it is unchanged if a value was previously set.

Gap_width is a number from 0 to 500 specifying the space between bar or column clusters as a percentage of the width of a bar or column.

- Gap_width is ignored if type_num is not 2 or 3 (bar or column chart).
- If gap_width is omitted, it is assumed to be 50, or it is unchanged if a value was previously set.

Several of the following arguments are logical values corresponding to check boxes in the Overlay dialog box. If an argument is TRUE, Microsoft Excel selects the corresponding check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the setting is unchanged.

Vary corresponds to the Vary By Categories check box. Vary is not available for area charts.

Drop corresponds to the Drop Lines check box. Drop is available only for area and line charts.

Hilo corresponds to the Hi-Lo Lines check box. Hilo is available only for line charts.

Angle is a number from 0 to 360 specifying the angle of the first pie slice (in degrees) if the chart is a pie chart. If angle is omitted, it is assumed to be 0, or it is unchanged if a value was previously set.

Series_dist is the number 1 or 2 and specifies automatic or manual series distribution.

- If series_dist is 1 or omitted, Microsoft Excel uses automatic series distribution.
- If series_dist is 2, Microsoft Excel uses manual series distribution, and you must specify which series is first in the distribution by using the series_num argument.

Series_num is the number of the first series in the overlay chart and corresponds to the First Overlay Series box in the Overlay dialog box. If series_dist is 1 (automatic series distribution), this argument is ignored.

Up_down corresponds to the Up/Down Bars check box. Up_down is available only for line charts.

Series_line corresponds to the Series Lines check box. Series_line is available only for stacked bar and column charts.

Labels corresponds to the Radar Axis Labels check box. Labels is available only for radar charts.

Related Functions

DELETE.OVERLAY

Deletes the overlay on a chart

FORMAT.CHART

Formats a chart

List of Command-Equivalent Functions

FORMULA.CONVERT

Macro Sheets Only

Changes the style and type of references in a formula between A1 and R1C1 and between relative and absolute. Use FORMULA.CONVERT to convert references of one style or type to another style or type.

Syntax

FORMULA.CONVERT(formula_text, from_a1, to_a1, to_ref_type, rel_to_ref)

Formula_text is the formula, given as text, containing the references you want to change.

Formula_text must be a valid formula, and an equal sign must be included.

From_a1 is a logical value specifying whether the references in formula_text are in A1 or R1C1 style. If from_a1 is TRUE, references are in A1 style; if FALSE, references are in R1C1 style.

To_a1 is a logical value specifying the form for the references FORMULA.CONVERT returns. If to_a1 is TRUE, references are returned in A1 style; if FALSE, references are returned in R1C1 style. If to_a1 is omitted, the reference style is not changed.

To_ref_type is a number from 1 to 4 specifying the reference type of the returned formula. If to_ref_type is omitted, the reference type is not changed.

To_ref_type	Reference type returned
1	Absolute
2	Absolute row, relative column
3	Relative row, absolute column
4	Relative

Rel_to_ref is an absolute reference that specifies what cell the relative references are or should be relative to.

Examples

Use FORMULA.CONVERT to convert relative references entered by the user in an INPUT function or custom dialog box into absolute references. The following macro formula converts the given formula to an absolute, R1C1-style reference:

`FORMULA.CONVERT("=A1:A10", TRUE, FALSE, 1)` equals `"=R1C1:R10C1"`

The following macro formula converts the references in the given formula to relative, A1-style references:

`FORMULA.CONVERT("=SUM(R10C2:R15C2)", FALSE, TRUE, 4)` equals `"=SUM(B10:B15)"`

Tip To put the converted formula into a cell or range of cells, use the FORMULA.CONVERT function as the formula_text argument to the FORMULA function.

Related Functions

[ABSREF](#) Returns the absolute reference of a range of cells to another range

[FORMULA](#) Enters values into a cell or range or onto a chart

[RELREF](#) Returns a relative reference

List of [Command-Equivalent Functions](#)

FREEZE.PANES

Macro Sheets Only

Equivalent to choosing the Freeze Panes or Unfreeze Panes commands from the Window menu or clicking the Freeze Panes or Unfreeze Panes button. Splits the active window into panes, creates frozen panes, or freezes or unfreezes existing panes. Use FREEZE.PANES to keep row or column titles on the screen while scrolling to other parts of the sheet.

Syntax

FREEZE.PANES(logical, col_split, row_split)

Logical is a logical value specifying which command FREEZE.PANES is equivalent to.

- If logical is TRUE, the function is equivalent to the Freeze Panes command. It freezes panes if they exist, or creates them, splits them at the specified position, and freezes them if they do not exist. If the panes are already frozen, FREEZE.PANES takes no action.

- If logical is FALSE, the function is equivalent to the Unfreeze Panes command. If no panes exist, FREEZE.PANES takes no action.

- If logical is omitted, FREEZE.PANES creates and then freezes panes if no panes exist, freezes existing panes if they're not currently frozen, or unfreezes existing panes if they're currently frozen.

Col_split specifies where to split the window vertically and is measured in columns from the left of the window.

Row_split specifies where to split the window horizontally and is measured in rows from the top of the window.

Col_split and row_split are ignored unless logical is TRUE and split panes do not exist.

Remarks

To create panes without freezing or unfreezing them, use the SPLIT function. You can freeze the panes later using the FREEZE.PANES function.

Related Functions

ACTIVATE Switches to a window

SPLIT Splits a window

List of Command-Equivalent Functions

GALLERY.3D.AREA

Macro Sheets Only

Equivalent to choosing a 3-D Area template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a 3-D area chart.

Syntax

GALLERY.3D.AREA(type_num)

GALLERY.3D.AREA?(type_num)

Type_num is the number of the 3-D Area format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.3D.BAR

Macro Sheets Only

Equivalent to choosing a 3-D Bar template from the AutoFormat dialog box when a chart is active.
Changes the active chart to a 3-D bar chart.

Syntax

GALLERY.3D.BAR(type_num)

GALLERY.3D.BAR?(type_num)

Type_num is the number of the 3-D Bar format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.3D.COLUMN

Macro Sheets Only

Equivalent to choosing a 3-D Column template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a 3-D column chart.

Syntax

GALLERY.3D.COLUMN(type_num)

GALLERY.3D.COLUMN?(type_num)

Type_num is the number of the 3-D Column format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.3D.LINE

Macro Sheets Only

Equivalent to choosing a 3-D Line template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a 3-D line chart.

Syntax

GALLERY.3D.LINE(type_num)

GALLERY.3D.LINE?(type_num)

Type_num is the number of the 3-D Line format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.3D.PIE

Macro Sheets Only

Equivalent to choosing a 3-D Pie template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a 3-D pie chart.

Syntax

GALLERY.3D.PIE(type_num)

GALLERY.3D.PIE?(type_num)

Type_num is the number of the 3-D Pie format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.3D.SURFACE

Macro Sheets Only

Equivalent to choosing a 3-D Surface template from the AutoFormat dialog box when a chart is active.
Changes the active chart to a 3-D surface chart.

Syntax

GALLERY.3D.SURFACE(type_num)

GALLERY.3D.SURFACE?(type_num)

Type_num is the number of the 3-D Surface format that you want to apply to the chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.AREA

Macro Sheets Only

Equivalent to choosing an Area template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to an area chart.

Syntax

GALLERY.AREA(type_num, delete_overlay)

GALLERY.AREA?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the area chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.BAR

Macro Sheets Only

Equivalent to choosing a Bar template from the AutoFormat dialog box when a chart is active. Changes the format of the active chart to a bar chart.

Syntax

GALLERY.BAR(type_num, delete_overlay)

GALLERY.BAR?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the bar chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.COLUMN

Macro Sheets Only

Equivalent to choosing a Column template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a column chart.

Syntax

GALLERY.COLUMN(type_num, delete_overlay)

GALLERY.COLUMN?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the column chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.CUSTOM

Macro Sheets Only

Equivalent to choosing a custom format from the AutoFormat dialog box when a chart is active. Changes the format of the active chart to the custom format.

Syntax

GALLERY.CUSTOM(name_text)

Name_text is the name of the custom template you want to apply.

Related Functions

[ADD.CHART.AUTOFORMAT](#)

Formats a chart using a custom gallery

[DELETE.CHART.AUTOFORMAT](#)

Deletes a custom gallery

List of [Command-Equivalent Functions](#)

GALLERY.DOUGHNUT

Macro Sheets Only

Equivalent to choosing a Doughnut template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a doughnut chart.

GALLERY.DOUGHNUT(type_num)

GALLERY.DOUGHNUT?(type_num)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the doughnut chart.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.LINE

Macro Sheets Only

Equivalent to choosing a Line template from the AutoFormat dialog box when a chart is active. Changes the format of the active chart to a line chart.

Syntax

GALLERY.LINE(type_num, delete_overlay)

GALLERY.LINE?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the line chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.PIE

Macro Sheets Only

Equivalent to choosing a Pie template from the AutoFormat dialog box when a chart is active. Changes the format of the active chart to a pie chart.

Syntax

GALLERY.PIE(type_num, delete_overlay)

GALLERY.PIE?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the pie chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.RADAR

Macro Sheets Only

Equivalent to choosing a Radar template from the AutoFormat dialog box when a chart is active.
Changes the format of the active chart to a radar chart.

Syntax

GALLERY.RADAR(type_num, delete_overlay)

GALLERY.RADAR?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the radar chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GALLERY.SCATTER

Macro Sheets Only

Equivalent to choosing an XY (Scatter) template from the AutoFormat dialog box when a chart is active. Changes the format of the active chart to an xy (scatter) chart.

Syntax

GALLERY.SCATTER(type_num, delete_overlay)

GALLERY.SCATTER?(type_num, delete_overlay)

Type_num is the number of a format in the AutoFormat dialog box when a chart is active dialog box that you want to apply to the xy (scatter) chart.

Delete_overlay is a logical value specifying whether to delete an overlay chart.

- If delete_overlay is TRUE, Microsoft Excel deletes all overlays, if present, and applies the new format to the main chart.
- If delete_overlay is FALSE or omitted, Microsoft Excel applies the new format to either the main chart or the overlay, depending on the location of the selected series.

Related Functions

List of [Command-Equivalent Functions](#)

GET.CHART.ITEM

Macro Sheets Only

Returns the vertical or horizontal position of a point on a chart item. Use these position numbers with FORMAT.MOVE and FORMAT.SIZE to change the position and size of chart items. Position is measured in points; a point is 1/72nd of an inch.

Syntax

GET.CHART.ITEM(x_y_index, point_index, item_text)

X_y_index is a number specifying which of the coordinates you want returned.

X_y_index	Coordinate returned
1	Horizontal coordinate
2	Vertical coordinate

Point_index is a number specifying the point on the chart item. These indexes are described later. If point_index is omitted, it is assumed to be 1.

If the specified item is a point, point_index must be 1.

If the specified item is any line other than a data line, use the following values for point_index.

Point_index	Chart item position
1	Lower or left
2	Upper or right

If the selected item is a legend, plot area, chart area, or an area in an area chart, use the following values for point_index.

Point_index	Chart item position
1	Upper left
2	Upper middle
3	Upper right
4	Right middle
5	Lower right
6	Lower middle
7	Lower left
8	Left middle

If the selected item is an arrow in Microsoft Excel 4.0, use the following values for point_index. In Microsoft Excel 5.0, arrows are named lines, and the arrowhead position returned is equivalent to the end of a line where the arrowhead begins.

Point_index	Chart item position
1	Arrow shaft
2	Arrowhead

If the selected item is a pie slice, use the following values for point_index.

Point_index	Chart item position
1	Outermost counterclockwise point
2	Outer center point
3	Outermost clockwise point
4	Midpoint of the most clockwise radius
5	Center point
6	Midpoint of the most counterclockwise radius

Item_text is a selection code that specifies which item of a chart to select. See the chart form of SELECT for the item_text codes to use for each item of a chart.

- If item_text is omitted, it is assumed to be the currently selected item.
- If item_text is omitted and no item is selected, GET.CHART.ITEM returns the #VALUE! error

value.

Remarks

If the specified item does not exist, or if a chart is not active when the function is carried out, the #VALUE! error value is returned.

Examples

The following macro formulas return the horizontal and vertical locations, respectively, of the top of the main-chart value axis:

```
GET.CHART.ITEM(1, 2, "Axis 1")
```

```
GET.CHART.ITEM(2, 2, "Axis 1")
```

You could then use FORMAT.MOVE to move a floating text item to the position returned by these two formulas.

Related Functions

[GET.DOCUMENT](#) Returns information about a workbook

[GET.FORMULA](#) Returns the contents of a cell

List of [Command-Equivalent Functions](#)

GET.DEF

Macro Sheets Only

Returns the name, as text, that is defined for a particular area, value, or formula in a workbook. Use GET.DEF to get the name corresponding to a definition. To get the definition of a name, use GET.NAME.

Syntax

GET.DEF(def_text, document_text, type_num)

Def_text can be anything you can define a name to refer to, including a reference, a value, an object, or a formula.

- References must be given in R1C1 style, such as "R3C5".
- If def_text is a value or formula, it is not necessary to include the equal sign that is displayed in the Refers To box in the Define Name dialog box, which appears when you choose the Name command from the Define submenu on the Insert Menu.
- If there is more than one name for def_text, GET.DEF returns the first name. If no name matches def_text, GET.DEF returns the #NAME? error value.

Document_text specifies the sheet or macro sheet that def_text is on. If document_text is omitted, it is assumed to be the active macro sheet.

Type_num is a number from 1 to 3 specifying which types of names are returned.

Type_num	Returns
1 or omitted	Normal names only
2	Hidden names only
3	All names

Examples

If the specified range in Sheet4 is named Sales, the following macro formula returns "Sales":

```
GET.DEF("R2C2:R9C6", "Sheet4")
```

If the value 100 in Sheet4 is defined as Constant, the following macro formula returns "Constant":

```
GET.DEF("100", "Sheet4")
```

If the specified formula in Sheet4 is named SumTotal, the following macro formula returns "SumTotal":

```
GET.DEF("SUM(R1C1:R10C1)", "Sheet4")
```

If 3 is defined as the hidden name Counter on the active macro sheet, the following macro formula returns "Counter":

```
GET.DEF("3", , 2)
```

Related Functions

<u>GET.CELL</u>	Returns information about the specified cell
<u>GET.NAME</u>	Returns the definition of a name
<u>GET.NOTE</u>	Returns characters from a note
<u>NAMES</u>	Returns the names defined on a workbook

List of [Information Functions](#)

GET.FORMULA

Macro Sheets Only

Returns the contents of a cell as they would appear in the formula bar. The contents are given as text, for example, " $=2*PI()/360$ ". If the formula contains references, they are returned as R1C1-style references, such as " $=RC[1]*(1+R1C1)$ ". Use GET.FORMULA to get a formula from a cell in order to edit its arguments. Use GET.CELL(6) to get a formula in either A1 or R1C1 format, depending on the workspace setting.

Syntax

GET.FORMULA(reference)

Reference is a cell or range of cells on a sheet or macro sheet.

- If a range of cells is selected, GET.FORMULA returns the contents of the upper-left cell in reference.
- Reference can be an external reference.
- Reference can be the object identifier of a picture created by the camera tool.
- Reference can also be a reference to a chart series in the form "Sn" where n is the number of the series. When a chart series is specified, GET.FORMULA returns the series formula using R1C1-style references.

Tip If you want to get the formula in the active cell, use the ACTIVE.CELL function as the reference argument.

Examples

If cell A3 on the active sheet contains the number 523, then:

GET.FORMULA(!A\$3) equals "523"

If cell C2 on the active sheet contains the formula $=B2*(1+\$A\$1)$, then:

GET.FORMULA(!C\$2) equals " $=RC[-1]*(1+R1C1)$ "

The following macro formula returns the contents of the active cell on the active sheet:

GET.FORMULA(ACTIVE.CELL())

Related Functions

<u>GET.CELL</u>	Returns information about the specified cell
<u>GET.DEF</u>	Returns a name matching a definition
<u>GET.NAME</u>	Returns the definition of a name
<u>GET.NOTE</u>	Returns characters from a note

List of [Information Functions](#)

GET.LINK.INFO

Macro Sheets Only

Returns information about the specified link. Use GET.LINK.INFO to get information about the update settings of a link.

Syntax

GET.LINK.INFO(link_text, type_num, type_of_link, reference)

Link_text is the path of the link as displayed in the Links dialog box, which appears when you choose the Links command from the Edit menu. The path to the file you wish to return DDE information on must be surrounded by single quotes.

Type_num is a number that specifies what type of information about the currently selected link to return. Type_num 2 applies only to publishers and subscribers in Microsoft Excel for the Macintosh.

Type_num	Returns
1	If the link is set to automatic update, returns 1; otherwise 2.
2	Date of the latest edition as a serial number. Returns #N/A if link_text is not a publisher or a subscriber.

Type_of_link is a number from 1 to 6 that specifies what type of link you want to get information about.

Type_of_link	Link document type
1	Not applicable
2	DDE link (Microsoft Windows)
3	Not applicable
4	Not applicable
5	Publisher (Macintosh)
6	Subscriber (Macintosh)

Reference specifies the cell range in R1C1 format of the publisher or subscriber that you want information about. Reference is required if you have more than one publisher or subscriber of a single edition name on the active workbook. Use reference to specify the location of the subscriber you want to return information about. If the subscriber is a picture, or if the publisher is an embedded chart, reference is the number of the object as displayed in the Name box.

Remarks

- If Microsoft Excel cannot find link_text, or if type_of_link does not match the link specified by link_text, GET.LINK.INFO returns the #VALUE! error value.
- If you have more than one subscriber to the edition link_text or if the same area is published more than once, you must specify reference.

Example

In Microsoft Excel for Windows, the following macro formula returns information about a DDE link to a Microsoft Word for Windows document. The document is named NEWPROD.DOC.

```
GET.LINK.INFO("WinWord|'C:\WINWORD\NEWPROD.DOC'!DDE_LINK1", 1, 2)
```

In Microsoft Excel for the Macintosh, the following macro formula returns information about a link to a publisher defined in cells A1:C3 on a workbook named New Products.

```
GET.LINK.INFO("A1:C3 New Products Edition #1", 2, 5, "'New Products'!  
R1C1:R3C3")
```

Related Functions

[CREATE.PUBLISHER](#)

Creates a publisher from the selection

[SUBSCRIBE.TO](#)

Inserts contents of an edition into the active workbook

[UPDATE.LINK](#)

Updates a link to another workbook

List of [Information Functions](#)

GET.NAME

Macro Sheets Only

Returns the definition of a name as it appears in the Refers To box of the Define Name dialog box, which appears when you choose the Define command from the Name submenu on the Insert menu. If the definition contains references, they are given as R1C1-style references. Use GET.NAME to check the value defined by a name. To get the name corresponding to a definition, use GET.DEF.

Syntax

GET.NAME(name_text, info_type)

Name_text can be a name defined on the macro sheet; an external reference to a name defined on the active workbook, for example, "!Sales"; or an external reference to a name defined on a particular open workbook, for example, "[Book1]SHEET1!Sales". Name_text can also be a hidden name.

Info_type specifies the type of information to return about the name. If 1 or omitted, the definition is returned. If 2, returns TRUE if the name is defined for just the sheet, FALSE if the name is defined for the entire workbook.

Remarks

If the Contents check box has been selected in the Protect Sheet dialog box to protect the workbook containing the name, GET.NAME returns the #N/A error value. To see the Protect Sheet dialog box, choose the Protect Sheet command on the Protection submenu from the Tools menu.

Examples

If the name Sales on a macro sheet is defined as the number 523, then:

`GET.NAME("Sales")` equals `"=523"`

If the name Profit on the active sheet is defined as the formula =Sales-Costs, then:

`GET.NAME("!Profit")` equals `"=Sales-Costs"`

If the name Database on the active sheet is defined as the range A1:F500, then:

`GET.NAME("!Database")` equals `"=R1C1:R500C6"`

Related Functions

<u>DEFINE.NAME</u>	Defines a name on the active or macro sheet
<u>GET.CELL</u>	Returns information about the specified cell
<u>GET.DEF</u>	Returns a name matching a definition
<u>NAMES</u>	Returns the names defined in a workbook
<u>SET.NAME</u>	Defines a name as a value

List of Information Functions

GET.NOTE

Macro Sheets Only

Returns characters from a note.

Syntax

GET.NOTE(cell_ref, start_char, num_chars)

Cell_ref is the cell to which the note is attached. If cell_ref is omitted, the note attached to the active cell is returned.

Start_char is the number of the first character in the note to return. If start_char is omitted, it is assumed to be 1, the first character in the note.

Num_chars is the number of characters to return. Num_chars must be less than or equal to 255. If num_chars is omitted, it is assumed to be the length of the note attached to cell_ref.

Examples

The following macro formula returns the first 200 characters in the note attached to cell A3 on the active sheet:

```
GET.NOTE(!$A$3, 1, 200)
```

In Microsoft Excel for Windows, the following macro formula returns the 10th through the 39th characters of the note attached to cell C2 on SALES.XLS:

```
GET.NOTE("[SALES.XLS]Sheet1!R2C3", 10, 30)
```

In Microsoft Excel for the Macintosh, the following macro formula returns the 10th through the 39th characters of the note attached to cell C2 on SALES:

```
GET.NOTE("[SALES]Sheet1!R2C3", 10, 30)
```

Use GET.NOTE with the NOTE function to move the contents of a note to a cell or text box or to another cell note:

```
NOTE(GET.NOTE(!$B$10), ACTIVE.CELL())
```

Related Functions

<u>GET.CELL</u>	Returns information about the specified cell
<u>NOTE</u>	Creates or changes a cell note.
<u>SOUND.NOTE</u>	Records or imports sound into or erases sound from cell notes

List of [Information Functions](#)

GET.TOOLBAR

Macro Sheets Only

Returns information about one toolbar or all toolbars. Use GET.TOOLBAR to get information about a toolbar to use with functions that add, delete, or alter toolbars.

Syntax

GET.TOOLBAR(type_num, bar_id)

Type_num specifies what type of information to return. If type_num is 8 or 9, GET.TOOLBAR returns an array of names or numbers of all visible or hidden toolbars. Otherwise, bar_id is required, and GET.TOOLBAR returns the requested information about the specified toolbar.

Type_num	Returns
1	A horizontal array of all tool IDs on the toolbar, ordered by position. Gaps are represented by zeros.
2	Number indicating the horizontal position (x-coordinate) of the toolbar in the docked or floating region. For more information, see SHOW.TOOLBAR.
3	Number indicating the vertical position (y-coordinate) of the toolbar in the docked or floating region.
4	Number indicating the width of the toolbar in points.
5	Number indicating the height of the toolbar in points.
6	Number indicating the toolbar location: 1 = Top dock in the workspace 2 = Left dock in the workspace 3 = Right dock in the workspace 4 = Bottom dock in the workspace 5 = Floating
7	If the toolbar is visible, returns TRUE. If the toolbar is hidden, returns FALSE.
8	An array of toolbar IDs (names or numbers in the bar_id array) for all toolbars, visible and hidden.
9	An array of toolbar IDs (names or numbers in the bar_id array) for all visible toolbars.
10	If the toolbar is visible in full-screen mode, returns TRUE; otherwise, returns FALSE.
Bar_id	specifies the number or name of a toolbar for which you want information. If type_num is 8 or 9, Microsoft Excel ignores bar_id. For detailed information about bar_id, see ADD.TOOL.

Remarks

If you request position information for a hidden toolbar, Microsoft Excel returns the position where the toolbar would appear if shown.

Examples

The following macro formula returns information about the width of Toolbar1:

```
GET.TOOLBAR(4, "Toolbar1")
```

When the following macro formula is entered as an array with CTRL+SHIFT+ENTER, the IDs of all visible toolbars are returned, and the array is named All_Bar_Ids:

```
SET.NAME("All_Bar_Ids", GET.TOOLBAR(9))
```

Related Functions

[ADD.TOOLBAR](#)

Creates a new toolbar with the specified tools

[DELETE.TOOLBAR](#)

Deletes custom toolbars

[GET.TOOL](#)

Returns information about a tool or tools on a toolbar

[SHOW.TOOLBAR](#)

Hides or displays a toolbar

List of [Information Functions](#)

GRIDLINES

Macro Sheets Only

Equivalent to choosing the Gridlines command from the Insert menu. Allows you to turn chart gridlines on and off.

Arguments are logical values corresponding to the check boxes in the Gridlines dialog box. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If omitted, the setting is not changed. If a chart is not active, produces a error and halts the macro.

Syntax

GRIDLINES(x_major, x_minor, y_major, y_minor, z_major, z_minor, 2D_effect)

GRIDLINES?(x_major, x_minor, y_major, y_minor, z_major, z_minor, 2D_effect)

X_major corresponds to the Category (X) Axis: Major Gridlines check box.

X_minor corresponds to the Category (X) Axis: Minor Gridlines check box.

Y_major corresponds to the Value (Y) Axis: Major Gridlines check box. On 3-D charts, y_major corresponds to the Series (Y) Axis: Major Gridlines check box.

Y_minor corresponds to the Value (Y) Axis: Minor Gridlines check box. On 3-D charts, y_minor corresponds to the Series (Y) Axis: Minor Gridlines check box.

Z_major corresponds to the Value (Z) Axis: Major Gridlines check box (3-D only).

Z_minor corresponds to the Value (Z) Axis: Minor Gridlines check box (3-D only).

2D_effect corresponds to the 2-D Walls and Gridlines check box (3-D only).

Related Functions

List of [Command-Equivalent Functions](#)

HELP

Macro Sheets Only

Starts or switches to Help and displays the specified custom Help topic. Use HELP with custom Help files to create your own Help system, which can be used just like the built-in Microsoft Excel Help.

Syntax

HELP(help_ref)

Help_ref is a reference to a topic in a Help file, in the form "filename!topic_number".

- Help_ref must be given as text.
- If help_ref is omitted, HELP displays the Contents topic for Microsoft Excel Help.

Remarks

- Microsoft Excel for Windows does not support the use of Help files in the text file format for custom Help.
- If you want to create custom Help with advanced features in Microsoft Excel for Windows, you need the Microsoft Windows Help compiler version 3.1 or later. To obtain the compiler, you can purchase the Microsoft Windows Software Development Kit from your software vendor or Microsoft Corporation.
- In Microsoft Excel for the Macintosh, custom Help files are plain text files or text files with line breaks.

Tips

In Microsoft Excel for Windows, the following macro formula switches back to Microsoft Excel when Help is active:

```
APP.ACTIVATE()
```

The following macro formula closes Help when Help is active:

```
SEND.KEYS ("% {F4} ")
```

Examples

In Microsoft Excel for Windows, the following macro formula displays the Help topic numbered 101 in the file CUSTHELP.DOC. The Help window remains open if the user switches to another window or application.

```
HELP ("CUSTHELP.DOC!101")
```

If the custom Help file is not in the current directory, specify the full path along with the name of the file. For example:

```
HELP ("C:\EXCEL\CUSTHELP.DOC!101")
```

In Microsoft Excel for the Macintosh, the following macro formula displays the Help topic numbered 101 in the file CUSTOM HELP:

```
HELP ("CUSTOM HELP!101")
```

If the custom Help file is not in the current folder, specify the full path along with the name of the file. For example:

```
HELP ("HARD DISK:EXCEL:HELP:CUSTOM HELP!101")
```

Related Functions

List of [Customizing Functions](#)

HIDE.OBJECT

Macro Sheets Only

Hides or displays the specified object.

Syntax

HIDE.OBJECT(object_id_text, hide)

Object_id_text is the name and number, or number alone, of the object, as text, as it appears in the reference area when the object is selected. The name of the object is also the text returned by the CREATE.OBJECT function, so object_id_text can be a reference to a cell containing CREATE.OBJECT. To give the name of more than one object, use the following format for object_id_text:

"oval 3, text 2, arc 5"

If object_id_text is omitted, the function operates on all selected objects. If no object is selected or if the object specified by object_id_text does not exist, HIDE.OBJECT returns the #VALUE! error value.

Hide is a logical value that specifies whether to hide or display the specified object. If hide is TRUE or omitted, Microsoft Excel hides the object; if FALSE, Microsoft Excel displays the object.

Remarks

Objects are not automatically selected after they are unhidden.

Examples

The following macro formula hides the selected object:

```
HIDE.OBJECT(, TRUE)
```

The following macro formula displays the object named Oval 3:

```
HIDE.OBJECT("Oval 3", FALSE)
```

The following macro formula displays the three specified objects:

```
HIDE.OBJECT("oval 3, text 2, arc 5", FALSE)
```

Related Functions

CREATE.OBJECT Creates an object

DISPLAY Controls how an object is displayed

List of Customizing Functions

INPUT

Macro Sheets Only

Displays a dialog box for user input. Returns the information entered in the dialog box. Use INPUT to display a simple dialog box for the user to enter information to be used in a macro.

The dialog box has an OK and a Cancel button. If you choose the OK button, INPUT returns the default value specified or the value typed in the edit box. If you choose the Cancel button, INPUT returns FALSE.

Syntax

INPUT(message_text, type_num, title_text, default, x_pos, y_pos, help_ref)

Message_text is the text to be displayed in the dialog box. Message_text must be enclosed in quotation marks.

Type_num is a number specifying the type of data to be entered.

Type_num	Data type
0	Formula
1	Number
2	Text
4	Logical
8	Reference
16	Error
64	Array

You can also use the sum of the allowable data types for type_num. For example, for an input box that can accept formulas, text, or numbers, set type_num equal to 3 (the sum of 0, 1, and 2, which are the type specifiers for formula, number, and text). If type_num is omitted, it is assumed to be 2.

- If type_num is 0, INPUT returns the formula in the form of text, for example, " $=2*PI()/360$ ".
- To enter a formula, include an equal sign at the beginning of the formula; otherwise the formula is returned as text.
- If the formula contains references, they are returned as R1C1-style references, for example, " $=RC[-1]*(1+R1C1)$ ".
- If type_num is 8, INPUT returns an absolute reference to the specified cells.
- If you enter a single-cell reference in the dialog box, the value in that cell is returned by the INPUT function.
- If the information entered in the dialog box is not of the correct data type, Microsoft Excel attempts to convert it to the specified type. If the information can't be converted, Microsoft Excel displays an error message.

Title_text is text specifying a title to be displayed in the title bar of the dialog box. If title_text is omitted, it is assumed to be "Input".

Default specifies a value to be shown in the edit box when the dialog box is initially displayed. If default is omitted, the edit box is left empty.

X_pos, y_pos specify the horizontal and vertical position, in points, of the dialog box. A point is 1/72nd of an inch. If either or both arguments are omitted, the dialog box is centered in the corresponding direction.

Help_ref is a reference to a custom online Help topic in a text file, in the form "filename!topic_number".

- If help_ref is present, a Help button appears in the lower-right corner of the dialog box. Choosing the Help button starts Help and displays the specified topic.
- If help_ref is omitted, no Help button appears.
- Help_ref must be given as text.

For more information about custom Help topics, see HELP.

Remarks

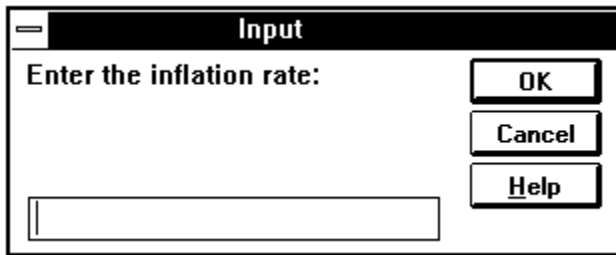
Relative references entered in formulas in the INPUT dialog box are relative to the active cell at the time the INPUT function is calculated. If you are using the reference entered into the dialog box in a cell other

than the active cell, it may not refer to the cells you intend it to. For example, if the active cell is A3 and you enter the formula " $=A1+A2$ " in an INPUT dialog box, intending to add the values in those cells, and then use the FORMULA function to enter the formula in cell B3, the formula in cell B3 will read " $=B1+B2$ " because you gave a relative reference. You can use FORMULA.CONVERT to solve this problem.

Examples

In Microsoft Excel for Windows, the following macro formula displays the following dialog box:

```
INPUT("Enter the inflation rate:", 1, "Inflation Rate", , , , "CUSTHELP.DOC!101")
```



If you then enter 12%, INPUT returns the value 0.12.

In Microsoft Excel for the Macintosh, the following macro formula displays the following dialog box:

```
INPUT("Enter the inflation rate:", 1, "Inflation Rate", , , , "CUSTOM HELP!101")
```

■

If you then enter 12%, INPUT returns the value 0.12.

If the active cell is C2 and you enter the formula $=B2*(1+\$A\$1)$ in response to the following macro formula:

```
INPUT("Enter your monthly increase formula:", 0)
```

INPUT returns " $=RC[-1]*(1+R1C1)$ "

If you select the range $\$A\$2:\$A\8 in the INPUT dialog box:

```
REFTEXT(INPUT("Please make your selection.", 8)) returns R2C1:R8C1
```

Related Functions

<u>ALERT</u>	Displays a dialog box and a message
<u>DIALOG.BOX</u>	Displays a custom dialog box
<u>FORMULA.CONVERT</u>	Changes the style and type of references in a formula
<u>HELP</u>	Displays a custom Help topic
List of <u>Customizing Functions</u>	

INSERT

Macro Sheets Only

Inserts a blank cell or range of cells or pastes cells from the Clipboard into a sheet. Shifts the selected cells to accommodate the new ones. The size and shape of the inserted range are the same as those of the current selection.

Syntax

INSERT(shift_num)

INSERT?(shift_num)

Shift_num is a number from 1 to 4 specifying which way to shift the cells. If an entire row or column is selected, shift_num is ignored. If shift_num is omitted, Microsoft Excel shifts cells in the logical direction based on the selection.

Shift_num	Direction
1	Shift cells right
2	Shift cells down
3	Shift entire row
4	Shift entire column

Remarks

If you have just cut or copied information to the Clipboard, INSERT performs both an insert and a paste operation. First, Microsoft Excel inserts new blank cells into the sheet; then, Microsoft Excel pastes information from the Clipboard into the newly inserted cells. If you have used the INSERT function in macros written for Microsoft Excel version 2.2 or earlier, make sure you consider this feature when you use your old macros with later versions of Microsoft Excel.

Related Functions

COPY Copies and pastes data or objects

CUT Cuts or moves data or objects

EDIT.DELETE Removes cells from a sheet

PASTE Pastes cut or copied data

List of Command-Equivalent Functions

LEGEND

Macro Sheets Only

Equivalent to choosing the Legend command from the Insert menu or the Clear command from the Edit menu when a chart legend is selected. Adds a legend to or removes a legend from a chart. This is also equivalent to choosing the Legend button on the Chart toolbar when a chart is active.

Syntax

LEGEND(logical)

Logical is a logical value specifying which command LEGEND is equivalent to.

- If logical is TRUE or omitted, LEGEND is equivalent to the Legend command on the Insert menu.
- If logical is FALSE, LEGEND is equivalent to the Delete command on the Edit menu.
- If logical is FALSE and the active chart has no legend, LEGEND takes no action.

Related Function

FORMAT.LEGEND

Determines the position and orientation of the legend on a chart

List of Command-Equivalent Functions

LINE.PRINT

Macro Sheets Only

Prints the active worksheet using methods compatible with those of Lotus 1-2-3. LINE.PRINT does not use the Microsoft Windows printer drivers. Unless you have a specific need for the LINE.PRINT function, use the PRINT function instead.

Note This function is only available in Microsoft Excel for Windows.

Syntax 1

Go, Line, Page, Align, and Clear

LINE.PRINT(command, file, append)

Syntax 2

Worksheet settings

LINE.PRINT(command, setup_text, leftmarg, rightmarg, topmarg, botmarg, pglen, formatted)

Syntax 3

Global settings

LINE.PRINT(command, setup_text, leftmarg, rightmarg, topmarg, botmarg, pglen, wait, autolf, port, update)

Command is a number corresponding to the command you want LINE.PRINT to carry out. For syntax 2, command must be 5. For syntax 3, command must be 6.

Command	Command that is carried out
1	Go
2	Line
3	Page
4	Align
5	Worksheet settings
6	Global settings (saved in EXCEL5.INI)
7	Clear (change to current global settings)

File is the name of a file to which you want to print. If omitted, Microsoft Excel prints to the printer port determined by the current global settings.

Append is a logical value specifying whether to append text to file. If TRUE, the file you are printing is appended to file; if FALSE or omitted, the file you are printing overwrites the contents of file.

Setup_text is text that includes a printer initialization sequence or other control codes to prepare your printer for printing. If omitted, no setup text is used.

Leftmarg is the size of the left margin measured in characters from the left side of the page. If omitted, it is assumed to be 4.

Rightmarg is the size of the right margin measured in characters from the left side of the page. If omitted, it is assumed to be 76.

Topmarg is the size of the top margin measured in lines from the top of the page. If omitted, it is assumed to be 2.

Botmarg is the size of the bottom margin measured in lines from the bottom of the page. If omitted, it is assumed to be 2.

Pglen is the number of lines on one page. If omitted, it is assumed to be 66 (11 inches with 6 lines per inch). If you're using an HP LaserJet or compatible printer, set pglen to 60 (the printer reserves six lines).

Formatted is a logical value specifying whether to format the output. If TRUE or omitted, the output is formatted; if FALSE, it is not formatted.

Wait is a logical value specifying whether to wait after printing a page. If TRUE, Microsoft Excel waits; if FALSE or omitted, Microsoft Excel continues printing.

Autolf is a logical value specifying whether your printer has automatic line feeding. If TRUE, Microsoft

Excel prints lines normally; if FALSE or omitted, Microsoft Excel sends an additional line feed character after printing each line.

Port is a number from 1 to 8 specifying which port to use when printing.

Port	Port used when printing
1 or omitted	LPT1
2	COM1
3	LPT2
4	COM2
5	LPT1
6	LPT2
7	LPT3
8	LPT4

Update is a logical value specifying whether to update and save global settings. If TRUE, the settings are saved in the EXCEL5.INI file; if FALSE or omitted, the global settings are not saved.

Remarks

The default values for print settings on your worksheet are determined by the current global settings.

Example

The following macro formula prints the currently defined print area to the currently defined printer port:

```
LINE.PRINT(1)
```

Related Functions

PRINT Prints the active document

List of Command-Equivalent Functions

MAIN.CHART

Macro Sheets Only

Equivalent to choosing the Main Chart command from the Format menu when a chart document is active in Microsoft Excel version 2.2 or earlier. This function is included only for macro compatibility.

Syntax

MAIN.CHART(type_num, stack, 100, vary, overlap, drop, hilo, overlap%, cluster, angle)

MAIN.CHART?(type_num, stack, 100, vary, overlap, drop, hilo, overlap%, cluster, angle)

Related Functions

List of [Command-Equivalent Functions](#)

MAIN.CHART.TYPE

Macro Sheets Only

Equivalent to choosing the Main Chart Type command from the Chart menu in Microsoft Excel for the Macintosh version 1.5 or earlier. This function is included only for macro compatibility.

Syntax

MAIN.CHART.TYPE(type_num)

Related Functions

List of Command-Equivalent Functions

MESSAGE

Macro Sheets Only

Displays and removes messages in the message area of the status bar. MESSAGE is useful for displaying text that doesn't need a response, such as descriptions of commands in user-defined menus.

Syntax

MESSAGE(logical, text)

Logical is a logical value specifying whether to display or remove a message.

- If logical is TRUE, Microsoft Excel displays text in the message area of the status bar.
- If logical is FALSE, Microsoft Excel removes any messages, and the status bar is returned to normal (that is, command help messages are displayed).

Text is the message you want to display in the status bar. If text is "" (empty text), Microsoft Excel removes any messages currently displayed in the status bar.

Remarks

- Only one message can be displayed in the status bar at a time. Messages are always displayed in the same place.
- MESSAGE works the same way whether the status bar is displayed or not. You can, for example, use MESSAGE while the status bar isn't displayed. As soon as you display the status bar, you see your message.
- If you display any message (even empty text) and don't remove it with MESSAGE(FALSE), that message is displayed until you quit Microsoft Excel.
- You can also use the ALERT function to get the user's attention; however, this interrupts the macro and requires the user's intervention before the macro can continue.

Example

The following lines in a macro display a message warning that you must wait for a moment while the macro calls a subroutine.

```
MESSAGE(TRUE, "One moment please...")
```

Related Functions

ALERT Displays a dialog box and a message

BEEP Sounds a tone

List of Command-Equivalent Functions

NEW.WINDOW

Macro Sheets Only

Equivalent to choosing the New Window command from the Window menu. Creates a new window for the active workbook.

Syntax

NEW.WINDOW()

After you use NEW.WINDOW, use the WINDOW.MOVE, WINDOW.SIZE, and ARRANGE.ALL functions to size and position the new window.

Related Functions

ARRANGE.ALL Arranges all displayed windows to fill the workspace and synchronizes windows for scrolling

WINDOW.MOVE Moves a window

WINDOW.SIZE Changes the size of a window

List of Command-Equivalent Functions

OBJECT.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Properties tab in the Object dialog box, which appears when you choose the Object command from the Format menu. Determines how the selected object or objects are attached to the cells beneath them and whether they are printed. The way an object is attached to the cells beneath it affects how the object is moved or sized whenever you move or size the cells.

OBJECT.PROPERTIES(placement_type, print_object)

OBJECT.PROPERTIES?(placement_type, print_object)

Placement_type is a number from 1 to 3 specifying how to attach the selected object or objects. If placement_type is omitted, the current status is unchanged.

If placement_type is	The selected object is
1	Moved and sized with cells.
2	Moved but not sized with cells.
3	Free-floating; it is not affected by moving and sizing cells.

Print_object is a logical value specifying whether to print the selected object or objects. If TRUE or omitted, the objects are printed; if FALSE, they are not printed.

Remarks

If an object is not selected, OBJECT.PROPERTIES interrupts the macro and returns the #VALUE! error value.

Related Functions

CREATE.OBJECT Creates an object

FORMAT.MOVE Moves the selected object

FORMAT.SIZE Changes the size of the selected object

List of Command-Equivalent Functions

OBJECT.PROTECTION

Macro Sheets Only

Equivalent to choosing the Protection tab in the Format Object dialog box, which appears when you choose the Object command from the Format menu. Changes the protection status of the selected object.

Syntax

OBJECT.PROTECTION(locked, lock_text)

OBJECT.PROTECTION?(locked, lock_text)

Locked is a logical value that determines whether the selected object is locked or unlocked. If locked is TRUE, Microsoft Excel locks the object; if FALSE, Microsoft Excel unlocks the object.

Lock_text is a logical value that determines whether text in a text box or button can be changed.

Lock_text applies only if the object is a text box, button, or worksheet control. If lock_text is TRUE or omitted, text cannot be changed; if FALSE, text can be changed.

Remarks

- You cannot lock or unlock an individual object with OBJECT.PROTECTION when document protection is selected for objects in the Protect Sheet dialog box.
- If an object is not selected, the function returns the #VALUE! error value and halts the macro.
- In order for an object to be protected, you must use the PROTECT.DOCUMENT(, , TRUE) function after changing the object's status with OBJECT.PROTECTION.

Related Functions

PROTECT.DOCUMENT Controls protection for the active worksheet

WORKBOOK.PROTECT Controls protection for the active workbook

List of Command-Equivalent Functions

OPEN.MAIL

Macro Sheets Only

Equivalent to choosing the Open Mail command from the Mail submenu on File menu.

Note This function is available for only Microsoft Excel for the Macintosh and Microsoft Mail version 2.0 or later.

Syntax

OPEN.MAIL(subject, comments)

OPEN.MAIL?(subject, comments)

Subject is the subject of the message containing a file that Microsoft Excel can open.

- For each message whose subject matches the subject argument and contains a file that Microsoft Excel can open, the file is opened in Microsoft Excel; if the message has no unread enclosures, it is deleted from the list of pending mail.
- If subject is omitted, then for all messages containing files that Microsoft Excel can open, the files are opened; each message that has no unread enclosures is deleted from the list of pending mail.

Comments is a logical value that specifies whether comments associated with the Microsoft Excel files are displayed. If comments is TRUE, Microsoft Excel displays the comments; if FALSE, comments are not displayed. If omitted, the current setting is not changed.

Tips

- If you use consistent subjects in your Microsoft Mail messages, you can easily create a macro that always opens mail messages with certain files attached. For example, an OPEN.MAIL formula with subject specified as "Weekly Report" would open the Microsoft Excel file attached to the message containing that subject each week.
 - In OPEN.MAIL?, the dialog-box form of the function, the currently running macro pauses while the Microsoft Mail documents window is displayed. The macro resumes after you close the Microsoft Mail documents window.
-

Related Function

SEND.MAIL Sends the active workbook

List of Command-Equivalent Functions

PASTE.PICTURE

Macro Sheets Only

Equivalent to choosing the Paste Picture command from the Edit menu while holding down the SHIFT key in Microsoft Excel version 4.0. Pastes a picture of the Clipboard contents onto the sheet. This picture is not linked, so changes to the source data will not be reflected in the picture. In Microsoft Excel for Windows version 5.0, use INSERT.PICTURE to import pictures.

Syntax

PASTE.PICTURE()

Related Functions

[COPY.PICTURE](#)

Creates a picture of the current selection for use in another program

[INSERT.PICTURE](#)

Inserts a picture from a file

[PASTE.PICTURE.LINK](#)

Pastes a linked picture of the currently copied area

List of [Command-Equivalent Functions](#)

PASTE.PICTURE.LINK

Macro Sheets Only

Equivalent to holding down the SHIFT key and choosing the Paste Picture Link command from the Edit menu in Microsoft Excel version 4.0 or to using the camera tool on the utility toolbar. Pastes a linked picture of the Clipboard contents. This picture is linked, so changes to the source data will be reflected in the picture.

Syntax

PASTE.PICTURE.LINK()

Related Functions

[COPY.PICTURE](#) Creates a picture of the current selection for use in another program

[CREATE.OBJECT](#) Creates an object

[PASTE.PICTURE](#) Pastes a picture of the currently copied area

List of [Command-Equivalent Functions](#)

PREFERRED

Macro Sheets Only

Equivalent to choosing the Preferred command from the Gallery menu in Microsoft Excel version 4.0. Changes the format of the active chart to the format currently defined by the Default Chart Template option in the Chart tab of the Options dialog box or the SET.PREFERRED macro function.

Syntax

PREFERRED()

Related Function

SET.PREFERRED Changes the default chart format

List of Command-Equivalent Functions

PRINTER.SETUP

Macro Sheets Only

Equivalent to choosing the Print command from the File menu and then choosing the Printer button in Microsoft Excel for Windows. Use PRINTER.SETUP to change the printer you are using.

Syntax

PRINTER.SETUP(printer_text)

PRINTER.SETUP?(printer_text)

Printer_text is the name of the printer you want to switch to. Enter printer_text exactly as it appears in the Setup dialog box.

Note This function is only available in Microsoft Excel for Windows.

Related Functions

PAGE.SETUP Sets page printing specifications

PRINT Prints the active workbook

List of Command-Equivalent Functions

REMOVE.PAGE.BREAK

Macro Sheets Only

Equivalent to choosing the Remove Page Break command from the Insert menu. Removes manual page breaks that you set with the SET.PAGE.BREAK function or the Page Break command on the Insert menu. If the active cell is not below or to the right of a manual page break, REMOVE.PAGE.BREAK takes no action. If the entire sheet is selected, REMOVE.PAGE.BREAK removes all manual page breaks. REMOVE.PAGE.BREAK does not remove automatic page breaks.

Syntax

REMOVE.PAGE.BREAK()

Related Function

SET.PAGE.BREAK Sets manual page breaks

List of Command-Equivalent Functions

SELECT.CHART

Macro Sheets Only

Equivalent to the Select Chart command on the Chart menu in Microsoft Excel version 4. This function is equivalent to using the third form of SELECT with "Chart" as the item_text argument.

Syntax

SELECT.CHART()

Remarks

This function is included for compatibility with macros written with Microsoft Excel for the Macintosh version 1.5 or earlier.

Related Functions

SELECT Selects a chart object

List of Command-Equivalent Functions

SELECT.PLOT.AREA

Macro Sheets Only

Equivalent to choosing the Select Plot Area command from the Chart menu in Microsoft Excel version 4.
Selects the plot area of the active chart.

Syntax

SELECT.PLOT.AREA()

Remarks

SELECT.PLOT.AREA is included only for compatibility with previous versions of Microsoft Excel for the Macintosh. SELECT.PLOT.AREA is the same as the SELECT("Plot") function.

Related Function

SELECT Selects a cell, graphic object, or chart

List of Command-Equivalent Functions

SELECTION

Macro Sheets Only

Returns the reference or object identifier of the selection as an external reference. Use SELECTION to return information about the current selection for use in other macro formulas.

Syntax

SELECTION()

If a cell or range of cells is selected, Microsoft Excel returns the corresponding external reference. If an object is selected, Microsoft Excel returns the object identifier listed in the following table.

Item selected	Identifier returned
Imported graphic	Picture n
Linked graphic	Picture n
Chart picture	Picture n
Linked chart	Chart n
Range	Picture n
Linked range	Picture n
Text box	Text n
Button	Button n
Rectangle	Rectangle n
Oval	Oval n
Line	Line n
Arc	Arc n
Group	Group n
Freehand drawing or polygon	Drawing n

SELECTION also returns the identifiers of chart items. The identifiers returned are the same as the identifiers you specify when you use the SELECT function. For a list of these identifiers, see the description of item_text in SELECT.

If you select cells and use the value returned by SELECTION in a function or operation, you usually get the value contained in the selection instead of its reference. References are automatically converted to the contents of the reference. If you want to work with the actual reference, use SET.NAME to assign a name to it, even if the reference refers to objects. See the last example following. You can also use the REFTXT function to convert the reference to text, which you can then store or manipulate.

Remarks

- If an object is selected, SELECTION returns the identifier of the object. If multiple objects are selected, it returns the identifiers of all the selected objects, as a string separated by commas.
- If more than 1024 characters would be returned, SELECTION returns the #VALUE! error value.

Examples

If the sheet in the active window is named SHEET1 in the workbook BOOK1, and if A1:A3 is the selection, then:

```
SELECTION() equals [BOOK1]SHEET1!A1:A3
```

The following macro formula moves the current selection one row down:

```
SELECT(OFFSET(SELECTION(), 1, 0))
```

The above formula is particularly useful for moving incrementally through a database to add or modify records.

The following macro formula defines the name "EntryRange" on the active sheet to refer to one row below the current selection on the active sheet:

```
DEFINE.NAME("EntryRange", OFFSET(SELECTION(), 1, 0))
```

The following macro formula defines the name "Objects" on your macro sheet to refer to the object

names in the current multiple selection:

```
SET.NAME("Objects", SELECTION())
```

Related Functions

<u>ACTIVE.CELL</u>	Returns the reference of the active cell
<u>OFFSET</u>	Returns a reference offset from a given reference
<u>SELECT</u>	Selects a cell, graphic object, or chart

List of Information Functions

SEND.MAIL

Macro Sheets Only

Equivalent to choosing the Send Mail command from the File menu. Sends the active workbook using email.

Syntax

SEND.MAIL(recipients, subject, return_receipt)

SEND.MAIL?(recipients, subject, return_receipt)

Important To use SEND.MAIL in Microsoft Excel for Windows, you must be using a mail client that supports the Messaging Applications Programming Interface (MAPI) or Vendor-Independent Messaging (VIM). To use SEND.MAIL in Microsoft Excel for the Macintosh, you must be using Microsoft Mail version 2.0 or later.

Recipients is the name of the person to whom you want to send the mail. The name should be given as text.

- To specify more than one name, give the list of names as an array. For example, SEND.MAIL({"John", "Paul", "George", "Ringo"}) would send the active workbook to the four names in the array. You can also refer to a range on a sheet or macro sheet that contains a list of names to whom you want the mail to be sent.
- To send mail to users on different Microsoft Mail for the Macintosh servers, specify the server name along with the user name. The following text, as the recipients argument, sends mail to wandagr on server2, gregpr on the current server, and victorge on server7:
{"wandagr@server2", "gregpr", "victorge@server7"}

Subject is a text string that specifies the subject of the message. If subject is omitted, the name of the active workbook is used as the subject.

Return_receipt is a logical value that corresponds to the Return Receipt check box. If return_receipt is TRUE, Microsoft Excel selects the check box and sends a return receipt; if FALSE or omitted, Microsoft Excel clears the check box.

Related Function

[OPEN.MAIL](#) Opens files sent via Microsoft Mail that Microsoft Excel can open

List of [Command-Equivalent Functions](#)

SET.PAGE.BREAK

Macro Sheets Only

Equivalent to choosing the Page Break command from the Insert menu. Sets manual page breaks for a printed workbook. Use SET.PAGE.BREAK to override the automatic page breaks. Setting a manual page break changes the automatic page breaks that follow it.

The page break occurs above and to the left of the active cell and appears as dotted lines if you have set up a printer. If the active cell is in column A, a manual page break is added only above the cell. If the active cell is in row 1, a manual page break is added only at the left edge of the cell. If the row or column next to the active cell already has a page break, SET.PAGE.BREAK takes no action.

Syntax

SET.PAGE.BREAK()

Related Functions

[PRINT.PREVIEW](#)

Previews pages and page breaks before printing

[REMOVE.PAGE.BREAK](#)

Removes manual page breaks

List of [Command-Equivalent Functions](#)

SET.PREFERRED

Macro Sheets Only

Equivalent to setting the Default Chart Format in the Chart tab of the Options dialog box, which appears when you choose the Options command from the Tools menu. Changes the default format that Microsoft Excel uses when you create a new chart or when you format a chart PREFERRED macro function.

When you use the SET.PREFERRED function, the format of the active chart becomes the preferred format.

Syntax

SET.PREFERRED(format)

Format is the name of the format that you want as the default format for charts. If omitted, the format of the currently active chart is used. If format is "Built_in", then Microsoft Excel will use the standard, built-in chart as the default. If the chart was created in Microsoft Excel 4.0 and if format is "PREFERRED", then the preferred chart format used in Microsoft Excel 4.0 will be used. Format is case sensitive.

Related Function

PREFERRED Changes the format of the active chart to the preferred format

List of Command-Equivalent Functions

SOUND.NOTE

Macro Sheets Only

Records sound into or erases sound from a cell note or imports sound from another file into a cell note. This function requires that you have recording hardware installed in your computer, and you must be running Microsoft Windows version 3.1 or later, or Apple system software version 6.07 or later.

Syntax 1

Recording or erasing sound

SOUND.NOTE(cell_ref, erase_snd)

Syntax 2

Importing sound from another file

SOUND.NOTE(cell_ref, file_text, resource)

Cell_ref is a reference to the cell containing a note into which you want to record or import sounds or from which you want to erase a sound.

Erase_snd is a logical value specifying whether to erase the sound in the note. If erase_snd is TRUE, Microsoft Excel erases only the sound from the note. If FALSE or omitted, Microsoft Excel displays the Record dialog box so that you can record sound into the note.

File_text is the name of a file containing sounds.

Resource is the number or name of a sound resource in file_text that you want to import into your note.

- This argument applies only to Microsoft Excel for the Macintosh.
- If resource is omitted, Microsoft Excel uses the first resource in the file.
- If the file does not contain a sound resource with the specified name or number, Microsoft Excel halts the macro and displays an error message.

Remarks

To find out if a cell has sound attached to it, use GET.CELL(47).

Examples

The following macro formula erases the sound, if present, from cell A1 on the active sheet:

```
SOUND.NOTE(!$A$1, TRUE)
```

The following macro formula displays the Record dialog box so that you can record sound into a note for cell A1 on the active sheet:

```
SOUND.NOTE(!$A$1)
```

In Microsoft Excel for Windows, the following macro formula imports the sound from a file named CHIMES.WAV into a note for the cell named Doorbell on the active sheet:

```
SOUND.NOTE(!Doorbell, "C:\SOUNDS\CHIMES.WAV")
```

In Microsoft Excel for the Macintosh, the following macro formula imports a sound called Chimes from a file named SOFT SOUNDS into a note for the cell named Doorbell on the active sheet:

```
SOUND.NOTE(!Doorbell, "HARD DISK:SOUNDS:SOFT SOUNDS", "Chimes")
```

Related Functions

NOTE Creates or changes a cell note

SOUND.PLAY Plays the sound from a cell note or a file

List of Command-Equivalent Functions

SOUND.PLAY

Macro Sheets Only

Plays the sound from a cell note or a file. Equivalent to choosing the Note command from the Insert menu and choosing the Play button, or choosing the Note command from the Insert menu, choosing the Import button, and then opening a file, selecting a sound, and choosing the Play button. To play sounds in Microsoft Excel for Windows, you must have a sound board installed in your computer.

Syntax

SOUND.PLAY(cell_ref, file_text, resource)

Cell_ref is a reference to the cell note containing sound that you want to play. If cell_ref is omitted, Microsoft Excel plays the sound from the active cell, or from a file if you specify one.

File_text is the name of a file containing sounds. If cell_ref is specified, file_text is ignored.

Resource is a number or name given as text specifying a sound resource in file_text that you want to play.

- This argument applies only to Microsoft Excel for the Macintosh.
- If cell_ref is specified, resource is ignored.
- If resource is omitted, Microsoft Excel uses the first sound resource in the file.
- If the file does not contain a sound resource with the specified name or number, Microsoft Excel halts the macro and displays an error message.

Related Function

SOUND.NOTE Records or imports sound into or erases sound from cell notes

List of Command-Equivalent Functions

SPELLING.CHECK

Macro Sheets Only

Checks the spelling of a word. Returns TRUE if the word is spelled correctly; FALSE otherwise.

Syntax

SPELLING.CHECK(word_text, custom_dic, ignore_uppercase)

Word_text is the word whose spelling you want to check. It can be text or a reference to text.

Custom_dic is the filename of a custom dictionary to examine if the word is not found in the main dictionary.

Ignore_uppercase is a logical value corresponding to the Ignore Words In Uppercase check box. If ignore_uppercase is TRUE, the check box is selected, and Microsoft Excel ignores words in all uppercase letters; if FALSE, the check box is cleared, and Microsoft Excel checks all words; if omitted, the current setting is used.

Remarks

This function does not have a dialog-box form. To display the Spelling dialog box, use SPELLING.

Related Function

SPELLING Checks the spelling of words in the current selection

List of Command-Equivalent Functions

TEXT.BOX

Macro Sheets Only

Replaces characters in a text box or button with the text you specify.

Syntax

TEXT.BOX(add_text, object_id_text, start_num, num_chars)

Add_text is the text you want to add to the text box or button.

Object_id_text is the name of the text box or button to which you want to add text (for example, "Text 1" or "Button 2"). If object_id_text is omitted, it is assumed to be the selected item.

Start_num is a number specifying the position of the first character you want to replace (or the position at which you want to insert characters if you do not want to replace any). If start_num is omitted, it is assumed to be 1.

Num_chars is the number of characters you want to replace. If num_chars is 0, then no characters are replaced, and add_text is inserted starting at the position start_num. If num_chars is omitted, all the characters are replaced.

Examples

The following macro formula replaces the first five characters in a text box named "Text 5" with the text "Net Income":

```
TEXT.BOX("Net Income", "Text 5", 1, 5)
```

The following macro formula inserts the words "Account Summary for 1991" at the beginning of a text box named "Text 6":

```
TEXT.BOX("Account Summary for 1991", "Text 6", 1, 0)
```

Related Functions

CREATE.OBJECT Creates an object

FONT.PROPERTIES Applies a font to the selection

GET.OBJECT Returns information about an object

List of Command-Equivalent Functions

WINDOW.TITLE

Macro Sheets Only

Changes the title of the active window to the title you specify. The title appears at the top of the workbook window. Use WINDOW.TITLE to control window titles when you're using Microsoft Excel to create a custom application.

Syntax

WINDOW.TITLE(text)

Text is the title you want to assign to the window. If text is omitted, it is assumed to be the name of the workbook as it is stored on your disk. Empty text ("") specifies no title.

Important WINDOW.TITLE changes the name of the window, not the actual name of the workbook as it is stored on your disk. To change the name of the workbook, use the SAVE.AS function.

Remarks

- The window name you create using WINDOW.TITLE will appear on the Window menu, and will be returned by the WINDOWS function, but not by the DOCUMENTS function. You must use the new window name in the ACTIVATE function and the ON.WINDOW function.
- If you want to communicate with a Microsoft Excel workbook using DDE functions like INITIATE or REQUEST, you must specify the filename of the workbook and not the window title specified with the WINDOW.TITLE function.
- If you use NEW.WINDOW to create new windows on the workbook, the window title will be restored to its original name.

Example

The following macro formula changes the title of the active window to First Quarter.

```
WINDOW.TITLE("First Quarter")
```

Related Function

<u>APP.TITLE</u>	Changes the title of the application workspace
<u>SAVE.AS</u>	Specifies a new filename, file type, protection password, or write-reservation password, or to create a backup file

List of [Customizing Functions](#)

WINDOWS

Macro Sheets Only

Returns the names of the specified open Microsoft Excel windows, including hidden windows. Use **WINDOWS** to get a list of active windows for use by other macro functions that return information about or manipulate windows, such as **GET.WINDOW** and **ACTIVATE**. The names are returned as a horizontal array of text values, in order of their appearance on your screen. The first name is the active window, the next name is the window directly under the active window, and so on.

Syntax

WINDOWS(type_num, match_text)

Type_num is a number that specifies which types of workbooks are returned by **WINDOWS**, according to the following table.

Type_num	Returns window names from these types of documents
----------	--

1 or omitted	All windows except those belonging to add-in workbooks
2	Add-in workbooks only
3	All types of workbooks

Match_text specifies the windows whose names you want returned and can include wildcard characters. If match_text is omitted, **WINDOWS** returns the names of all open windows.

Tips

- You can change the output of a horizontal array to vertical with the **TRANSPOSE** function.
- You can use **WINDOWS** with the **INDEX** function to choose individual window names from the array for use in other functions that take window names as arguments.
- You can use the **COLUMNS** functions to count the number of entries in the array, which is the number of windows.

Examples

If the active window, named **BOOK1**, is on top of a window named **MACROS:2**, which is on top of a window named **MACROS:1**, then:

WINDOWS () equals {"BOOK1", "MACROS:2", "MACROS:1"}

Related Functions

<u>ACTIVATE</u>	Switches to a window
<u>DOCUMENTS</u>	Returns the names of the specified open workbooks
<u>GET.WINDOW</u>	Returns information about a window
<u>NEW.WINDOW</u>	Creates a new window for an existing sheet or macro sheet
<u>ON.WINDOW</u>	Runs a macro when you switch to a window

List of [Information Functions](#)

APP.ACTIVATE

Macro Sheets Only

Switches to an application. Use APP.ACTIVATE to switch to another application that is already running or that you have started by using EXEC.

Syntax

APP.ACTIVATE(title_text, wait_logical)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Title_text is the name of an application as displayed in its title bar.

- If title_text is omitted, APP.ACTIVATE switches to Microsoft Excel.
- If title_text is not a currently running application, APP.ACTIVATE returns the #VALUE! error value and interrupts the macro.
- Title_text is not necessarily the name of the application file. Use the text that appears in the title bar of the application, which might include the name of the open document and path information.
- In Microsoft Excel for the Macintosh, title_text can also refer to the Process Serial Number (PSN) that is returned by an EXEC function.

Wait_logical is a logical value determining when to switch to the application specified by title_text.

- If wait_logical is TRUE, Microsoft Excel waits to be switched to before switching to the application specified by title_text.
- If wait_logical is FALSE or omitted, Microsoft Excel immediately switches to the application specified by title_text.

Remarks

If you are running an application using Microsoft Excel macros, and you want to switch to a third application without switching to Microsoft Excel first, use FALSE as the wait_logical argument. With FALSE, you can use the application title_text without having to switch to Microsoft Excel first.

Examples

In Microsoft Excel for Windows, this macro formula switches to the Control Panel application as soon as you switch to Microsoft Excel:

```
APP.ACTIVATE("CONTROL PANEL", TRUE)
```

The following macro formula switches to Microsoft Word, which is currently displaying the document MONTHRPT.DOC in full screen mode:

```
APP.ACTIVATE("MICROSOFT WORD - MONTHRPT.DOC")
```

In Microsoft Excel for the Macintosh, the following macro formula switches to Microsoft Word:

```
APP.ACTIVATE("MICROSOFT WORD")
```

Tip Use an IF statement with APP.ACTIVATE to run an EXEC function if the application you want to switch to is not yet running.

Related Functions

The first five functions following are only for Microsoft Excel for Windows.

<u>APP.MAXIMIZE</u>	Maximizes the Microsoft Excel application window
<u>APP.MINIMIZE</u>	Minimizes the Microsoft Excel application window
<u>APP.MOVE</u>	Moves the Microsoft Excel application window
<u>APP.RESTORE</u>	Restores the Microsoft Excel application window
<u>APP.SIZE</u>	Changes the size of the Microsoft Excel application window
<u>EXEC</u>	Starts another application

List of DDE/External Functions

COPY

Macro Sheets Only

Equivalent to choosing the Copy command from the Edit menu. Copies and pastes data or objects.

Syntax

COPY(from_reference, to_reference)

From_reference is a reference to the cell or range of cells you want to copy. If from_reference is omitted, it is assumed to be the current selection.

To_reference is a reference to the cell or range of cells where you want to paste what you have copied.

- To_reference should be a single cell or an enlarged multiple of from_reference. For example, if from_reference is a 2 by 4 rectangle, to_reference can be a 4 by 8 rectangle.
- To_reference can be omitted so that you can subsequently paste using the PASTE, PASTE.LINK, or PASTE.SPECIAL functions.

Related Functions

CUT Cuts or moves data or objects

PASTE Pastes cut or copied data

PASTE.LINK Pastes copied data or objects and establishes a link to the source of the data or object

PASTE.SPECIAL Pastes specific components of copied data

List of Command-Equivalent Functions

DATA.DELETE

Macro Sheets Only

Equivalent to choosing the Delete command from the Data menu in Microsoft Excel version 4.0. Deletes data that matches the current criteria in the current database.

In the dialog-box form, DATA.DELETE?, Microsoft Excel displays a message warning you that matching records will be permanently deleted, and you can approve or cancel. In the plain form, DATA.DELETE, matching records are deleted without any message being displayed.

Syntax

DATA.DELETE()

DATA.DELETE?()

Related Functions

List of Command-Equivalent Functions

DATA.FIND

Macro Sheets Only

Equivalent to choosing the Find and Exit Find commands from the Data menu in Microsoft Excel version 4.0. Selects records in the database range which match criteria in the criteria range.

Syntax

DATA.FIND(logical)

Logical is a logical value that specifies whether to enter or exit the Data Find mode. If logical is TRUE, Microsoft Excel carries out the Find command; if FALSE, Microsoft Excel carries out the Exit Find command. If logical is omitted, the function toggles between Find and Exit Find.

Related Functions

DATA.FIND.NEXT Finds next matching record in a database

DATA.FIND.PREV Finds previous matching record in a database

List of Command-Equivalent Functions

DATA.FIND.NEXT

DATA.FIND.PREV

Macro Sheets Only

Equivalent to pressing the DOWN ARROW or UP ARROW key after the Find command has been chosen from the Data menu in Microsoft Excel version 4.0. Finds the next or previous matching record in a database. If the function cannot find a matching record, it returns the logical value FALSE.

Syntax

DATA.FIND.NEXT()

DATA.FIND.PREV()

Related Function

DATA.FIND Enters or exits Data Find mode

List of Command-Equivalent Functions

DATA.FORM

Macro Sheets Only

Equivalent to choosing the Form command from the Data menu. Displays the data form.

If Microsoft Excel cannot determine what database or list of information to use, the function returns the #VALUE! error value and interrupts the macro.

Syntax

DATA.FORM()

Remarks

- You can still use custom data forms created in Microsoft Excel version 4.0 or earlier. To edit the definition table of the custom data form, use the Dialog Editor from Microsoft Excel version 4.0.
- The data form can handle up to 32 fields.

Related Functions

List of [Command-Equivalent Functions](#)

DEFINE.STYLE

Equivalent to choosing the Define button in the Style dialog box, which appears when you choose the Style command from the Format menu. Creates and changes cell styles. There are seven syntax forms of this function. Use syntax 1 of DEFINE.STYLE to define styles based on the format of the active cell. To create a style by specifying number, font, and other formats, use syntaxes 2 through 7 of DEFINE.STYLE.

Syntax 1

Syntaxes 2-7

DEFINE.STYLE - Syntax 1

Macro Sheets Only

Equivalent to choosing the Define button in the Style dialog box, which appears when you choose the Style command from the Format menu. Creates and changes cell styles. There are seven syntax forms of this function. Use syntax 1 of DEFINE.STYLE to define styles based on the format of the active cell. To create a style by specifying number, font, and other formats, use syntaxes 2 through 7 of DEFINE.STYLE.

Syntax

DEFINE.STYLE(style_text, number, font, alignment, border, pattern, protection)

DEFINE.STYLE?(style_text, number, font, alignment, border, pattern, protection)

Style_text is the name, as text, that you want to assign to the style.

The following arguments are logical values corresponding to check boxes in the Style dialog box. If an argument is TRUE, Microsoft Excel selects the check box and uses the corresponding format of the active cell in the style; if FALSE, Microsoft Excel clears the check box and omits formatting descriptions for that attribute. If style_text is omitted and all selected cells have identical formatting, the default is TRUE; if cells have different formatting, the default is FALSE.

Number corresponds to the Number check box.

Font corresponds to the Font check box.

Alignment corresponds to the Alignment check box.

Border corresponds to the Border check box.

Pattern corresponds to the Pattern check box.

Protection corresponds to the Protection check box.

Related Functions

APPLY.STYLE Applies a style to the selection

DELETE.STYLE Deletes a cell style

MERGE.STYLES Imports styles from another workbook into the active workbook

Syntaxes 2-7

List of Command-Equivalent Functions

DEFINE.STYLE - Syntaxes 2 - 7

Macro Sheets Only

Equivalent to choosing the Define button in the Style dialog box, which appears when you choose the Style command from the Format menu. Creates and changes cell styles. Use one of the following syntax forms of DEFINE.STYLE to select cell formats for a new style or to alter the formats of an existing style. Use syntax 1 of DEFINE.STYLE to define styles based on the format of the active cell.

Syntax 2

Number format, using the arguments from the FORMAT.NUMBER function

DEFINE.STYLE(style_text, attribute_num, format_text)

Syntax 3

Font format, using the arguments from the FORMAT.FONT and FONT.PROPERTIES functions

DEFINE.STYLE(style_text, attribute_num, name_text, size_num, bold, italic, underline, strike, color, outline, shadow, superscript, subscript)

Syntax 4

Alignment, using the arguments from the ALIGNMENT function

DEFINE.STYLE(style_text, attribute_num, horiz_align, wrap, vert_align, orientation)

Syntax 5

Border, using the arguments from the BORDER function

DEFINE.STYLE(style_text, attribute_num, left, right, top, bottom, left_color, right_color, top_color, bottom_color)

Syntax 6

Pattern, using the arguments from the cell form of the PATTERNS function

DEFINE.STYLE(style_text, attribute_num, apattern, afore, aback)

Syntax 7

Cell protection, using the arguments from the CELL.PROTECTION function

DEFINE.STYLE(style_text, attribute_num, locked, hidden)

Style_text is the name, as text, that you want to assign to the style.

Attribute_num is a number from 2 to 7 that specifies which attribute of the style, such as its font, alignment, or number format, you want to designate with this function.

Attribute_num	Specifies
2	Number format
3	Font format
4	Alignment
5	Border
6	Pattern
7	Cell protection

Remarks

- The remaining arguments are different for each form and are identical to arguments in the corresponding function. For example, form 2 of DEFINE.STYLE defines the number format of a style and corresponds to the FORMAT.NUMBER function. The exception is form 5, which does not include every argument for BORDER. For details on the values you can use for these arguments, see the description under the corresponding function.
- If you define a style using one of these forms, then any attributes you don't explicitly define are not changed.

Related Functions

ALIGNMENT Aligns or wraps text in cells

<u>APPLY.STYLE</u>	Applies a style to the selection
<u>BORDER</u>	Adds a border to the selected cell or object
<u>CELL.PROTECTION</u>	Allows you to control cell protection and display
<u>DELETE.STYLE</u>	Deletes a cell style
<u>FONT.PROPERTIES</u>	Applies a font to the selection
<u>FORMAT.NUMBER</u>	Formats numbers, dates, and times in the selected cells
<u>MERGE.STYLES</u>	Imports styles from another workbook into the active workbook
<u>PATTERNS</u>	Changes the appearance of the selected object
<u>Syntax 1</u>	
List of <u>Command-Equivalent Functions</u>	

DELETE.NAME

Macro Sheets Only

Equivalent to choosing the Delete button in the Define Name dialog box, which appears when you choose the Define command from on the Name submenu on the Insert menu. Deletes the specified name.

Syntax

DELETE.NAME(name_text)

Name_text is a text value specifying the name that you want to delete.

Important Formulas that use names in their arguments may return incorrect or error values when a name used in the formula is deleted.

Related Functions

DEFINE.NAME Defines a name on the active worksheet or macro sheet

GET.NAME Returns the definition of a name

SET.NAME Defines a name as a value

List of Command-Equivalent Functions

DOCUMENTS

Macro Sheets Only

Returns, as a horizontal array in text form, the names of the specified open workbooks in alphabetic order. Use DOCUMENTS to retrieve the names of open workbooks to use in other functions that manipulate open workbooks.

Syntax

DOCUMENTS(type_num, match_text)

Type_num is a number specifying whether to include add-in workbooks in the array of workbooks, according to the following table.

Type_num	Returns
1 or omitted	Names of all open workbooks except add-in workbooks
2	Names of add-in workbooks only
3	Names of all open workbooks

Match_text specifies the workbooks whose names you want returned and can include wildcard characters. If match_text is omitted, DOCUMENTS returns the names of all open workbooks.

Remarks

- Use the INDEX function to select individual workbook names from the array to use in other functions that take workbook names as arguments.
- Use COLUMNS to count the number of entries in the horizontal array.
- Use TRANSPOSE to change a horizontal array to a vertical one.
- Since the DOCUMENTS function only returns actual workbook names, it ignores any changes made by the WINDOW.TITLE function.

Examples

In Microsoft Excel for Windows, if your workspace contains windows named BUDGET.XLS, CHART1, ACTUAL.XLS:1, ACTUAL.XLS:2, and BOOK.XLS, then:

DOCUMENTS(1) equals the four-cell array {"ACTUAL.XLS", "BOOK.XLS", "BUDGET.XLS", "CHART1"}

SET.NAME("Document_array", DOCUMENTS()) defines the name, Document_array, as {"ACTUAL.XLS", "BOOK.XLS", "BUDGET.XLS", "CHART1"}

In Microsoft Excel for the Macintosh, if your workspace contains windows named BUDGET CHART1, ACTUALS, ACTUALS:2, and BOOK then:

DOCUMENTS (1) equals the four-cell array {"ACTUALS", "BOOK", "BUDGET", "CHART1"}

Related Functions

FILES Returns the filenames in the specified directory or folder

GET.DOCUMENT Returns information about a workbook

GET.WINDOW Returns information about a window

WINDOWS Returns the names of all open windows

List of Information Functions

EDIT.OBJECT

Macro Sheets Only

Equivalent to choosing the Edit command from the (selected object) Object submenu on the Edit menu. Starts the application associated with the selected object and makes the object available for editing or other actions.

Syntax

EDIT.OBJECT(verb_num)

Verb_num is a number specifying which verb to use while working with the object, that is, what you want to do with the object.

- The available verbs are determined by the object's source application. 1 often specifies "edit, " and 2 often specifies "play" (for sound, animation, and so on). For more information, consult the documentation for the object's application to see how it supports object linking and embedding (OLE).
- If the object does not support multiple verbs, verb_num is ignored.
- If verb_num is omitted, it is assumed to be 1.

Remarks

Your macro pauses while you're editing the object and resumes when you return to Microsoft Excel.

Related Function

INSERT.OBJECT Creates an object of a specified type

List of Command-Equivalent Functions

EDITION.OPTIONS

Macro Sheets Only

Sets options in, or performs actions on, the specified publisher or subscriber. In Microsoft Excel for Windows, EDITION.OPTIONS also allows you to cancel a publisher or subscriber created in Microsoft Excel for the Macintosh.

Syntax

EDITION.OPTIONS(**edition_type**, edition_name, reference, **option**, appearance, size, formats)

Edition_type is the number 1 or 2 specifying the type of edition.

Edition_type	Type of edition
1	Publisher
2	Subscriber

Edition_name is the name of the edition you want to change the edition options for or to perform actions on. If edition_name is omitted, reference is required.

Reference specifies the range (given in text form as a name or an R1C1-style reference) occupied by the publisher or subscriber.

- Reference is required if you have more than one publisher or subscriber of edition_name on the active workbook. Use reference to specify the location of the publisher or subscriber for which you want to set options.
- If edition_type is 1 and the publisher is an embedded chart, or if edition_type is 2 and the subscriber is a picture, reference is the object identifier as displayed in the reference area.
- If reference is omitted, edition_name is required.

Option is a number from 1 to 6 specifying the edition option you want to set or the action you want to take, according to the following two tables. Options 2 to 6 are only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

If a publisher is specified, then option applies as follows.

Option	Action
1	Cancels the publisher
2	Sends the edition now
3	Selects the range or object published to the specified edition
4	Automatically updates the edition when the file is saved
5	Updates the edition on request only
6	Changes the edition file as specified by appearance, size, and formats

If a subscriber is specified, then option applies as follows.

Option	Action
1	Cancels the subscriber
2	Gets the latest edition
3	Opens the publisher workbook
4	Automatically updates when new data is available
5	Update on request only

The following three arguments are available only when option is 6.

Appearance specifies whether the selection is published as shown on screen or as shown when printed. The default value for appearance is 1 if the selection is a sheet or macro sheet and 2 if the selection is a chart.

Appearance	Selection is published
1	As shown on screen
2	As shown when printed

Size specifies the size of a published chart. Size is only available if a chart is to be published.

Size	Chart size is published
1 or omitted	As shown on screen
2	As shown when printed

Formats is a number specifying the format of the file.

Formats	File format
1 or omitted	PICT
2	BIFF
4	RTF
8	VALU

You can also use the sum of the allowable file formats. For example, a value of 6 specifies BIFF and RTF.

Example

The following macro formula opens the workbook (and application) that published the edition named Monthly Totals:

```
EDITION.OPTIONS(2, "Monthly Totals", , 3)
```

Related Functions

[CREATE.PUBLISHER](#)

Creates a publisher from the selection

[GET.LINK.INFO](#)

Returns information about a link

[SUBSCRIBE.TO](#)

Inserts contents of an edition into the active workbook

List of [Command-Equivalent Functions](#)

EMBED

Displayed in the formula bar when an embedded object is selected. EMBED cannot be entered on a sheet or used in a macro.

Syntax

EMBED(object_class, item)

Object_class is the name of the application and document type that created the embedded object. For example, the object_class arguments used when Microsoft Excel sheets are embedded in other applications are "Excel.sheet.5" and "Excel.Chart.5".

Item is the area selected to copy, and determines the view on the embedded document. When item is empty text (""), EMBED creates a view on the entire document.

Remarks

If you delete the EMBED formula, the embedded object remains on the document as a graphic, and the link to the creating application is deleted. Double-clicking the object no longer starts the creating application.

EXEC

Macro Sheets Only

Starts a separate program. Use EXEC to start other programs with which you want to communicate. Use EXEC with Microsoft Excel's other DDE functions (INITIATE, EXECUTE, and SEND.KEYS) to create a channel to another program and to send keystrokes and commands to the program. (SEND.KEYS is available only in Microsoft Excel for Windows.)

Syntax 1 is for Microsoft Excel for Windows. Syntax 2 is for Microsoft Excel for the Macintosh.

Syntax 1

For Microsoft Excel for Windows

EXEC(program_text, window_num)

Syntax 2

For Microsoft Excel for the Macintosh

EXEC(program_text, background, preferred_size_only)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for the last two arguments of this function.

Program_text is the name, as a text string, of any executable file or, in Microsoft Excel for Windows, any data file that is associated with an executable file.

- Use paths when the file or program to be started is not in the current directory or folder.
- In Microsoft Excel for Windows, program_text can include any arguments and switches that are accepted by the program to be started. Also, if program_text is the name of a file associated with a specific installed program, EXEC starts the program and loads the specified file.

Window_num is a number from 1 to 3 that specifies how the window containing the program should appear. Window_num is only available for use with Microsoft Excel for Windows. The window_num argument is allowed on the Macintosh, but it is ignored.

Window_num	Window appears
1	Normal size
2 or omitted	Minimized size
3	Maximized size

Background is a logical value that determines whether the program specified by program_text is opened as the active program or in the background, leaving Microsoft Excel as the active program. If background is TRUE, the program is started in the background; if FALSE or omitted, the program is started in the foreground. Background is only available for use with Microsoft Excel for the Macintosh and system software version 7.0 or later.

Preferred_size_only is a logical value that determines the amount of memory allocated to the program. If preferred_size_only is TRUE, the program is opened with its preferred memory allocation; if FALSE or omitted, it opens with the available memory if greater than its minimum requirement. Preferred_size_only is only available for use with Microsoft Excel for the Macintosh and system software version 7.0 or later. For information about changing the preferred memory size, see your Macintosh documentation.

Remarks

In Microsoft Excel for Windows and in Microsoft Excel for the Macintosh with system software version 7.0, if the EXEC function is successful, it returns the task ID number of the started program. The task ID number is a unique number that identifies a program. Use the task ID number in other macro functions, such as APP.ACTIVATE, to refer to the program. In Microsoft Excel for the Macintosh with system software version 6.0, if EXEC is successful, it returns TRUE. If EXEC is unsuccessful, it returns the #VALUE! error value.

Examples

In Microsoft Excel for Windows, the following macro formula starts the program SEARCH.EXE. Use paths when the file or program to be started is not in the current directory:

```
EXEC ("C:\WINDOWS\SEARCH.EXE")
```

The following macro formula starts Microsoft Word for Windows and loads the document SALES.DOC:

```
EXEC ("C:\WINWORD\WINWORD.EXE C:\MYFILES\SALES.DOC")
```

In Microsoft Excel for the Macintosh, the following macro formula starts Microsoft Word:

```
EXEC ("HARD DISK:APPS:WORD")
```

Related Functions

<u>APP.ACTIVATE</u>	Switches to another application
<u>EXECUTE</u>	Carries out a command in another application
<u>INITIATE</u>	Opens a channel to another application
<u>SEND.KEYS</u>	Sends a key sequence to an application
<u>TERMINATE</u>	Closes a channel to another application
<u>REQUEST</u>	Requests an array of a specific type of information from an application with which you have a dynamic data exchange (DDE) link
<u>POKE</u>	Sends data to another application with which you have a dynamic data exchange (DDE) link

List of DDE/External Functions

EXECUTE

Macro Sheets Only

Carries out commands in another program with which you have a dynamic data exchange (DDE) link. Use with EXEC, INITIATE, and SEND.KEYS to run another program through Microsoft Excel. (SEND.KEYS is available only in Microsoft Excel for Windows.)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Syntax

EXECUTE(channel_num, execute_text)

Channel_num is a number returned by a previously run INITIATE function. Channel_num refers to a channel through which Microsoft Excel communicates with another program.

Execute_text is a text string representing commands you want to carry out in the program specified by channel_num. The form of execute_text depends on the program you are referring to. To include specific key sequences in execute_text, use the format described under key_text in the ON.KEY function.

If EXECUTE is not successful, it returns one of the following error values:

Value returned	Situation
#VALUE!	Channel_num is not a valid channel number.
#N/A	The program you are accessing is busy.
#DIV/0!	The program you are accessing does not respond after a certain length of time or you have pressed ESC to cancel.
#REF!	The keys specified in execute_text are refused by the application which you want to access.

Remarks

Commands sent to another program with EXECUTE will not work when a dialog box is displayed in the program. In Microsoft Excel for Windows, you can use SEND.KEYS to send commands that make selections in a dialog box.

Examples

The following macro formula sends the number 25 and a carriage return to the application identified by channel_num 14:

```
EXECUTE(14, "25~")
```

Related Functions

<u>EXEC</u>	Starts another application
<u>INITIATE</u>	Opens a channel to another application
<u>POKE</u>	Sends data to another application
<u>REQUEST</u>	Returns data from another application
<u>SEND.KEYS</u>	Sends a key sequence to an application
<u>TERMINATE</u>	Closes a channel to another application
List of <u>DDE/External Functions</u>	

EXTRACT

Macro Sheets Only

Equivalent to choosing the Extract command from the Data menu in Microsoft Excel version 4.0. Finds database records that match the criteria defined in the criteria range and copies them into a separate extract range.

Syntax

EXTRACT(unique)

EXTRACT?(unique)

Unique is a logical value corresponding to the Unique Records Only check box in the Extract dialog box.

- If unique is TRUE, Microsoft Excel selects the check box and excludes duplicate records from the extract list.
- If unique is FALSE or omitted, Microsoft Excel clears the check box and extracts all records matching the criteria.

Related Functions

<u>DATA.FIND</u>	Finds records in a database
<u>SET.CRITERIA</u>	Defines the name Criteria for the selected range on the active sheet
<u>SET.DATABASE</u>	Defines the name Database for the selected range on the active sheet
<u>SET.EXTRACT</u>	Defines the name Extract for the selected range on the active sheet

List of Command-Equivalent Functions

FORMAT.NUMBER

Macro Sheets Only

Equivalent to choosing the Number tab in the Format Cells dialog box, which appears when you choose Cells from the Format menu. Formats numbers, dates, and times in the selected cells, data labels, and axis labels on charts. Use FORMAT.NUMBER to apply built-in formats or to create and apply custom formats.

Syntax

FORMAT.NUMBER(format_text)

FORMAT.NUMBER?(format_text)

Format_text is a format string, such as "#, ##0.00", specifying which format to apply to the selection.

Related Functions

DELETE.FORMAT Deletes the specified custom number format

FONT.PROPERTIES Applies a font to the selection

FORMAT.TEXT Formats a sheet text box or a chart text item

List of Command-Equivalent Functions

GET.OBJECT

Macro Sheets Only

Returns information about the specified object. Use GET.OBJECT to return information you can use in other macro formulas that manipulate objects.

Syntax

GET.OBJECT(type_num, object_id_text, start_num, count_num, item_index)

Type_num is a number specifying the type of information you want returned about an object.

GET.OBJECT returns the #VALUE! error value (and the macro is halted) if an object isn't specified or if more than one object is selected.

Type_num	Returns
1	Number specifying the type of the selected object: 0 = Grouped objects 1 = Line 2 = Rectangle 3 = Oval 4 = Arc 5 = Embedded chart 6 = Text box 7 = Button 8 = Picture 9 = Closed polygon 10 = Open polygon (freehand drawing) 11 = Check box 12 = Option button 13 = Edit box 14 = Label 15 = Dialog Frame 16 = Spinner 17 = Scroll bar 18 = List box 19 = Group box 20 = Drop down box
2	If the object is locked, returns TRUE; otherwise FALSE.
3	Z-order position (layering) of the object; that is, the relative position of the overlapping objects, starting with 1 for the object that is most under the others.
4	Reference of the cell under the upper-left corner of the object as text in R1C1 reference style; for a line or arc, returns the start point.
5	X offset from the upper-left corner of the cell under the upper-left corner of the object, measured in points.
6	Y offset from the upper-left corner of the cell under the upper-left corner of the object, measured in points.
7	Reference of the cell under the lower-right corner of the object as text in R1C1 reference style; for a line or arc, returns the end point.
8	X offset from the upper-left corner of the cell under the lower-right corner of the object, measured in points.
9	Y offset from the upper-left corner of the cell under the lower-right corner of the object, measured in points.
10	Name, including the filename, of the macro assigned to the object. If no macro is assigned, returns FALSE.
11	Number indicating how the object moves and sizes: 1 = Object moves and sizes with cells 2 = Object moves with cells 3 = Object is fixed

Values 12 to 21 for type_num apply only to text boxes and buttons. If another type of object is selected, GET.OBJECT returns the #VALUE! error value.

Type_num	Returns
12	Text starting at start_num for count_num characters.
13	Font name of all text starting at start_num for count_num characters. If the text contains more than one font name, returns the #N/A error value.
14	Font size of all text starting at start_num for count_num characters. If the text contains more than one font size, returns the #N/A error value.
15	If all text starting at start_num for count_num characters is bold, returns TRUE. If text contains only partial bold formatting, returns the #N/A error value.
16	If all text starting at start_num for count_num characters is italic, returns TRUE. If text contains only partial italic formatting, returns the #N/A error value.
17	If all text starting at start_num for count_num characters is underlined, returns TRUE. If text contains only partial underline formatting, returns the #N/A error value.
18	If all text starting at start_num for count_num characters is struck through, returns TRUE. If text contains only partial struck-through formatting, returns the #N/A error value.
19	In Microsoft Excel for the Macintosh, if all text starting at start_num for count_num characters is outlined, returns TRUE. If text contains only partial outline formatting, returns the #N/A error value. Always returns FALSE in Microsoft Excel for Windows.
20	In Microsoft Excel for the Macintosh, if all text starting at start_num for count_num characters is shadowed, returns TRUE. If text contains only partial shadow formatting, returns the #N/A error value. Always returns FALSE in Microsoft Excel for Windows.
21	Number from 0 to 56 indicating the color of all text starting at start_num for count_num characters; if color is automatic, returns 0. If more than one color is used, returns the #N/A error value.

Values 22 to 25 for type_num also apply only to text boxes and buttons. If another type of object is selected, GET.OBJECT returns the #N/A error value.

Type_num	Returns
22	Number indicating the horizontal alignment of text: 1 = Left 2 = Center 3 = Right 4 = Justified
23	Number indicating the vertical alignment of text: 1 = Top 2 = Center 3 = Bottom 4 = Justified
24	Number indicating the orientation of text: 0 = Horizontal 1 = Vertical 2 = Upward 3 = Downward
25	If button or text box is set to automatic sizing, returns TRUE; otherwise FALSE.

The following values for type_num apply to all objects, except where indicated.

Type_num	Returns
26	If the object is visible, returns TRUE; if the object has been hidden by the HIDE.OBJECT function, returns FALSE.
27	Number indicating the type of the border or line: 0 = Custom 1 = Automatic 2 = None
28	Number indicating the style of the border or line as shown in the Patterns tab in the Format Objects dialog box: 0 = None 1 = Solid line 2 = Dashed line 3 = Dotted line 4 = Dashed dotted line 5 = Dashed double-dotted line 6 = 50% gray line 7 = 75% gray line 8 = 25% gray line
29	Number from 0 to 56 indicating the color of the border or line; if the border is automatic, returns 0.
30	Number indicating the weight of the border or line: 1 = Hairline 2 = Thin 3 = Medium 4 = Thick
31	Number indicating the type of fill: 0 = Custom 1 = Automatic 2 = None
32	Number from 1 to 18 indicating the fill pattern as shown in the Format Object dialog box.
33	Number from 0 to 56 indicating the foreground color of the fill pattern; if the fill is automatic, returns 0. If the object is a line, returns the #N/A error value.
34	Number from 0 to 56 indicating the background color of the fill pattern; if the fill is automatic, returns 0. If the object is a line, returns the #N/A error value.
35	Number indicating the width of the arrowhead: 1 = Narrow 2 = Medium 3 = Wide If the object is not a line, returns the #N/A error value.
36	Number indicating the length of the arrowhead: 1 = Short 2 = Medium 3 = Long If the object is not a line, returns the #N/A error value.
37	Number indicating the style of the arrowhead: 1 = No head 2 = Open head 3 = Closed head 4 = Open double-ended head 5 = Closed double-ended head If the object is not a line, returns the #N/A error value.
38	If the border has round corners, returns TRUE; if the corners are

- square, returns FALSE. If the object is a line, returns the #N/A error value.
- 39 If the border has a shadow, returns TRUE; if the border has no shadow, returns FALSE. If the object is a line, returns the #N/A error value.
- 40 If the Lock Text check box in the Protection Tab of the Format Object dialog box is selected, returns TRUE; otherwise FALSE.
- 41 If objects are set to be printed, returns TRUE; otherwise FALSE.
- 42 The horizontal distance, measured in points, from the left edge of the active window to the left edge of the object. May be a negative number if the window is scrolled beyond the object.
- 43 The vertical distance, measured in points, from the top edge of the active window to the top edge of the object. May be a negative number if the window is scrolled beyond the object.
- 44 The horizontal distance, measured in points, from the left edge of the active window to the right edge of the object. May be a negative number if the window is scrolled beyond the object.
- 45 The vertical distance, measured in points, from the top edge of the active window to the bottom edge of the object. May be a negative number if the window is scrolled beyond the object.
- 46 The number of vertices in a polygon, or the #N/A error value if the object is not a polygon.
- 47 A count_num by 2 array of vertex coordinates starting at start_num in a polygon's array of vertices.
- 48 If the object is a text box, returns the cell reference that the text box is linked to. If the object is a control on a worksheet, returns the cell reference that the control's value is linked to. This information is returned as a string.
- 49 Returns the ID number of the object. For example, "Rectangle 5" returns 5. Note that the name of the object may not have this index in it if the object has been renamed by the user.
- 50 Returns the object's classname. For example, "Rectangle".
- 51 Returns the object name. By default, object names are the classname followed by the ID. For example, "Rectangle 1" is an object name, of which "Rectangle" is the classname, and 1 is the ID number. The object can also be renamed, in which case the name picked by the user is returned.
- 52 Returns the distance from cell A1 to the Left of the object bounding rectangle in points
- 53 Returns the distance from Cell A1 to the top of the object bounding rectangle in points
- 54 Returns the width of object bounding rectangle in points
- 55 Returns the height of object bounding rectangle in points
- 56 If the object is enabled, returns TRUE; otherwise, it returns FALSE.
- 57 Returns the shortcut key assignment for the control object, as text.
- 58 Returns TRUE is the button control on a dialog sheet is the default button of the dialog; otherwise, returns FALSE
- 59 Returns TRUE if the button control on the dialog sheet is clicked when the user presses the ESCAPE Key; otherwise, returns FALSE.
- 60 Returns TRUE if the button control on a dialog sheet will close the dialog box when pressed; otherwise, returns FALSE
- 61 Returns TRUE if the button control on a dialog sheet will be clicked when the user presses F1.

- 62 Returns the value of the control. For a check box or radio button, Returns 1 if it is selected, zero if it is not selected, or 2 if mixed. For a List box or dropdown box, returns the index number of the selected item, or zero if no item is selected. For a scroll bar, returns the numeric value of the scroll bar.
- 63 Returns the minimum value that a scroll bar or spinner button can have
- 64 Returns the maximum value that a scroll bar or spinner button can have
- 65 Returns the step increment value added or subtracted from the value of a scroll bar or spinner. This value is used when the arrow buttons are pressed on the control.
- 66 Returns the large, or "page" step increment value added or subtracted from the value of a scroll bar when it is clicked in the region between the thumb and the arrow buttons.
- 67 Returns the input type allowed in an edit box control:
1 = Text
2 = Integer
3 = Number (what type)
4 = Cell reference
5 = Formula
- 68 Returns TRUE if the edit box control allows multi-line editing with wrapped text; otherwise, it returns FALSE.
- 69 Returns TRUE if the edit box has a vertical scroll bar; otherwise, it returns FALSE.
- 70 Returns the object ID of the object that is linked to a list box or edit box. For a dropdown combo box that has an editable entry field, returns the object ID of itself. A dropdown box that can't be edited, returns FALSE.
- 71 Returns the number of entries in a List box, dropdown List box, or dropdown combo box.
- 72 Returns the text of the selected entry in a List box, dropdown List box, or dropdown combo box.
- 73 Returns the range used to fill the entries in a List box, dropdown List box, or dropdown combo box, as text. If an empty string is returned, then the control isn't filled from a range.
- 74 Returns the number of list lines displayed when a dropdown control is dropped.
- 75 Returns TRUE the object is displayed as 3-D; otherwise, it returns FALSE.
- 76 Returns the Far East phonetic accelerator key as text. Used for Far East versions of Microsoft Excel.
- 77 Returns the select status of the list box:
0 = single
1 = simple multi-select
2 = extended multi-select
- 78 Returns an array of TRUE and FALSE values indicating which items are selected in a list box. If TRUE, the item is selected; If FALSE, the item is not selected.
- 79 Returns TRUE if the add indent attribute is on for alignment. Returns FALSE if the add indent attribute is off for alignment. Used for only Far East versions of Microsoft Excel.

Object_id_text is the name and number, or number alone, of the object you want information about.

Object_id_text is the text displayed in the reference area when the object is selected. If object_id_text is omitted, it is assumed to be the selected object. If object_id_text is omitted and no object is selected, GET.OBJECT returns the #REF! error value and interrupts the macro.

Start_num is the number of the first character in the text box or button or the first vertex in a polygon you want information about. Start_num is ignored unless a text box, button, or polygon is specified by type_num and object_id_text. If start_num is omitted, it is assumed to be 1.

Count_num is the number of characters in a text box or button, or the number of vertices in a polygon, starting at start_num, that you want information about. Count_num is ignored unless a text box, button, or polygon is specified by type_num and object_id_text. If count_num is omitted, it is assumed to be 255.

Item_index is the index number or position of the item in the list box or drop-down box that you want information about, ranging from 1 to the number of items in the list box or drop-down box.

Tip Use GET.OBJECT(45) - GET.OBJECT(43) to determine the height of an object and GET.OBJECT(44) - GET.OBJECT(42) to determine the width.

Examples

The following macro formula returns the reference of the cell under the upper-left corner of the object Oval 3 (assume the cell is E2):

GET.OBJECT(4, "Oval 3") returns "R2C5"

The following macro formula changes the protection status of the object Rectangle 2 if it is locked:

IF(GET.OBJECT(2, "Rectangle 2"), OBJECT.PROTECTION(FALSE))

The following macro formula returns characters 25 through 185 from the object Text 5:

GET.OBJECT(12, "Text 5", 25, 160)

Related Functions

CREATE.OBJECT

Creates an object

FONT.PROPERTIES

Applies a font to the selection

OBJECT.PROTECTION

Controls how an object is protected

PLACEMENT

Determines an object's relationship to underlying cells

List of Information Functions

INITIATE

Macro Sheets Only

Opens a dynamic data exchange (DDE) channel to an application and returns the number of the open channel. Once you have opened a channel to another application with INITIATE, you can use EXECUTE and SEND.KEYS to control the other application from a Microsoft Excel macro. (SEND.KEYS is available only with Microsoft Excel for Windows.) If INITIATE is successful, it returns the number of the open channel. All the subsequent DDE macro functions use this number to specify the channel. If INITIATE is unsuccessful, FALSE is returned.

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Syntax

INITIATE(app_text, topic_text)

App_text is the DDE name of the application with which you want to begin a DDE session, in text form. The form of app_text depends on the application you are accessing. The DDE name of Microsoft Excel, for example, is "Excel".

Topic_text describes something, such as a document or a record in a database, in the application that you are accessing; the form of topic_text depends on the application you are accessing. Microsoft Excel accepts the names of the current documents as topic_text, as well as the name "System".

Remarks

- You can specify an instance of an application by appending the application's task ID number to the app_text argument. If you start an application by using the EXEC function, EXEC returns the task ID number for that instance of the application.
- If more than one instance of an application is running and you do not specify which instance you would like to open a channel to, INITIATE displays a dialog box from which you can choose the instance you want. You can prevent this dialog box from appearing by disabling or redirecting errors with the ERROR function.

Example

The following macro formula opens a channel to the document named MEMO in the application named WORD:

```
INITIATE ("WORD", "MEMO")
```

Related Functions

<u>POKE</u>	Sends data to another application
<u>REQUEST</u>	Returns data from another application
<u>TERMINATE</u>	Closes a channel to another application
<u>EXECUTE</u>	Carries out a command in another application
<u>EXEC</u>	Starts a separate program

List of DDE/External Functions

INSERT.OBJECT

Macro Sheets Only

Equivalent to choosing the Object command from the Insert menu, and then selecting an object type and choosing the OK button. Creates an embedded object whose source data is supplied by another application. Also starts an application of the appropriate class for the specified object type.

Syntax

INSERT.OBJECT(object_class, file_name, link_logical, display_icon_logical, icon_file, icon_number, icon_label)

INSERT.OBJECT?(object_class, file_name, link_logical, display_icon_logical, icon_file, icon_number, icon_label)

Object_class is a text string containing the classname for the object you want to create.

- Object_class is the classname corresponding to the Object Type selection in the Insert Object dialog box.
- For more information about object classnames, consult the documentation for your source application to see how it supports object linking and embedding (OLE).

File_name is a text string specifying the file from which to create an OLE object.

Link_logical is a logical value indicating whether the new object based on file_name should be linked to file_name. If it is not linked, the object is created as a copy of the file. Link_logical is ignored if file_name is not specified. If link_logical is FALSE or omitted, then no link is established.

Display_icon_logical is a logical value corresponding to the Display as Icon checkbox. If it is FALSE or omitted, then the regular picture for the object is displayed. If it is TRUE, then the icon icon_number found in icon_file is displayed with the label icon_label. If display_icon_logical is not TRUE, then icon_file, icon_number, and icon_label are ignored.

Icon_file is the name of the file where the icon to display is located.

Icon_number is the number of the icon within icon_file that should be used.

Icon_label is a text string indicating a label to display beneath the icon. If the parameter is an empty string ("") or is omitted, no label is displayed.

Remarks

- If INSERT.OBJECT starts another application, your macro pauses. Your macro resumes when you return to Microsoft Excel.
- Although you will not normally use Microsoft Excel classnames in a Microsoft Excel macro, you may need them in macros written for other applications. Microsoft Excel uses classnames "Excel.Sheet.5" and "Excel.Chart.5".

Related Function

EDIT.OBJECT Edits an object

List of Command-Equivalent Functions

ON.DATA

Macro Sheets Only

Runs a specified macro when another application sends data to a particular workbook via dynamic data exchange (DDE) or via Publish and Subscribe on the Macintosh. Links to workbooks in other applications are called remote references.

Syntax

ON.DATA(document_text, macro_text)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Document_text is the name of the sheet to which remote data will be sent or the name of the source of the remote data.

- If document_text is the name of the remote data source, it must be in the form app|topic!item. You can use the form app|topic to include all items for a particular topic, or app| to specify an app alone, but you must include the | to indicate that you are specifying the name of a data source.
- If document_text is omitted, the macro specified by macro_text is run whenever remote data is sent to any sheet not already assigned to another ON.DATA function.
- In Microsoft Excel for the Macintosh, document_text can also be the name of a published edition file. Unless the file is in the current folder, document_text must include the complete path.

Macro_text is the name of, or an R1C1-style reference to, a macro that you want to run when data comes into the workbook or from the source specified by document_text. The name or reference must be in text form.

- If macro_text is omitted, the ON.DATA function is turned off for the specified workbook or source.

Remarks

- ON.DATA remains in effect until you either clear it or quit Microsoft Excel. You can clear ON.DATA by specifying document_text and omitting the macro_text argument.
- If the macro sheet containing macro_text is closed when data is sent to document_text, an error is returned.
- If the incoming data causes recalculation, Microsoft Excel first runs the macro macro_text and then performs the recalculation.

Examples

In Microsoft Excel for Windows, the following macro formula runs the macro AddOrders when data is sent to the sheet New in the workbook ORDERSDB.XLS:

```
ON.DATA("[ORDERSDB.XLS]New", "AddOrders")
```

In Microsoft Excel for the Macintosh, the following macro formula runs the macro beginning at cell R2C3 when data is sent to the sheet North in the workbook SALES DATABASE:

```
ON.DATA("[SALES DATABASE]North", "R2C3")
```

Related Functions

<u>ERROR</u>	Specifies what action to take if an error is encountered while a macro is running
<u>INITIATE</u>	Opens a channel to another application
<u>ON.ENTRY</u>	Runs a macro when data is entered
<u>ON.RECALC</u>	Runs a macro when a workbook is recalculated
List of <u>Customizing Functions</u>	

ON.DOUBLECLICK

Macro Sheets Only

Runs a macro when you double-click any cell or object on the specified sheet or macro sheet or double-click any item on the specified chart.

Syntax

ON.DOUBLECLICK(sheet_text, macro_text)

Sheet_text is a text value specifying the name of a sheet in a workbook. If sheet_text is omitted, the macro is run whenever you double-click any sheet not specified by a previous ON.DOUBLECLICK formula. Sheet_text must be in the form "[book1]sheet1".

Macro_text is the name of, or an R1C1-style reference to, a macro you want to run when you double-click the sheet specified by sheet_text. The name or reference must be in text form. If macro_text is omitted, double-clicking reverts to its normal behavior, and any macros assigned by previous ON.DOUBLECLICK functions are turned off.

Remarks

- ON.DOUBLECLICK overrides Microsoft Excel's normal double-click behavior, such as displaying cell notes, displaying the Patterns dialog box, or allowing editing cell text directly in cells.
- To determine what cell, object, or chart item has been double-clicked, use a CALLER function in the macro specified by macro_text.
- ON.DOUBLECLICK does not affect objects to which ASSIGN.TO.OBJECT macros have already been assigned. Use ON.DOUBLECLICK (TRUE) to make Microsoft Excel carry out the action that would normally occur if you double-click on the current selection.

Related Functions

[ASSIGN.TO.OBJECT](#)

Assigns a macro to an object

[ON.WINDOW](#)

Runs a macro when you switch to a window

List of [Customizing Functions](#)

ON.ENTRY

Macro Sheets Only

Runs a macro when you enter data into any cell on the specified sheet.

Syntax

ON.ENTRY(sheet_text, macro_text)

Sheet_text is a text value specifying the name of a sheet in a workbook. If sheet_text is omitted, the macro is run whenever you enter data into any sheet or macro sheet.

Macro_text is the name of, or an R1C1-style reference to, a macro you want to run when you enter data into the sheet specified by sheet_text. The name or reference must be in text form. If macro_text is omitted, entering data reverts to its normal behavior, and any macros assigned by previous ON.ENTRY functions are turned off.

Remarks

- The macro is run only when you enter data using the formula bar, not when you use edit commands or macro functions.
- To determine what cell had data entered into it, use a CALLER function in the macro specified by macro_text.

Related Functions

ENTER.DATA Turns Data Entry mode on or off

ON.RECALC Runs a macro when a workbook is recalculated

List of Customizing Functions

ON.KEY

Macro Sheets Only

Runs a specified macro when a particular key or key combination is pressed.

Syntax

ON.KEY(key_text, macro_text)

Key_text can specify any single key, or any key combined with ALT, CTRL, or SHIFT, or any combination of those keys (in Microsoft Excel for Windows) or COMMAND, CTRL, OPTION, or SHIFT or any combination of those keys (in Microsoft Excel for the Macintosh). Each key is represented by one or more characters, such as "a" for the character a, or "{ENTER}" for the ENTER key.

To specify characters that aren't displayed when you press the key, such as ENTER or TAB, use the codes shown in the following table. Each code in the table represents one key on the keyboard.

Key	Code
BACKSPACE	"{BACKSPACE}" or "{BS}"
BREAK	"{BREAK}"
CAPS LOCK	"{CAPSLOCK}"
CLEAR	"{CLEAR}"
DELETE or DEL	"{DELETE}" or "{DEL}"
DOWN	"{DOWN}"
END	"{END}"
ENTER (numeric keypad)	"{ENTER}"
ENTER	"~" (tilde)
ESC	"{ESCAPE}" or "{ESC}"
HELP	"{HELP}"
HOME	"{HOME}"
INS	"{INSERT}"
LEFT	"{LEFT}"
NUM LOCK	"{NUMLOCK}"
PAGE DOWN	"{PGDN}"
PAGE UP	"{PGUP}"
RETURN	"{RETURN}"
RIGHT	"{RIGHT}"
SCROLL LOCK	"{SCROLLLOCK}"
TAB	"{TAB}"
UP	"{UP}"
F1 through F15	"{F1}" through "{F15}"

In Microsoft Excel for Windows, you can also specify keys combined with SHIFT and/or CTRL and/or ALT. In Microsoft Excel for the Macintosh, you can also specify keys combined with SHIFT and/or CTRL and/or OPTION and/or COMMAND. To specify a key combined with another key or keys, use the following table.

To combine with	Precede the key code by
SHIFT	"+" (plus sign)
CTRL	"^" (caret)
ALT or OPTION	"%" (percent sign)
COMMAND	"*" (asterisk)

To assign a macro to one of the special characters (+, ^, %, and so on), enclose the character in brackets. For example, ON.KEY("^{+}", "InsertItem") assigns a macro named InsertItem to the key sequence CTRL+PLUS SIGN.

Macro_text is the name of a macro that you want to run when key_text is pressed. The reference

must be in text form.

- If macro_text is "" (empty text), nothing happens when key_text is pressed. This form of ON.KEY disables the normal meaning of keystrokes in Microsoft Excel.
- If macro_text is omitted, key_text reverts to its normal meaning in Microsoft Excel, and any special key assignments made with previous ON.KEY functions are cleared.

Remarks

- ON.KEY remains in effect until you clear it or quit Microsoft Excel. You can clear ON.KEY by specifying key_text and omitting the macro_text argument.
- If the macro sheet containing macro_text is closed when you press key_text, an error is returned.
- If another macro is running when you press key_text, macro_text will not run.

Examples

Suppose you wanted the key combination SHIFT+CTRL+RIGHT to run the macro Print. You use the following macro formula:

```
ON.KEY ("+^{RIGHT}", "Print")
```

To return SHIFT+CTRL+RIGHT to its normal meaning, you would use the following macro formula:

```
ON.KEY ("+^{RIGHT}")
```

To disable SHIFT+CTRL+RIGHT altogether, you would use the following macro formula:

```
ON.KEY ("+^{RIGHT}", "")
```

Related Functions

<u>CANCEL.KEY</u>	Disables macro interruption
<u>ERROR</u>	Specifies what action to take if an error is encountered while a macro is running
<u>SEND.KEYS</u>	Sends a key sequence to an application
List of <u>Customizing Functions</u>	

ON.RECALC

Macro Sheets Only

Runs a macro when a specific sheet is recalculated. Use ON.RECALC to perform an operation on a sheet each time the sheet is recalculated, such as checking that a certain condition is still being met.

Syntax

ON.RECALC(sheet_text, macro_text)

Sheet_text is the name of a sheet, given as text. If sheet_text is omitted, the macro is run whenever any open sheet not specified by a previous ON.RECALC formula is recalculated. Only one ON.RECALC formula can be run for each recalculation.

Macro_text is the name of, or an R1C1-style reference to, a macro you want to run when the sheet specified by sheet_text is recalculated. The name or reference must be in text form. The macro will be run each time the sheet is recalculated until ON.RECALC is cleared. If macro_text is omitted, recalculating reverts to its normal behavior, and any macros assigned by previous ON.RECALC functions are turned off.

Remarks

A macro specified to be run by ON.RECALC is not run by actions taken by other macros. For example, a macro specified by ON.RECALC will not be run after the CALCULATE.NOW function is carried out, but will be run if you change data in a sheet set to calculate automatically or choose the Calc Now button in the Calculation panel of the Options dialog box, which appears when you choose the Options command from the Tools menu.

Examples

In Microsoft Excel for Windows, the following macro formula specifies that the macro Printer on the macro sheet AUTOREPT.XLS be run when the sheet named REPORT.XLS is recalculated:

```
ON.RECALC ("REPORT.XLS", "AUTOREPT.XLS!Printer")
```

In Microsoft Excel for the Macintosh, the following macro formula turns off ON.RECALC for the workbook named SALES:

```
ON.RECALC ("SALES")
```

Related Functions

[CALCULATE.DOCUMENT](#)

Calculates the active document only

[CALCULATE.NOW](#)

Calculates all open documents immediately

[CALCULATION](#)

Controls calculation settings

[ON.ENTRY](#)

Runs a macro when data is entered

List of [Customizing Functions](#)

ON.TIME

Macro Sheets Only

Runs a macro at a specified time. Use ON.TIME to run a macro at a specific time of day or after a specified period has passed.

Syntax

ON.TIME(time, macro_text, tolerance, insert_logical)

Time is the time and date, given as a serial number, at which the macro is to be run. If time does not include a date (that is, if time is a serial number less than 1), the macro is run the next time time occurs.

Macro_text is the name of, or an R1C1-style reference to, a macro to run at the specified time and every subsequent day at that time.

Tolerance is the time and date, given as a serial number, that you are willing to wait until and still have the macro run. For example, if Microsoft Excel is not in Ready, Copy, Cut, or Find mode at time, because another macro is running, but is in Ready mode 15 seconds later, and tolerance is set to time plus 30 seconds, the macro specified by macro_text will run. If Microsoft Excel was not in Ready mode within 30 seconds, the macro would not run. If tolerance is omitted, it is assumed to be infinite.

Insert_logical is a logical value specifying whether you want every day macro_text to run at time. Use insert_logical when you want to clear a previously set ON.TIME formula. If insert_logical is TRUE or omitted, the macro specified by macro_text is carried out at time. If insert_logical is FALSE and macro_text is not set to run at time, ON.TIME returns the #VALUE error value.

Examples

The following macro formula runs a macro called Test at 5:00:00 P.M. every day when Microsoft Excel is in Ready mode:

```
ON.TIME("5:00:00 PM", "Test")
```

The following macro formula runs a macro called Test 5 seconds after the formula is evaluated:

```
ON.TIME(NOW()+"00:00:05", "Test")
```

The following macro formula runs a macro called Test 10 seconds after the formula is evaluated. If Microsoft Excel is not in Ready mode at that time (because it is in Edit mode, for example), the tolerance argument specifies 5 seconds of additional time to wait to run the macro. If Microsoft Excel is still not in Ready mode at that time, macro_text is not run.

```
ON.TIME(NOW()+"00:00:10", "Test", NOW()+"00:00:15")
```

Related Functions

[NOW](#) Returns the serial number of the current date and time

List of [Customizing Functions](#)

ON.WINDOW

Macro Sheets Only

Runs a specified macro when you switch to a particular window.

Syntax

ON.WINDOW(window_text, macro_text)

Window_text is the name of a window in the form of text. If window_text is omitted, ON.WINDOW starts the macro whenever you switch to any window, except for windows that are named in other ON.WINDOW statements.

Macro_text is the name of, or an R1C1-style reference to, a macro to run when you switch to window_text. If macro_text is omitted, switching to window_text no longer runs the previously specified macro.

Remarks

- A macro specified to be run by ON.WINDOW is not run when other macros switch to the window or when a command to switch to a window is received through a DDE channel. Instead, ON.WINDOW responds to a user's actions, such as clicking a window with the mouse, choosing the Copy command from the Edit menu, and so on.
- If a sheet or macro sheet has an Auto_Activate or Auto_Deactivate macro defined for it, those macros will be run after the macro specified by ON.WINDOW.

Examples

In Microsoft Excel for Windows, the following macro formula runs the macro beginning at cell R1C2 when you switch to the window MAIN.XLS:

```
ON.WINDOW("MAIN.XLS", "R1C2")
```

The following macro formula stops the macro from running when you switch to MAIN.XLS:

```
ON.WINDOW("MAIN.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula runs the macro named ShowAlert when you switch to the window MAIN WINDOW:

```
ON.WINDOW("MAIN WINDOW", "ShowAlert")
```

The following macro formula stops the macro from running when you switch to MAIN WINDOW:

```
ON.WINDOW("MAIN WINDOW")
```

Related Functions

<u>GET.WINDOW</u>	Returns information about a window
<u>ON.KEY</u>	Runs a macro when a specified key is pressed
<u>ON.SHEET</u>	Triggers a macro whenever the specified sheet is activated from another sheet
<u>WINDOWS</u>	Returns the names of all open windows
List of <u>Customizing Functions</u>	

OPEN

Macro Sheets Only

Equivalent to choosing the Open command from the File menu. Opens an existing workbook.

Syntax

OPEN(file_text, update_links, read_only, format, prot_pwd, write_res_pwd, ignore_rorec, file_origin, custom_delimit, add_logical, editable, file_access, notify_logical, converter)

OPEN?(file_text, update_links, read_only, format, prot_pwd, write_res_pwd, ignore_rorec, file_origin, custom_delimit, add_logical, editable, file_access, notify_logical, converter)

File_text is the name, as text, of the workbook you want to open. File_text can include a drive and path, and can be a network pathname. In the dialog-box form in Microsoft Excel for Windows, file_text can include an asterisk (*) to represent any sequence of characters and a question mark (?) to represent any single character.

Update_links specifies whether and how to update external and remote references. If update_links is omitted, Microsoft Excel displays a message asking if you want to update links.

If update_links is	Then Microsoft Excel
--------------------	----------------------

0	Updates neither external nor remote references
1	Updates external references only
2	Updates remote references only
3	Updates external and remote references

Note When you are opening a file in WKS, WK1, or WK3 format, the update_links argument specifies whether Microsoft Excel generates charts from any graphs attached to the WKS, WK1, or WK3 file.

If update_links is	Charts are
--------------------	------------

0	Not created
2	Created

Read_only corresponds to the Read Only check box in the Open dialog box. If read_only is TRUE, the workbook can be modified but changes cannot be saved; if FALSE or omitted, changes to the workbook can be saved.

Format specifies what character to use as a delimiter when opening text files. If format is omitted, Microsoft Excel uses the current delimiter setting.

If format is	Values are separated by
--------------	-------------------------

1	Tabs
2	Commas
3	Spaces
4	Semicolons
5	Nothing
6	Custom characters

Prot_pwd is the password, as text, required to unprotect a protected file. If prot_pwd is omitted and file_text requires a password, the Password dialog box is displayed. Passwords are case-sensitive. Passwords are not recorded when you open a workbook and supply the password with the macro recorder on.

Write_res_pwd is the password, as text, required to open a read-only file with full write privileges. If write_res_pwd is omitted and file_text requires a password, the Password dialog box is displayed.

Ignore_rorec is a logical value that controls whether the read-only recommended message is displayed. If ignore_rorec is TRUE, Microsoft Excel prevents display of the message; if FALSE or omitted, and if read_only is also FALSE or omitted, Microsoft Excel displays the alert when opening a read-only recommended workbook.

File_origin is a number specifying whether a text file originated on the Macintosh or in Windows.

File_origin	Original operating environment
-------------	--------------------------------

1	Macintosh
---	-----------

2	Windows (ANSI)
3	MS-DOS (PC-8)
Omitted	Current operating environment

Custom_delimit is the character you want to use as a custom delimiter when opening text files.

- Custom_delimit is text or a reference or formula that returns text, such as CHAR(124).
- Custom_delimit is required if format is 6; it is ignored if format is not 6.
- Only the first character in custom_delimit is used.

Add_logical is a logical value that specifies whether or not to add file_text to the open workbook. If add_logical is TRUE, the document is added; if FALSE or omitted, it is not added. This argument is for compatibility with workbooks from Microsoft Excel version 4.0.

Editable is a logical value that corresponds to opening a file (such as a template) while holding down SHIFT key. If TRUE, editable is the equivalent to holding down the SHIFT key while choosing the OK button in the Open dialog box. If FALSE or omitted, this argument is ignored.

File_access is a number specifying how the file is to be accessed. If the file is being opened for the first time, this argument is ignored. If the file is already opened, this argument determines how to change the user's access permissions for the file.

File Access How Accessed

1	Revert to saved copy
2	Change to read/write access
3	Change to read only access

Notify_logical is a logical value that specifies whether the user should be notified when the shared document is available to be opened across a network. If TRUE, the user will be notified when the document is available to be opened. If FALSE or omitted, the user will not be notified when the file available to be opened.

Converter is a number corresponding to the file converter to use to open the file. Normally, Microsoft Excel automatically determines which file converter to use; therefore, this argument can usually be excluded. If you want to be certain, however, that a specific manually installed converter be used, then include this argument. Use GET.WORKSPACE(62) to determine which numbers corresponds to all of the installed converters.

Related Functions

<u>CLOSE</u>	Closes the active window
<u>FCLOSE</u>	Closes a text file
<u>FOPEN</u>	Opens a file with the type of permission specified
<u>OPEN.LINKS</u>	Opens specified supporting workbooks

List of Command-Equivalent Functions

PAGE.SETUP

Macro Sheets Only

Equivalent to choosing the Page Setup command from the File menu. Use PAGE.SETUP to control the printed appearance of your sheets.

There are three syntax forms of PAGE.SETUP. Syntax 1 applies if a sheet or macro sheet is active; syntax 2 applies if a chart is active; syntax three applies to Visual Basic modules and the info Window.

Arguments correspond to check boxes and text boxes in the Page Setup dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. Arguments for margins are always in inches, regardless of your country setting.

Syntax 1

Worksheets and macro sheets

PAGE.SETUP(head, foot, left, right, top, bot, hdng, grid, h_cntr, v_cntr, orient, paper_size, scale, pg_num, pg_order, bw_cells, quality, head_margin, foot_margin, notes, draft)

PAGE.SETUP?(head, foot, left, right, top, bot, hdng, grid, h_cntr, v_cntr, orient, paper_size, scale, pg_num, pg_order, bw_cells, quality, head_margin, foot_margin, notes, draft)

Syntax 2

Charts

PAGE.SETUP(head, foot, left, right, top, bot, size, h_cntr, v_cntr, orient, paper_size, scale, pg_num, bw_chart, quality, head_margin, foot_margin, draft)

PAGE.SETUP?(head, foot, left, right, top, bot, size, h_cntr, v_cntr, orient, paper_size, scale, pg_num, bw_chart, quality, head_margin, foot_margin, draft)

Syntax 3

Visual Basic Modules and the Info Window

PAGE.SETUP(head, foot, left, right, top, bot, orient, paper_size, scale, quality, head_margin, foot_margin, pg_num)

PAGE.SETUP?(head, foot, left, right, top, bot, orient, paper_size, scale, quality, head_margin, foot_margin, pg_num)

Head specifies the text and formatting codes for the header for the current sheet. For information about formatting codes, see "Remarks" later in this topic.

Foot specifies the text and formatting codes for the workbook footer.

Left corresponds to the Left box and is a number specifying the left margin.

Right corresponds to the Right box and is a number specifying the right margin.

Top corresponds to the Top box and is a number specifying the top margin.

Bot corresponds to the Bottom box and is a number specifying the bottom margin.

Hdng corresponds to the Row & Column Headings check box. Hdng is available only in the sheet and macro sheet form of the function.

Grid corresponds to the Cell Gridlines check box. Grid is available only in the sheet and macro sheet form of the function.

H_cntr corresponds to the Center Horizontally check box in the Margins panel of the Page Setup dialog box.

V_cntr corresponds to the Center Vertically check box in the Margins panel of the Page Setup dialog box.

Orient determines the direction in which your workbook is printed.

Orient	Print format
---------------	---------------------

1	Portrait
---	----------

2	Landscape
---	-----------

Paper_size is a number from 1 to 26 that specifies the size of the paper.

Paper_size	Paper type
-------------------	-------------------

1	Letter
2	Letter (small)
3	Tabloid
4	Ledger
5	Legal
6	Statement
7	Executive
8	A3
9	A4
10	A4 (small)
11	A5
12	B4
13	B5
14	Folio
15	Quarto
16	10x14
17	11x17
18	Note
19	ENV9
20	ENV10
21	ENV11
22	ENV12
23	ENV14
24	C Sheet
25	D Sheet
26	E Sheet

Scale is a number representing the percentage to increase or decrease the size of the sheet. All scaling retains the aspect ratio of the original.

- To specify a percentage of reduction or enlargement, set scale to the percentage.
- For worksheets and macros, you can specify the number of pages that the printout should be scaled to fit. Set scale to a two-item horizontal array, with the first item equal to the width and the second item equal to the height. If no constraint is necessary in one direction, you can set the corresponding value to #N/A.
- Scale can also be a logical value. To fit the print area on a single page, set scale to TRUE.

Pg_num specifies the number of the first page. If zero, sets first page to zero. If "Auto" is used, then the page numbering is set to automatic. If omitted, PAGE.SETUP retains the existing pg_num.

Pg_order specifies whether pagination is left-to-right and then down, or top-to-bottom and then right.

Pg_order	Pagination
1	Top-to-bottom, then right
2	Left-to-right, then down

Bw_cells is a logical value that specifies whether to print cells and all graphic objects, such as text boxes and buttons, in color.

- If bw_cells is TRUE, Microsoft Excel prints cell text and borders in black and cell backgrounds in white.
- If bw_cells is FALSE, Microsoft Excel prints cell text, borders, and background patterns in color (or in gray scale).

Bw_chart is a logical value that specifies whether to print chart in color.

Size is a number corresponding to the options in the Chart Size box, and determines how you want the chart printed on the page within the margins. Size is available only in the chart form of the function.

Size	Size to print the chart
1	Screen size
2	Fit to page
3	Full page

Quality specifies the print quality in dots-per-inch. To specify both horizontal and vertical print quality, use an array of two values.

Head_margin is the placement, in inches, of the running head margin from the edge of the page.

Foot_margin is the placement, in inches, of the running foot margin from the edge of the page.

Draft corresponds to the Draft Quality checkbox in the Sheet tab and in the Chart tab of the Page Setup dialog box. If FALSE or omitted, graphics are printed with the sheet. If TRUE, no graphics are printed.

Notes specifies whether to print cell notes with the sheet. If TRUE, both the sheet and the cell notes are printed. If FALSE or omitted, just the sheet is printed.

Remarks

Microsoft Excel no longer requires you to enter formatting codes to format headers and footers, but the codes are still supported and recorded by the macro recorder. You can include these codes as part of the head and foot text strings to align portions of the header or footer to the left, right, or center; to include the page number, date, time, or workbook name; and to print the header or footer in bold or italic.

Formatting code	Result
&L	Left-aligns the characters that follow.
&C	Centers the characters that follow.
&R	Right-aligns the characters that follow.
&B	Turns bold printing on or off (now obsolete).
&I	Turns italic printing on or off.
&U	Turns single underlining printing on or off.
&S	Turns strikethrough printing on or off.
&O	Turns outline printing on or off (Macintosh only).
&H	Turns shadow printing on or off (Macintosh only).
&D	Prints the current date.
&T	Prints the current time.
&A	Prints the name of the sheet
&F	Prints the name of the workbook.
&P	Prints the page number.
&P+number	Prints the page number plus number.
&P-number	Prints the page number minus number.
&&	Prints a single ampersand.
& "fontname, fontstyle"	Prints the characters that follow in the specified font and style. Be sure to include a comma immediately following the fontname, and double quotation marks around fontname and fontstyle.
&nn	Prints the characters that follow in the specified font size. Use a two-digit number to specify a size in points.
&N	Prints the total number of pages in the workbook.
&E	Prints a double underline
&X	Prints the character as superscript
&Y	Prints the character as subscript

Related Functions

<u>DISPLAY</u>	Controls screen and Info Window display
<u>GET.DOCUMENT</u>	Returns information about a workbook

PRINT Prints the active workbook
WORKSPACE Changes workspace settings
List of Command-Equivalent Functions

PATTERNS

Macro Sheets Only

Equivalent to choosing the Patterns tab in the Format Cells dialog box, which appears when you choose the Cells command from the Format menu. Changes the appearance of the selected cells or objects or the selected chart item (you can select only one chart item at a time). The PATTERNS function has eight syntax forms: syntax 1 is for cells on a sheet or macro sheet. Syntax 2 is for lines or arrows on a worksheet, macro sheet, or chart. Syntax 3 is for objects on a sheet or macro sheet. Syntax 4 through syntax 8 are for chart items.

Syntax 1

Cells

PATTERNS(apattern, afore, aback, newui)

PATTERNS?(apattern, afore, aback, newui)

Syntax 2

Lines (arrows) on worksheets or charts

PATTERNS(lauto, lstyle, lcolor, lwt, hwidth, hlength, htype)

PATTERNS?(lauto, lstyle, lcolor, lwt, hwidth, hlength, htype)

Syntax 3

Text boxes, rectangles, ovals, arcs, and pictures on worksheets or macro sheets

PATTERNS(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, rounded, newui)

PATTERNS?(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, rounded, newui)

Syntax 4

Chart plot areas, bars, columns, pie slices, and text labels

PATTERNS(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, invert, apply, new_fill)

PATTERNS?(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, invert, apply, new_fill)

Syntax 5

Chart axes

PATTERNS(lauto, lstyle, lcolor, lwt, tmajor, tminor, tlabel)

PATTERNS?(lauto, lstyle, lcolor, lwt, tmajor, tminor, tlabel)

Syntax 6

Chart gridlines, hi-lo lines, drop lines, lines on a picture line chart, and picture charts of bar and column charts

PATTERNS(lauto, lstyle, lcolor, lwt, apply, smooth)

PATTERNS?(lauto, lstyle, lcolor, lwt, apply, smooth)

Syntax 7

Chart data lines

PATTERNS(lauto, lstyle, lcolor, lwt, mauto, mstyle, mfore, mback, apply, smooth)

PATTERNS?(lauto, lstyle, lcolor, lwt, mauto, mstyle, mfore, mback, apply, smooth)

Syntax 8

Picture chart markers

PATTERNS(type, picture_units, apply)

PATTERNS?(type, picture_units, apply)

The following argument descriptions are in alphabetic order. Arguments correspond to check boxes, list boxes, and options in the Patterns tab of the Format Cells dialog box for the selected item. The default for each argument reflects the setting in the dialog box.

Aauto is a number from 0 to 2 specifying area settings (that is, the object's "surface area").

If aauto is	Area settings are
-------------	-------------------

0	Set by the user (custom)
---	--------------------------

1	Automatic (set by Microsoft Excel)
---	------------------------------------

2 None

Aback is a number from 1 to 56 corresponding to the 56 area background colors in the Patterns tab of the Format Cells dialog box.

Afore is a number from 1 to 56 corresponding to the 56 area foreground colors in the Patterns tab of the Format Cells dialog box.

Apattern is a number corresponding to the area patterns in the Patterns tab of the Format Cells or Format Object dialog box. If an object is selected, apattern can be from 1 to 18; if a cell is selected, apattern can be from 0 to 18. If apattern is 0 and a cell is selected, Microsoft Excel applies no pattern.

Apply is a logical value corresponding to the Apply To All check box in Microsoft Excel version 4.0. This argument is for compatibility with previous versions of Microsoft Excel and applies only when a chart data point or a data series is selected.

- If apply is TRUE, Microsoft Excel applies any formatting changes to all items that are similar to the selected item on the chart.
- If apply is FALSE, Microsoft Excel applies formatting changes only to the selected item on the chart.

Bauto is a number from 0 to 2 specifying border settings.

If bauto is	Border settings are
-------------	---------------------

0	Set by the user (custom)
1	Automatic (set by Microsoft Excel)
2	None

Bcolor is a number from 1 to 56 corresponding to the 56 border colors in the Border tab of the Format Object or Format (chart element) dialog box.

Bstyle is a number from 1 to 8 corresponding to the eight border styles in the Border tab of the Format Object or Format (chart element) dialog box.

Bwt is a number from 1 to 4 corresponding to the four border weights in the Border tab of the Format Object or Format (chart element) dialog box.

If bwt is	Border is
-----------	-----------

1	Hairline
2	Thin
3	Medium
4	Thick

Hlength is a number from 1 to 3 specifying the length of the arrowhead.

If hlength is	Arrowhead is
---------------	--------------

1	Short
2	Medium
3	Long

Htype is a number from 1 to 5 specifying the style of the arrowhead.

If htype is	Style of arrowhead is
-------------	-----------------------

1	No head
2	Open head
3	Closed head
4	Double open head
5	Double closed head

Hwidth is a number from 1 to 3 specifying the width of the arrowhead.

If hwidth is	Arrowhead is
--------------	--------------

1	Narrow
2	Medium
3	Wide

Invert is a logical value corresponding to the Invert If Negative check box in the Patterns tab of the Format Data Series dialog box. This argument applies only to data markers.

- If invert is TRUE, Microsoft Excel inverts the pattern in the selected item if it corresponds to a negative number.
- If invert is FALSE, Microsoft Excel removes the inverted pattern, if present, from the selected item corresponding to a negative value.

Lauto is a number from 0 to 2 specifying line settings.

If lauto is	Line settings are
0	Set by the user (custom)
1	Automatic (set by Microsoft Excel)
2	None

Lcolor is a number from 1 to 56 corresponding to the 16 line colors in the Patterns tab of the Format Object or Format (chart element) dialog box.

Lstyle is a number from 1 to 8 corresponding to the eight line styles in the Patterns tab of the Format Object or Format (chart element) dialog box.

Lwt is a number from 1 to 4 corresponding to the four line weights in the Patterns tab of the Format Object or Format (chart element) dialog box.

If lwt is	Line is
1	Hairline
2	Thin
3	Medium
4	Thick

Mauto is a number from 0 to 2 specifying marker settings.

If mauto is	Marker settings are
0	Set by the user
1	Automatic (set by Microsoft Excel)
2	None

Mback is a number from 1 to 56 corresponding to the 56 marker background colors in the Patterns tab of the Format Data Series dialog box.

Mfore is a number from 1 to 56 corresponding to the 56 marker foreground colors in the Patterns tab of the Format Data Series dialog box.

Mstyle is a number from 1 to 9 corresponding to the nine marker styles in the Patterns tab of the Format Data Series dialog box.

Picture_units is the number of units you want each picture to represent in a scaled, stacked picture chart. This argument applies only to picture charts and only if type is 3.

Rounded is a logical value corresponding to the Round Corners check box and specifying whether to make the corners on text boxes and rectangles rounded. If rounded is TRUE, the corners are rounded; if FALSE, the corners are square. If the selection is an arc or an oval, rounded is ignored.

Newui is a logical value that specifies whether to use the foreground, background, and patterns of Microsoft Excel 5.0. If TRUE or omitted, the colors and patterns of Microsoft Excel 5.0 will be used. If FALSE, the colors and patterns of Microsoft Excel 4.0 will be used.

Newfill is a logical value that specifies whether to use the chart patterns of Microsoft Excel 5.0. If TRUE or omitted, the chart patterns of Microsoft Excel 5.0 will be used. If FALSE, the chart patterns of Microsoft Excel 4.0 will be used.

Shadow is a logical value corresponding to the Shadow check box. Shadow does not apply to area charts or bars in bar charts. If shadow is TRUE, Microsoft Excel adds a shadow to the selected item; if FALSE, Microsoft Excel removes the shadow, if one is present, from the selected item. If the selection is an arc, shadow is ignored.

Smooth is a logical value that applies smoothing to picture markers in line or xy (scatter) charts. The default is FALSE.

Tlabel is a number from 1 to 4 specifying the position of tick labels.

If tlabel is	Tick label position is
1	None
2	Low
3	High
4	Next to axis

Tmajor is a number from 1 to 4 specifying the type of major tick marks.

If tmajor is	Type of major tick marks is
1	None
2	Inside
3	Outside
4	Cross

Tminor is a number from 1 to 4 specifying the type of minor tick marks.

If tminor is	Type of minor tick marks is
1	None
2	Inside
3	Outside
4	Cross

Type is a number from 1 to 3 specifying the type of pictures to use in a picture chart.

If type is	Pictures should be
1	Stretched to reach a particular value
2	Stacked on top of each other to reach a particular value
3	Stacked on top of each other, but you specify the number of units each picture represents

Remarks

- You can select many graphic objects on a sheet or macro sheet and apply formatting to them at the same time, but you can select only one chart item at a time.
- If you select multiple objects and if one or more of the objects requires a different form of the PATTERNS function, then choose the syntax corresponding to the object with the most formatting attributes that is, the syntax with the most arguments. If you specify an argument that does not apply to an item, the argument has no effect on that item.
- To apply formatting to similar items on a chart, use the apply argument described above.

Related Functions

FONT.PROPERTIES Applies a font to the selection

FORMAT.TEXT Formats a text box or a chart text item

List of Command-Equivalent Functions

POKE

Macro Sheets Only

Sends data to another application. Use POKE to send data to documents in other applications you are communicating with through dynamic data exchange (DDE).

Syntax

POKE(channel_num, item_text, data_ref)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Channel_num is the channel number returned by a previously run INITIATE function.

Item_text is text that identifies the item you want to send data to in the application you are accessing through channel_num. The form of item_text depends on the application connected to channel_num.

Data_ref is a reference to the document containing the data to send.

If POKE is not successful, it returns the following values.

Value returned	Meaning
#VALUE!	Channel_num is not a valid channel number.
#DIV/0!	The application you are accessing does not respond after a certain length of time, and you press ESC to cancel.
#REF!	POKE is refused.

Examples

In Microsoft Excel for Windows, the following macro inserts the text from cell C3 into the Microsoft Word for Windows document SALES.DOC at the start of the document.

```
=POKE(SendChan1, "StartOfDoc", C3)
```

In Microsoft Excel for the Macintosh, the following macro inserts the text from cell C3 into the Microsoft Word document named Report.

```
=POKE(SendChan1, "TopicName", C3)
```

Related Functions

INITIATE Opens a channel to another application

REQUEST Returns data from another application

TERMINATE Closes a channel to another application

List of DDE/External Functions

PRINT

Macro Sheets Only

Equivalent to choosing the Print command from the File menu. Prints the active workbook.

Arguments correspond to options, check boxes, and edit boxes in the Print dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Syntax

PRINT(range_num, from, to, copies, draft, preview, print_what, color, feed, quality, y_resolution, selection)

PRINT?(range_num, from, to, copies, draft, preview, print_what, color, feed, quality, y_resolution, selection)

Range_num is a number specifying which pages to print.

Range_num	Prints the following pages
-----------	----------------------------

- | | |
|---|---|
| 1 | All the pages |
| 2 | Prints a specified range. If range_num is 2, then from and to are required arguments. |

From specifies the first page to print. This argument is ignored unless range_num equals 2.

To specifies the last page to print. This argument is ignored unless range_num equals 2.

Copies specifies the number of copies to print. If omitted, the default is 1.

Draft This argument overrides the draft argument from the PAGE.SETUP function. If omitted, the Draft Setting from the Page.Setup function is used.

Preview is a logical value corresponding to the Print Preview button in the Print dialog box. If TRUE, the print preview window will be displayed. If FALSE, the window will not be displayed

Print_what is a number from 1 to 3 that specifies what parts of the sheet or macro sheet to print. If a chart is active, print_what is ignored. This argument will override the setting in the Page Setup dialog box. If omitted, the note argument in the Page.Setup function is used to determine whether to print notes or not.

Print_what	Prints
------------	--------

- | | |
|---|----------------------|
| 1 | Sheet only |
| 2 | Notes only |
| 3 | Sheet and then notes |

Color corresponds to the Print Using Color check box. Color is available only in Microsoft Excel for the Macintosh. If omitted, the setting is not changed.

Feed is a number specifying the type of paper feed. Feed is available only in Microsoft Excel for the Macintosh.

Feed	Type of paper feed
------	--------------------

- | | |
|--------------|-----------------------------------|
| 1 or omitted | Continuous (paper cassette) |
| 2 | Cut sheet or manual (manual feed) |

Quality Specifies the DPI output quality you want. If omitted, the corresponding settings in the Page Setup dialog box will be used. If included, this argument overrides the quality argument in the PAGE SETUP dialog box.

Y_resolution corresponds to the Print Quality box in the Page Setup dialog box if you have specified a printer where the horizontal and vertical resolution are not equal, such as a dot-matrix printer. If omitted, the corresponding settings in the Page Setup dialog box will be used. If included, this argument overrides the print quality setting in the Page Setup dialog box.

Selection specifies what portion of the sheet to print.

Selection	Portion printed
-----------	-----------------

- | | |
|---|---|
| 1 | Prints the current selection from all selected sheets. For example, if A1:F40 is selected on the active sheet, A1:F40 will be printed from each of the selected sheets. |
| 2 | Prints the print area or entire sheet from all selected sheets. |

Related Functions

<u>PAGE.SETUP</u>	Sets page printing specifications
<u>PRINT.PREVIEW</u>	Previews pages and page breaks before printing
<u>PRINTER.SETUP</u>	Identifies the printer
<u>SET.PRINT.AREA</u>	Defines the print area
<u>SET.PRINT.TITLES</u>	Defines text to print as titles
<u>DEFINE.NAME</u>	Equivalent to choosing the Define command from the Name submenu on the Insert menu

List of Command-Equivalent Functions

PRINT.PREVIEW

Macro Sheets Only

Equivalent to choosing the Print Preview command from the File menu. Previews the pages and page breaks of the active workbook on the screen before printing.

Syntax

PRINT.PREVIEW()

Related Function

PRINT Prints the active workbook

List of Command-Equivalent Functions

REFTEXT

Macro Sheets Only

Converts a reference to an absolute reference in the form of text. Use REFTEXT when you need to manipulate references with text functions. After manipulating the reference text, you can convert it back into a normal reference by using TEXTREF.

Syntax

REFTEXT(reference, a1)

Reference is the reference you want to convert.

A1 is a logical value specifying A1-style or R1C1-style references.

- If a1 is TRUE, REFTEXT returns an A1-style reference.
- If a1 is FALSE or omitted, REFTEXT returns an R1C1-style reference.

Examples

REFTEXT(C3, TRUE) equals "\$C\$3"

REFTEXT(B2:F2) equals "R2C2:R2C6"

If the active cell is B9 on the active sheet named SHEET1, then:

REFTEXT(ACTIVE.CELL()) equals "[Book1]SHEET1!R9C2"

REFTEXT(ACTIVE.CELL(), TRUE) equals "[Book1]SHEET1!\$B\$9"

Related Functions

<u>ABSREF</u>	Returns the absolute reference of a range of cells to another range
<u>DEREF</u>	Returns the values of cells in the reference
<u>OFFSET</u>	Returns a reference offset from a given reference
<u>RELREF</u>	Returns a relative reference
<u>TEXTREF</u>	Converts text to a reference

List of [Lookup & Reference Functions](#)

REGISTER

Macro Sheets Only

Registers the specified dynamic link library (DLL) or code resource and returns the register ID. You can also specify a custom function name and argument names that will appear in the Function Wizard. If you register a command (macro_type = 2), you can also specify a shortcut key. Because Microsoft Excel for Windows and Microsoft Excel for the Macintosh use different types of code resources, REGISTER has a slightly different syntax form when used in each operating environment.

For more information about DLLs and code resources, see the "Using the CALL and REGISTER Functions" in the Appendix for the Worksheet Function Reference, or [Using the CALL and REGISTER Functions](#) in online Help.

Important This function is provided for advanced users only. If you use the CALL function incorrectly, you could cause errors that will require you to restart your computer.

Syntax 1

For Microsoft Excel for Windows

REGISTER(module_text, procedure, type_text, function_text, argument_text, macro_type, category, shortcut_text, help_topic, function_help, argument_help1, argument_help2,...)

Syntax 2

For Microsoft Excel for the Macintosh

REGISTER(file_text, resource, type_text, function_text, argument_text, macro_type, category, shortcut_text, help_topic, function_help, argument_help1, argument_help2,...)

Module_text or file_text is text specifying the name of the DLL that contains the function (in Microsoft Excel for Windows) or the name of the file that contains the code resource (in Microsoft Excel for the Macintosh).

Procedure or resource is text specifying the name of the function in the DLL (in Microsoft Excel for Windows) or the name of the code resource (in Microsoft Excel for the Macintosh). In Microsoft Excel for Windows, you can also use the ordinal value of the function from the EXPORTS statement in the module-definition file (.DEF). In Microsoft Excel for the Macintosh, you can also use the resource ID number. The ordinal value or resource ID number should not be in text form.

This argument may be omitted for stand-alone DLLs or code resources. In this case, REGISTER will register all functions or code resources and then return module_text or file_text.

Type_text is text specifying the data type of the return value and the data types of all arguments to the DLL or code resource. The first letter of type_text specifies the return value. The codes you use for type_text are described in "Using the CALL and REGISTER Functions" in the Appendix for the Worksheet Function Reference, or [Using the CALL and REGISTER Functions](#) in online Help. For stand-alone DLLs or code resources (XLLs), you can omit this argument.

Function_text is text specifying the name of the function as you want it to appear in the Function Wizard. If you omit this argument, the function will not appear in the Function Wizard.

Argument_text is text specifying the names of the arguments you want to appear in the Function Wizard. Argument names should be separated by commas.

Macro_type specifies the macro type: 1 for a function or 2 for a command. If macro_type is omitted, it is assumed to be 1 (function).

Category specifies the function category in the Function Wizard in which you want the registered function to appear. You can use the category number or the category name for category. If you use the category name, be sure to enclose it in double quotation marks. If category is omitted, it is assumed to be 14 (User Defined).

Category number	Category name
1	Financial
2	Date & Time
3	Math & Trig
4	Text
5	Logical

6	Lookup & Matrix
7	Database
8	Statistical
9	Information
10	Commands (macro sheets only)
11	Actions (macro sheets only)
12	Customizing (macro sheets only)
13	Macro Control (macro sheets only)
14	User Defined

Shortcut_text is a character specifying the shortcut key for the registered command. The shortcut key is case-sensitive. This argument is used only if **macro_type** = 2 (command). If **shortcut_text** is omitted, the command will not have a shortcut key.

Help_topic is the reference (including path) to the help file that you want displayed when the user chooses the Help button when your custom function is displayed.

Function_help is a text string describing your custom function when it is selected in the Function Wizard. The maximum number of characters is 255.

Argument_help1, argument_help2 are 1 to 21 text strings that describes you custom function's arguments when the function is selected in the Function Wizard.

Example

Syntax 1

In Microsoft Excel for Windows, the following macro formula registers the GetTickCount function from Microsoft Windows. This function returns the number of milliseconds that have elapsed since Microsoft Windows was started.

```
REGISTER("User", "GetTickCount", "J")
```

Assuming that the REGISTER function is in cell A5, after your macro registers GetTickCount, you can use the CALL function to return the number of milliseconds that have elapsed:

```
CALL(A5)
```

Example

Syntax 1 with optional function_text

You can use the following macro formula to register the GetTickCount function from Microsoft Windows and assign the custom name GetTicks to it. To do this, include "GetTicks" as the optional **function_text** argument to the REGISTER function.

```
REGISTER("User", "GetTickCount", "J", "GetTicks", , 1, 9)
```

After the function is registered, the custom name GetTicks will appear in the Information function category (category = 9) in the Function Wizard.

You can call the function from the same macro sheet on which it was registered using the following formula:

```
GetTicks()
```

You can call the function from another sheet or macro sheet by including the name of the original macro sheet in the formula. For example, assuming the macro sheet on which GetTicks was registered is named MACRO1.XLS, the following formula calls the function from another sheet:

```
MACRO1.XLS!GetTicks()
```

Tip You can use functions in a DLL or code resource directly on a sheet without first registering them from a macro sheet. Use syntax 2a or 2b of the CALL function. For more information, see CALL.

Related Functions

[CALL](#) Calls a procedure in a dynamic link library or code resource

[REGISTER.ID](#) Returns the register ID of the resource

[UNREGISTER](#) Removes a registered code resource from memory

List of [DDE/External Functions](#)

RELREF

Macro Sheets Only

Returns the reference of a cell or cells relative to the upper-left cell of `rel_to_ref`. The reference is given as an R1C1-style relative reference in the form of text, such as "R[1]C[1]".

Syntax

RELREF(reference, rel_to_ref)

Reference is the cell or cells to which you want to create a relative reference.

Rel_to_ref is the cell from which you want to create the relative reference.

Tip If you know the absolute reference of a cell that you want to include in a formula, but your formula requires a relative reference, use RELREF to generate the relative reference. This is especially useful with the FORMULA function, since its formula_text argument requires R1C1-style references, and RELREF returns relative R1C1-style references. You can also use the FORMULA.CONVERT function to convert absolute references to relative references.

Examples

RELREF(\$A\$1, \$C\$3) equals "R[-2]C[-2]"

RELREF(\$A\$1:\$E\$5, \$C\$3:\$G\$7) equals "R[-2]C[-2]:R[2]C[2]"

RELREF(\$A\$1:\$E\$5, \$C\$3) equals "R[-2]C[-2]:R[2]C[2]"

Related Functions

ABSREF Returns the absolute reference of a range of cells to another range

DEREF Returns the value of the cells in the reference

FORMULA Enters values into a cell or range or onto a chart

FORMULA.CONVERT Changes the reference style and type

OFFSET Returns a reference offset from a given reference

List of Lookup & Reference Functions

REPORT.DEFINE

Macro Sheets Only

Equivalent to choosing the Print Report command from the File menu and then choosing the Add option from the Print Report dialog box. Creates or replaces a report definition. If this function is not available, you must install the Reports add-in macro. For more information, see "Installing Addin Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Adding or removing an addin](#) in online Help.

Syntax

REPORT.DEFINE(report_name, sections_array, pages_logical)

Report_name specifies the name of the report. If the workbook already contains a report with report_name, the new report replaces the existing one.

Sections_array is an array that contains one or more rows of view, scenario, and sheet name that define the report. The sheet name is the sheet on which the view and scenario are defined. If the sheet name is not specified, the current sheet is used when REPORT.DEFINE is run.

Pages_logical is a logical value that, if TRUE or omitted, specifies continuous page numbers for multiple sections or, if FALSE, resets page numbers to 1 for each new section.

Remarks

- REPORT.DEFINE returns the #VALUE error value if report_name is invalid or if the workbook is protected.
- If there are no reports defined, this function will bring up the Add Report dialog box from the File Print Report menu.

Related Functions

[REPORT.DELETE](#) Removes a report from the active workbook

[REPORT.PRINT](#) Prints a report

[REPORT.GET](#) Returns information about reports defined for the active workbook

List of [Command-Equivalent Functions](#)

REPORT.DELETE

Macro Sheets Only

Equivalent to choosing the Print Report command from the File menu and then selecting a report in the Print Report dialog box and choosing the Delete button. Removes a report definition from the active workbook.

If this function is not available, you must install the Reports add-in macro. For more information, see "Installing Addin Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Adding or removing an addin](#) in online Help.

Syntax

REPORT.DELETE(report_name)

Report_name specifies the name of the report to be removed. Report_name can be any text that does not contain quotation marks.

Remarks

REPORT.DELETE returns the #VALUE error value if report_name is invalid or if the workbook is protected.

Related Functions

[REPORT.DEFINE](#) Creates a report

[REPORT.PRINT](#) Prints a report

[REPORT.GET](#) Returns information about reports defined for the active workbook

List of [Command-Equivalent Functions](#)

REPORT.GET

Macro Sheets Only

Returns information about reports defined for the active workbook. Use REPORT.GET to return information you can use in other macro commands that manipulate reports.

If this function is not available, you must install the Reports add-in macro. For more information, see "Installing Addin Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Adding or removing an addin](#) in online Help.

Syntax

REPORT.GET(type_num, report_name)

Type_num is a number from 1 to 3 specifying the type of information to return, as shown in the following table.

Type_num	Returns
1	An array of reports from all sheets in the active workbook or the #N/A error value if none are specified
2	An array of views, scenarios, and sheet names for the specified report in the active workbook. REPORT.GET returns the #N/A error value if the scenario check box is not selected. Returns the #VALUE! error value if name is invalid or the workbook is protected.
3	If continuous page numbers are used, returns TRUE. If page numbers start at 1 for each section, returns FALSE. Returns the #VALUE! error value if report_name is invalid or the workbook is protected.

Report_name specifies the name of a report in the active workbook.

Remarks

Report_name is required if type_num is 2 or 3.

Example

The following macro formula returns an array of reports from the active workbook.

```
REPORT.GET(1)
```

Related Functions

[REPORT.DEFINE](#) Creates a report

[REPORT.DELETE](#) Removes a report from the active workbook

[REPORT.PRINT](#) Prints a report

List of [Command-Equivalent Functions](#)

REPORT.PRINT

Macro Sheets Only

Equivalent to choosing the Print button from the Print Reports dialog box. Prints a report.

If this function is not available, you must install the Reports add-in macro. For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

REPORT.PRINT(report_name, copies_num, show_print_dlg_logical)

REPORT.PRINT?(report_name, copies_num)

Report_name specifies the name of a report in the active workbook.

Copies_num is the number of copies you want to print. If omitted, the default is 1.

Show_print_dlg_logical is a logical value that, if TRUE, displays a dialog box asking how many copies to print, or, if FALSE or omitted, prints the report immediately using existing print settings.

Remarks

REPORT.PRINT returns the #VALUE! error value if report_name is invalid or if the workbook is protected.

Related Functions

[REPORT.DEFINE](#) Creates a report

[REPORT.DELETE](#) Removes a report from the active workbook

List of [Command-Equivalent Functions](#)

REQUEST

Macro Sheets Only

Requests an array of a specific type of information from an application with which you have a dynamic data exchange (DDE) link. Use REQUEST with other Microsoft Excel DDE functions to move information from another application into Microsoft Excel.

Syntax

REQUEST(channel_num, item_text)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Channel_num is a number returned by a previously run INITIATE function. Channel_num refers to a channel through which Microsoft Excel communicates with another program.

Item_text is a code indicating the type of information you want to request from another application. The form of item_text depends on the application connected to channel_num.

REQUEST returns the data as an array. For example, suppose the remote data to be returned came from a sheet that looked like the following illustration.

	A	B	C	D	E
1	1	2	3		
2	4	5	6		

REQUEST would return that data as the following array:

{1, 2, 3;4, 5, 6}

If REQUEST is not successful, it returns the following error values.

Value returned	Situation
----------------	-----------

#VALUE!	Channel_num is not a valid channel number.
#N/A	The application you are accessing is busy doing something else.
#DIV/0!	The application you are accessing does not respond after a certain length of time, or you have pressed ESC or COMMAND+PERIOD to cancel.
#REF!	The request is refused.

Tip Use the ERROR.TYPE function to distinguish between the different error values.

Example

Suppose you had opened a DDE channel to Microsoft Word for Windows. WChan contains the number of the open channel. In Microsoft Excel for Windows, the following function returns the text specified by the bookmark named BMK1.

=REQUEST(WChan, "BMK1")

Related Functions

<u>EXECUTE</u>	Carries out a command in another application
<u>INITIATE</u>	Opens a channel to another application
<u>POKE</u>	Sends data to another application
<u>SEND.KEYS</u>	Sends a key sequence to another application
<u>TERMINATE</u>	Closes a dynamic data exchange (DDE) channel previously opened with the INITIATE function

List of DDE/External Functions

SCENARIO.ADD

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then choosing the Add button. Defines the specified values as a scenario. A scenario is a set of values to be used as input for a model on your worksheet.

Syntax

SCENARIO.ADD(scen_name, value_array, changing_ref, scen_comment, locked, hidden)

Scen_name is the name of the scenario you want to define.

Value_array is a horizontal array of values you want to use as input for the model on your worksheet.

- Any entry that would be valid for a cell in your model can be a value in value_array.
 - The values must be arranged in the same order as the model's changing cells. The changing cells are listed in the Changing Cells box in the Scenario Manager dialog box.
 - If value_array is omitted, it is assumed to contain the current values of the changing cells.
- Changing_ref is a reference to cells you want to define as changing cells for a scenario.
- If omitted, uses the changing cells for the last scenario defined for the sheet.
 - If changing_ref contains nonadjacent references, you must separate the reference areas by commas (or other list separator). If you are using A1-style references, then you must enclose reference in an extra set of parentheses.

Scen_comment is text specifying a descriptive comment for the scenario defined by scen_name.

Locked is a logical value that corresponds to the Prevent Changes check box in the Add or Edit Scenario dialogs boxes. If TRUE or omitted, prevents users from changing values in a scenario. If FALSE, users are allowed to make changes to the scenario. The locking will not become enabled until the sheet is protected with the Protect Sheet command from the Protection submenu on the Tools menu.

Hidden is a logical value that corresponds to the Hide check box in the Add or Edit Scenario dialog boxes. If TRUE, the scenario will be hidden from view from the users and will not appear in the Scenario Manager dialog box. If FALSE or omitted, the scenario will remain unhidden. The scenario will not become hidden until the sheet is protected with the Protect Sheet command from the Protection submenu on the Tools menu.

Related Function

REPORT.DEFINE Creates a report

SCENARIO.GET Returns the specified information about the scenarios defined on your worksheet

SCENARIO.CELLS

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then editing the Changing Cells box. Defines the changing cells for a model on your worksheet. Changing cells are the cells into which values will be entered when you display a scenario. If you have only one set of changing cells on your sheet, SCENARIO.CELLS will change the changing cells for all scenarios. If your sheet has scenarios defined with multiple sets of changing cell, this function returns an error and the macro is halted. This function is provided for compatibility with Microsoft Excel 4.0. Use SCENARIO.EDIT with the changing_ref argument instead of SCENARIO.CELLS if you want to change the changing cells of a scenario.

Syntax

SCENARIO.CELLS(changing_ref)

SCENARIO.CELLS? (changing_ref)

Changing_ref is a reference to the cells you want to define as changing cells for the model. If changing_ref contains nonadjacent references, you must separate the reference areas by commas and enclose changing_ref in an extra set of parentheses.

Related Functions

SCENARIO.EDIT Equivalent to choosing the Scenarios command from the Tools menu and then choosing the Edit button

List of Command-Equivalent Functions

SCENARIO.DELETE

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then selecting a scenario and choosing the Delete button. Deletes the specified scenario.

Syntax

SCENARIO.DELETE(scen_name)

Scen_name is the name of the scenario you want to delete.

Related Function

<u>SCENARIO.GET</u>	Returns the specified information about the scenarios defined on your worksheet
<u>SCENARIO.ADD</u>	Equivalent to choosing the Scenario Manager command from the Tools menu and then choosing the Add button
<u>SCENARIO.EDIT</u>	Equivalent to choosing the Scenario Manager command from the Tools menu and then choosing the Edit button

SCENARIO.EDIT

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then choosing the Edit button.

Syntax

SCENARIO.EDIT(scen_name, new_scenname, value_array, changing_ref, scen_comment, locked, hidden)

SCENARIO.EDIT?(scen_name, new_scenname, value_array, changing_ref, scen_comment, locked, hidden)

Scen_name is the name of the scenario that you want to edit.

New_scenname is the new name you want to give to the scenario.

Value_array is a horizontal array of values that you want to use for the scenario.

- If value_array is omitted but changing_ref is specified, Scenario Manager uses the values in changing_ref as value_array.

- Value_array must match the dimensions of changing_ref for the scenario being edit.

Changing_ref is a reference to cells you want to define as changing cells for a scenario.

Scen_comment is text specifying a descriptive comment for the scenario you want to edit.

Locked is a logical value that corresponds to the Prevent Changes check box in the Add or Edit Scenario dialogs boxes. If TRUE or omitted, prevents users from changing values in a scenario. If FALSE, users are allowed to make changes to the scenario. The locking will not become enabled until the sheet is protected with the Protect Sheet command from the Protection submenu on the Tools menu.

Hidden is a logical value that corresponds to the Hide check box in the Add or Edit Scenario dialog boxes. If TRUE, the scenario will be hidden from view from the users. If FALSE or omitted, the scenario will remain unhidden. The scenario will not become hidden until the sheet is hidden with the Hide command from the Window menu.

Related Function

SCENARIO.GET

Returns the specified information about the scenarios defined on your worksheet

SCENARIO.ADD

Equivalent to choosing the Scenario Manager command from the Tools menu and then choosing the Add button

SCENARIO.DELETE

Equivalent to choosing the Scenario Manager command from the Tools menu and then selecting a scenario and choosing the Delete button

SCENARIO.GET

Macro Sheets Only

Returns the specified information about the scenarios defined on your worksheet.

Syntax

SCENARIO.GET(type_num, scen_name)

Type_num is a number from 1 to 8 specifying the type of information you want.

Type_num	Information returned
1	A horizontal array of all scenario names in the form of text
2	A reference to the set of changing cells of scen_name (specified in the Changing Cells box of the Scenario Manager dialog box). If scen_name is omitted, the first scenario is used.
3	A reference to the result cells (specified in the Result Cells box in the Scenario Summary dialog box)
4	An array of scenario values for the scenario scen_name . Each scenario is in a separate row. If scen_name is omitted, the first scenario is used.
5	Comment, as text, for the scenario
6	Returns TRUE if the specified scenario is locked to prevent changes; FALSE, if unlocked. Scen_name is required.
7	Returns TRUE if the specified scenario is hidden; FALSE, if visible to the user. Scen_name is required.
8	Returns the user name of the person who last modified the scenario by either adding or editing a scenario. Scen_name is required.
Scen_name	is the name of the scenario that you want information about. Ignored if type_num equals 1 or 3.

Remarks

In the returned array of scenario values, the number of rows is the number of scenarios, and the number of columns is the number of changing cells.

SCENARIO.SHOW

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then selecting a scenario and choosing the Show button. Recalculates a model using the specified scenario and displays the result.

Syntax

SCENARIO.SHOW(scen_name)

Scen_name is the name of the previously defined scenario whose values you want to switch to.

Related Functions

List of [Command-Equivalent Functions](#)

SCENARIO.SHOW.NEXT

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu, selecting the next scenario from the Scenarios list, and choosing the Show button. Recalculates a model using the next scenario and displays the result.

Syntax

SCENARIO.SHOW.NEXT()

Remarks

After displaying the last scenario, running SCENARIO.SHOW.NEXT again displays the first scenario.

Related Functions

List of [Command-Equivalent Functions](#)

SCENARIO.SUMMARY

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then choosing the Summary button. Generates a table summarizing the results of all the scenarios for the model on your worksheet.

Syntax

SCENARIO.SUMMARY(result_ref, report_type)

SCENARIO.SUMMARY?(result_ref, report_type)

Result_ref is a reference to the result cells you want to include in the summary report. Normally, result_ref refers to one or more cells containing the formulas that depend on the changing cell values for your model that is, the cells that show the results of a particular scenario.

- If result_ref is omitted, no result cells are included in the report.
 - If result_ref contains nonadjacent references, you must separate the reference areas by commas and enclose result_ref in an extra set of parentheses.
- Report_type is a number specifying the type of report desired.

Report_type	Type of Report
1 or omitted	A scenario summary report (Microsoft Excel 4.0)
2	A scenario PivotTable (new for Microsoft Excel 5.0). Requires result_ref.

Remarks

- SCENARIO.SUMMARY generates a summary table of the changing cell and result cell values for each scenario.
- The table is generated on a new sheet in the current workbook. The sheet becomes active after SCENARIO.SUMMARY runs.

Related Functions

List of [Command-Equivalent Functions](#)

SEND.KEYS

Macro Sheets Only

Sends keystrokes to the active application just as if they were typed at the keyboard. Use SEND.KEYS to send keystrokes that perform actions and execute commands to applications you are running with Microsoft Excel's other dynamic data exchange (DDE) functions.

Syntax

SEND.KEYS(key_text, wait_logical)

Note This function is available only in Microsoft Excel for Windows.

Key_text is the key or key combination you want to send to another application. The format for key_text is described in the ON.KEY function.

Wait_logical is a logical value that determines whether the macro continues before the actions caused by key_text are carried out.

- If wait_logical is TRUE, Microsoft Excel waits for the keys to be processed before returning control to the macro.
- If wait_logical is FALSE or omitted, the macro continues running without waiting for the keys to be processed.

Remarks

If Microsoft Excel is the active application, wait_logical is assumed to be FALSE, even if you enter wait_logical as TRUE. This is because if wait_logical is TRUE, Microsoft Excel waits for the keys to be processed in the other application before returning control to the macro. Microsoft Excel doesn't process keys while a macro is running.

Example

The following macro uses the Calculator application in Microsoft Excel for Windows to multiply some numbers, and then cuts the result and pastes it into Microsoft Excel.

```
=EXEC("CALC.EXE", 1)
=SEND.KEYS("10*30", TRUE)
=SEND.KEYS("~", TRUE)
=SEND.KEYS("%ec", TRUE)
=APP.ACTIVATE(, FALSE)
=SELECT(!B1)
=PASTE()
=RETURN()
```

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>EXECUTE</u>	Carries out a command in another application
<u>ON.KEY</u>	Runs a macro when a specified key is pressed

List of [DDE/External Functions](#)

SET.CRITERIA

Macro Sheets Only

Equivalent to choosing the Set Criteria command from the Data menu in Microsoft Excel version 4.0.
Defines the name Criteria for the selected range on a sheet or macro sheet.

Syntax

SET.CRITERIA()

Related Functions

SET.DATABASE Equivalent to choosing the Set Database command from the Data menu in Microsoft Excel version 4.0

SET.EXTRACT Equivalent to choosing the Set Extract command from the Data menu in Microsoft Excel version 4.0

List of Command-Equivalent Functions

SET.DATABASE

Macro Sheets Only

Equivalent to choosing the Set Database command from the Data menu in Microsoft Excel version 4.0.
Defines the name Database for the selected range on a sheet or macro sheet.

Syntax

SET.DATABASE()

Related Functions

SET.CRITERIA Equivalent to choosing the Set Criteria command from the Data menu in Microsoft Excel version 4.0

SET.EXTRACT Equivalent to choosing the Set Extract command from the Data menu in Microsoft Excel version 4.0

List of Command-Equivalent Functions

SET.EXTRACT

Macro Sheets Only

Equivalent to choosing the Set Extract command from the Data menu in Microsoft Excel version 4.0.
Defines the name Extract for the selected range on the active sheet.

Syntax

SET.EXTRACT()

Related Functions

SET.DATABASE Equivalent to choosing the Set Database command from the Data menu in Microsoft Excel version 4.0

SET.CRITERIA Equivalent to choosing the Set Criteria command from the Data menu in Microsoft Excel version 4.0

List of Command-Equivalent Functions

SET.UPDATE.STATUS

Macro Sheets Only

Sets the update status of a link to automatic or manual. Use SET.UPDATE.STATUS to change the way a link is updated.

Syntax

SET.UPDATE.STATUS(link_text, status, type_of_link)

Link_text is the path of the linked file for which you want to change the update status.

Status is the number 1 or 2 and describes how you want the link to be updated.

Status	Update method
1	Automatic
2	Manual

Type_of_link is a number from 1 to 4 that specifies what type of link you want to get information about.

Type_of_link	Link document type
1	Not available
2	DDE/OLE link
3	Not available
4	Not available

Example

In Microsoft Excel for Windows, the following macro formula sets the update status of the DDE link to Microsoft Word for Windows to manual:

```
SET.UPDATE.STATUS("WordDocument|'C:\MEMO.DOC'!DDE.LINK1", 2, 2)
```

Related Functions

GET.LINK.INFO Returns information about a link

UPDATE.LINK Updates a link to another document

List of Command-Equivalent Functions

SIZE

Macro Sheets Only

Equivalent to choosing the Size command from the Control menu in Microsoft Excel for Windows version 3.0 or earlier or to changing the size of a window by dragging its border. In Microsoft Excel for the Macintosh version 3.0 or earlier, equivalent to changing the size of a window by dragging its size box. This function is included only for macro compatibility and will be converted to WINDOW.SIZE when you open older macro sheets. For more information, see WINDOW.SIZE.

Syntax

SIZE(width,height>window_text)

SIZE?(width,height>window_text)

Related Functions

WINDOW.SIZE Changes the size of the active window

List of Command-Equivalent Functions

SLIDE.COPY.ROW

Macro Sheets Only

Equivalent to choosing the Copy Row button on a slide show document. Copies the selected slides, each of which is defined on a single row, to the Clipboard.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.COPY.ROW()

Remarks

- SLIDE.COPY.ROW, SLIDE.CUT.ROW, SLIDE.DELETE.ROW, and SLIDE.PASTE.ROW return TRUE if successful, or FALSE if not successful. If the active document is not a slide show or is protected, these functions return the #N/A error value. If the current selection is not valid, these functions return the #VALUE! error value.

Related Functions

[SLIDE.CUT.ROW](#)

Cuts the selected slides and pastes them onto the Clipboard

[SLIDE.DEFAULTS](#)

Specifies default values for the active slide show document

[SLIDE.DELETE.ROW](#)

Deletes the selected slides

[SLIDE.EDIT](#)

Changes the attributes of the selected slide

[SLIDE.GET](#)

Returns information about a slide or slide show

[SLIDE.PASTE](#)

Pastes the contents of the Clipboard onto a slide

[SLIDE.PASTE.ROW](#)

Pastes previously cut or copied slides onto the current selection

[SLIDE.SHOW](#)

Starts a slide show in the active document

List of [Command-Equivalent Functions](#)

SLIDE.CUT.ROW

Macro Sheets Only

Equivalent to choosing the Cut Row button on a slide show document. Cuts the selected slides, each of which is defined on a single row, and pastes them onto the Clipboard. For more information, see [SLIDE.COPY.ROW](#).

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.CUT.ROW()

Related Functions

[SLIDE.COPY.ROW](#)

Copies the selected slides and pastes them onto the Clipboard

List of [Command-Equivalent Functions](#)

SLIDE.DEFAULTS

Macro Sheets Only

Equivalent to choosing the Set Defaults button on a slide show document. Specifies the default values for the transition effect, speed, advance rate, and sound on the active slide show document.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.DEFAULTS(effect_num, speed_num, advance_rate_num, soundfile_text)

SLIDE.DEFAULTS?(effect_num, speed_num, advance_rate_num, soundfile_text)

For a description of the arguments, see SLIDE.PASTE. If an argument is omitted, its default value is not changed.

Remarks

- SLIDE.DEFAULT returns TRUE if it successfully changes the default values, or FALSE if you choose the Cancel button when using the dialog-box form. If the active document is not a slide show or is protected, SLIDE.DEFAULT returns the #N/A error value.

Related Functions

List of [Command-Equivalent Functions](#)

SLIDE.DELETE.ROW

Macro Sheets Only

Equivalent to choosing the Delete Row button on a slide show document. Deletes the selected slides, each of which is defined on a single row. For more information, see SLIDE.COPY.ROW.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.DELETE.ROW()

Related Functions

[SLIDE.COPY.ROW](#)

Copies the selected slides and pastes them onto the Clipboard

List of [Command-Equivalent Functions](#)

SLIDE.EDIT

Macro Sheets Only

Equivalent to choosing the Edit button in a slide show document. Gives the currently selected slide the attributes you specify.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.EDIT(effect_num, speed_num, advance_rate_num, soundfile_text)

SLIDE.EDIT?(effect_num, speed_num, advance_rate_num, soundfile_text)

For a description of the arguments, see SLIDE.PASTE.

Remarks

- SLIDE.EDIT returns TRUE if it successfully edits the slide, or FALSE if you choose the Cancel button when using the dialog-box form. If the active document is not a slide show or is protected, SLIDE.EDIT returns the #N/A error value. If the current selection is not a valid slide, SLIDE.EDIT returns the #VALUE error value.

Related Functions

[SLIDE.PASTE](#)

Pastes the contents of the Clipboard onto a slide

List of [Command-Equivalent Functions](#)

SLIDE.GET

Macro Sheets Only

Returns the specified information about a slide show or a specific slide in the slide show.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.GET(type_num, name_text, slide_num)

Type_num is a number specifying the type of information you want.

These values of type_num return information about a slide show.

Type_num	Type of information
1	Number of slides in the slide show
2	A two-item horizontal array containing the numbers of the first and last slides in the current selection, or the #VALUE error value if the selection is nonadjacent
3	Version number of the Slide Show add-in macro that created the slide show document

These values of type_num return information about a specific slide in the slide show.

Type_num	Type of information
4	Transition effect number
5	Transition effect name
6	Transition effect speed
7	Number of seconds the slide is displayed before advancing
8	Name of the sound file associated with the slide, or empty text (""), if none is specified (in Microsoft Excel for the Macintosh, this includes the number or name of the sound resource within the sound file)

Name_text is the name of an open slide show document for which you want information. If name_text is omitted, it is assumed to be the active document.

Slide_num is the number of the slide about which you want information.

- If slide_num is omitted, it is assumed to be the slide associated with the active cell on the document specified by name_text.
- If type_num is 1 through 3, slide_num is ignored.

Related Functions

List of [Information Functions](#)

SLIDE.PASTE

Macro Sheets Only

Equivalent to choosing the Paste button in a slide show document. Pastes the contents of the Clipboard as the next available slide of the active slide show document, and gives the slide the attributes you specify.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.PASTE(effect_num, speed_num, advance_rate_num, soundfile_text)

SLIDE.PASTE?(effect_num, speed_num, advance_rate_num, soundfile_text)

Effect_num is a number specifying the transition effect you want to use when displaying the slide.

- The numbers correspond to the effects in the Effect list in the Edit Slide dialog box. The first effect in the list is 1 (None).
- If effect_num is omitted, the default setting is used.

Speed_num is a number from 1 to 10 specifying the speed of the transition effect.

- If speed_num is omitted, the default setting is used.
- If speed_num is greater than 10, Microsoft Excel uses the value 10 anyway.
- If effect_num is 1 (none), speed_num is ignored.

Advance_rate_num is a number specifying how long (in seconds) the slide is displayed before advancing to the next one.

- If advance_rate_num is omitted, the default setting is used.
- If advance_rate_num is 0, you must press a key or click with the mouse to advance to the next slide.

Soundfile_text is the name of a file enclosed in quotation marks and specifies sound that will be played when the slide is displayed.

- If soundfile_text is omitted, Microsoft Excel plays the default sound defined for the slide show document, if any.
- If soundfile_text is empty text (""), no sound is played.
- In Microsoft Excel for the Macintosh, soundfile_text also includes the number or name of the sound resource to play in the file.

Resource is the number or name of a sound resource in soundfile_text.

- This argument applies only to Microsoft Excel for the Macintosh.
- If resource is omitted, Microsoft Excel uses the first resource in the file.
- If the file does not contain a sound resource with the specified name or number, Microsoft Excel halts the macro and displays an error message.

Remarks

- SLIDE.PASTE returns TRUE if it successfully pastes the slide, or FALSE if you choose the Cancel button when using the dialog-box form. If the active document is not a slide show or is protected, SLIDE.PASTE returns the #N/A error value. If the Clipboard format is not compatible with the slide show document's format, SLIDE.PASTE returns the #VALUE error value.

Examples

In Microsoft Excel for Windows, the following macro formula pastes the contents of the Clipboard into the active slide show document. The slide's transition effect is fade, at a speed of 8; it is displayed for five seconds; and Microsoft Excel plays the specified sound file:

```
SLIDE.PASTE(3, 8, 5, "C:\SLIDES\SOUND\MACHINES.WAV")
```

In Microsoft Excel for the Macintosh, the formula is:

```
SLIDE.PASTE(3, 8, 5, "HARD DISK:SLIDES:SOUND:MACHINE SOUNDS")
```

Related Functions

List of [Command-Equivalent Functions](#)

SLIDE.PASTE.ROW

Macro Sheets Only

Equivalent to choosing the Paste Row button in a slide show document. Pastes previously cut or copied slides onto the current selection. For more information, see SLIDE.COPY.ROW.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.PASTE.ROW()

Related Functions

[SLIDE.COPY.ROW](#)

Copies the selected slides and pastes them onto the Clipboard

List of [Command-Equivalent Functions](#)

SLIDE.SHOW

Macro Sheets Only

Equivalent to choosing the Start Show button in a slide show document. Starts the slide show in the active document.

If this function is not available, you must install the Slide Show add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SLIDE.SHOW(initialslide_num, repeat_logical, dialogtitle_text, allownav_logical, allowcontrol_logical)
SLIDE.SHOW?(initialslide_num, repeat_logical, dialogtitle_text, allownav_logical, allowcontrol_logical)

All arguments except dialogtitle_text correspond to options and settings in the Start Show dialog box.

Initialslide_num is a number from 1 to the number of slides in the slide show and specifies which slide to display first. If omitted, it is assumed to be 1.

Repeat_logical is a logical value specifying whether to repeat or end the slide show after displaying the last slide. If repeat_logical is TRUE, the slide show repeats; if FALSE or omitted, the slide show ends.

Dialogtitle_text is text enclosed in quotation marks that specifies the title of the dialog boxes displayed during the slide show. If dialogtitle_text is omitted, it is assumed to be "Slide Show".

Allownav_logical is a logical value specifying whether to enable or disable navigational keys (arrow keys, PAGE UP, PAGE DOWN, and so on) or the mouse during the slide show. If allownav_logical is TRUE or omitted, you can press navigational keys or use the mouse to move between slides; if FALSE, all movement is controlled by the slide show document settings.

Allowcontrol_logical is a logical value specifying whether to enable or disable the Slide Show Options dialog box during the slide show. If allowcontrol_logical is TRUE or omitted, you can press ESC to interrupt the slide show and display the dialog box; if FALSE, pressing ESC stops the slide show but does not display the dialog box.

Tip If you want to display the last slide in a show but don't know its number, use SLIDE.GET(1) as the initialslide_num argument.

Remarks

SLIDE.SHOW returns the values shown in the following table:

Situation	Returned value
The slide show ends normally.	TRUE
You press the Cancel button when using the dialog-box form.	FALSE
The active document is not a slide show or is protected.	#N/A
You interrupt the slide show, and then stop it.	1

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.ADD

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and choosing the Add button in the Solver Parameters dialog box. Adds a constraint to the current problem. For an explanation of constraints, see "Remarks" later in this topic.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.ADD(cell_ref, relation, formula)

Cell_ref is a reference to a cell or range of cells on the active sheet and forms the left side of the constraint.

Relation specifies the arithmetic relationship between the left and right sides, or whether cell_ref must be an integer.

Relation	Arithmetic relationship
1	<=
2	=
3	>=
4	Int (cell_ref is an integer)

Formula is the right side of the constraint and will often be a single number, but it may be a formula (as text) or a reference to a range of cells.

- If relation is 4, cell_ref must be a subset of the references in the By Changing cells text box.
- If relation is 4, formula must be either "=integer" or "integer".
- Any cell reference in a formula must use the R1C1 reference style.
- If formula is a reference to a range of cells, the number of cells in the range usually matches the number of cells in cell_ref, although the shape of the areas need not be the same. For example, cell_ref could be a row and formula could refer to a column, as long as the number of cells is the same. Formula can also be a single reference, as in the following relationship: A1:A4 <= B1.

Remarks

- The SOLVER.ADD, SOLVER.CHANGE and SOLVER.DELETE functions correspond to the Add, Change, and Delete buttons in the Tools Solver Parameters dialog box. You use these functions to define constraints. For many macro applications, however, you may find it more convenient to load the problem specifications from the sheet in a single step using the SOLVER.LOAD function.
- Each constraint is uniquely identified by the combination of the cell reference on the left and the relationship (<=, =, or >=) between its left and right sides, or the cell reference may be defined as an integer only. This takes the place of selecting the appropriate constraint in the Tools Solver Parameters dialog box. You can manipulate the constraints with SOLVER.CHANGE or SOLVER.DELETE. The constraints in a Solver problem can refer to a maximum of 400 cells.

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.CHANGE

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and choosing the Change button in the Solver Parameters dialog box. Changes the right side of an existing constraint.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.CHANGE(cell_ref, relation, formula)

For an explanation of the arguments and constraints, see SOLVER.ADD.

Remarks

- If the combination of cell_ref and relation does not match any existing constraint, the function returns the value 4 and no action is taken.
- To change the cell_ref or relation of an existing constraint, use SOLVER.DELETE to delete the old constraint and then use SOLVER.ADD to add the constraint in the form you want.

Related Functions

[SOLVER.DELETE](#) Deletes an existing constraint

[SOLVER.ADD](#) Adds a constraint to the current problem

List of [Command-Equivalent Functions](#)

SOLVER.DELETE

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and choosing the Delete button in the Solver Parameters dialog box. Deletes an existing constraint.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.DELETE(cell_ref, relation, formula)

For an explanation of the arguments and constraints, see SOLVER.ADD.

Remarks

If the combination of cell_ref and relation does not match any existing constraint, the function returns the value 4 and no action is taken. If the constraint is found, it is deleted, and the function returns the value 0.

Related Functions

[SOLVER.ADD](#) Adds a constraint to the current problem

List of [Command-Equivalent Functions](#)

SOLVER.FINISH

Macro Sheets Only

Equivalent to choosing OK in the Solver Results dialog box that appears when the solution process is complete. The dialog-box form displays the dialog box with the arguments that you supply as defaults. This function must be used if you supplied the value TRUE for the userfinish argument to SOLVER.SOLVE.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.FINISH(keep_final, report_array)

SOLVER.FINISH?(keep_final, report_array)

Keep_final is the number 1 or 2 and specifies whether to keep the final solution. If keep_final is 1 or omitted, the final solution values are kept in the changing cells. If keep_final is 2, the final solution values are discarded and the former values of the changing cells are restored.

Report_array is an array argument specifying what reports to create when Solver is finished.

If report_array is	Microsoft Excel creates
--------------------	-------------------------

{1}	An answer report
{2}	A sensitivity report
{3}	A limit report

Any combination of these produces multiple reports. For example, if report_array is {1, 2}, Microsoft Excel creates an answer report and a sensitivity report.

Related Functions

[SOLVER.SOLVE](#) Equivalent to choosing the Solver command from the Tools menu and choosing the Solve button in the Solver Parameters dialog box

List of [Command-Equivalent Functions](#)

SOLVER.GET

Macro Sheets Only

Returns information about current settings for Solver. The settings are specified in the Solver Parameters and Solver Options dialog boxes.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.GET(type_num, sheet_name)

Type_num is a number specifying the type of information you want.

The following settings are specified in the Solver Parameters dialog box.

Type_Num	Returns
1	The reference in the Set Cell box, or the #N/A error value if Solver has not been used on the active sheet
2	A number corresponding to the Equal To option 1 = Max 2 = Min 3 = Value of
3	The value in the Value Of box
4	The reference (as a multiple reference if necessary) in the By Changing Cells box
5	The number of constraints
6	An array of the left sides of the constraints in the form of text
7	An array of numbers corresponding to the relationships between the left and right sides of the constraints: 1 = <= 2 = = 3 = >= 4 = int
8	An array of the right sides of the constraints in the form of text

The following settings are specified in the Solver Options dialog box:

Type_Num	Returns
9	The maximum calculation time
10	The maximum number of iterations
11	The precision
12	The integer tolerance value
13	TRUE if the Assume Linear Model check box is selected; FALSE otherwise
14	TRUE if the Show Iteration Results check box is selected; FALSE otherwise
15	TRUE if the Use Automatic Scaling check box is selected; FALSE otherwise
16	A number corresponding to the type of estimates: 1 = Tangent 2 = Quadratic
17	A number corresponding to the type of derivatives: 1 = Forward 2 = Central
18	A number corresponding to the type of search: 1 = Quasi-Newton 2 = Conjugate Gradient

Sheet_name is the name of a sheet that contains the scenario for which you want information. If sheet_name is omitted, it is assumed to be the active sheet.

Related Functions

List of [Information Functions](#)

SOLVER.LOAD

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu, choosing the Options button from the Solver Parameters dialog box, and choosing the Load Model button in the Solver Options dialog box.

Loads Solver problem specifications that you have previously saved on the worksheet.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.LOAD(load_area)

Load_area is a reference on the active sheet to a range of cells from which you want to load a complete problem specification.

- The first cell in load_area contains a formula for the Set Cell box; the second cell contains a formula for the changing cells; subsequent cells contain constraints in the form of logical formulas. The last cell optionally contains an array of Solver option values. The order of the Solver option values is the same as the top-to-bottom order in the Solver Options dialog box.
- Although load_area must be on the active sheet, it need not be the current selection.

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.OK

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and specifying options in the Solver Parameters dialog box. Specifies basic Solver options, except that constraints are added via SOLVER.ADD.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.OK(set_cell, max_min_val, value_of, **by_changing**)

SOLVER.OK?(set_cell, max_min_val, value_of, by_changing)

Set_cell corresponds to the Set Target Cell box in the Solver Parameters dialog box.

- Set_cell must be a reference to a cell on the active worksheet.
- If you enter a cell reference, you must also enter a value for max_min_val. If you do not enter a cell, you must include three commas before the by_changing value.

Max_min_val corresponds to the options Max, Min, and Value Of in the Solver Parameters dialog box. Use this option only if you entered a reference for set_cell.

Max_min_val	Option specified
1	Maximize
2	Minimize
3	Match specific value

Value_of is a number that becomes the target for the cell in the Set Target Cell box if max_min_val is 3. Value_of is ignored if the cell is being maximized or minimized.

By_changing indicates the changing cells, as entered in the By Changing Cells box. By_changing must refer to a cell or range of cells on the active worksheet, and can be a multiple selection.

Remarks

The constraints in a Solver problem can refer to a maximum of 400 cells.

Related Function

[SOLVER.SOLVE](#) Returns an integer value indicating the condition that caused Solver to stop

List of [Command-Equivalent Functions](#)

SOLVER.OPTIONS

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and then choosing the Options button in the Solver Parameters dialog box. Specifies the available options.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.OPTIONS(max_time, iterations, precision, assume_linear, step_thru, estimates, derivatives, search, int_tolerance, scaling)

The arguments correspond to the options in the dialog box. If an argument is omitted, Microsoft Excel uses an appropriate value based on the current situation. If any of the arguments are the wrong type, the function returns the #N/A error value. If an argument has the correct type but an invalid value, the function returns a positive integer corresponding to its position. A zero indicates all options were accepted.

Max_time must be an integer greater than zero and less than 32768. It corresponds to the Max Time box.

Iterations must be an integer greater than zero and less than 32768. It corresponds to the Iterations box.

Precision must be a number between zero and one, but not equal to zero or one. It corresponds to the Precision box.

Assume_linear is a logical value corresponding to the Assume Linear Model check box and allows Solver to arrive at a solution more quickly. If TRUE, Solver assumes that the underlying model is linear; if FALSE, it does not.

Step_thru is a logical value corresponding to the Show Iteration Results check box. If you have supplied SOLVER.SOLVE with a valid command macro reference, your macro will be called each time Solver pauses. If TRUE, Solver pauses at each trial solution; if FALSE, it does not.

Estimates is the number 1 or 2 and corresponds to the Estimates options: 1 for the Tangent option and 2 for the Quadratic option.

Derivatives is the number 1 or 2 and corresponds to the Derivatives options: 1 for the Forward option and 2 for the Central option.

Search is the number 1 or 2 and corresponds to the Search options: 1 for the Quasi-Newton option and 2 for the Conjugate Gradient option.

Int_tolerance is a decimal number corresponding to the Tolerance box in the Solver Options dialog box, and must be between zero and 1, inclusively. This argument applies only if integer constraints have been defined.

Scaling is a logical value corresponding to the Use Automatic Scaling check box. If scaling is TRUE, then if two or more constraints differ by several orders of magnitude, Solver scales the constraints to similar orders of magnitude during computation. If scaling is FALSE, Solver calculates normally.

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.RESET

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and choosing the Reset All button in the Solver Parameters dialog box. Erases all cell selections and constraints from the Solver Parameters dialog box and restores all the settings in the Solver Options dialog box to their defaults.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.RESET()

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.SAVE

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu, choosing the Options button from the Solver Parameters dialog box, and choosing the Save Model button in the Solver Options dialog box.

Saves the Solver problem specifications on the worksheet.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.SAVE (save_area)

Save_area is a reference on the active sheet to a range of cells or to the upper-left corner of a range of cells into which you want to paste the current problem specification.

- If you specify only one cell for save_area, the area is extended downwards for as many cells as are required to hold the problem specifications (3 plus the number of constraints).
- If you specify more than one cell and if the area is too small, the last constraints (in alphabetic order by cell reference) or options will be omitted and the function will return a nonzero value.
- Save_area must be on the active worksheet, but it need not be the current selection.

Related Functions

List of [Command-Equivalent Functions](#)

SOLVER.SOLVE

Macro Sheets Only

Equivalent to choosing the Solver command from the Tools menu and choosing the Solve button in the Solver Parameters dialog box. If successful, returns an integer value indicating the condition that caused Solver to stop as described in "Remarks" later in this topic.

If this function is not available, you must install the Solver add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

SOLVER.SOLVE(user_finish, show_ref)

User_finish is a logical value specifying whether to display the Solver Results dialog box.

- If user_finish is TRUE, SOLVER.SOLVE returns its integer value without displaying anything. Your macro should decide what action to take (for example, by examining the return value or presenting its own dialog box); it must call SOLVER.FINISH in any case to restore the sheet to its proper state.
- If user_finish is FALSE or omitted, Solver displays the Solver Results dialog box, which allows you to keep or discard the final solution and run reports.

Show_ref is a macro to be called in place of the Show Trial Solution dialog box. It is used when you want to regain control whenever Solver finds a new intermediate solution value.

- For this argument to have an effect, the Show Iteration Results check box must be selected in the Solver Options dialog box. This can be done manually by selecting the check box, or automatically by calling SOLVER.OPTIONS in your macro.
- The macro you call can inspect the current solution values on the sheet or take other actions such as saving or charting the intermediate values. It must return the value TRUE with a statement such as =RETURN(TRUE) if the solution process is to continue, or FALSE if the solution process should stop at this point.

Remarks

If a problem has not been completely defined, SOLVER.SOLVE returns the #N/A error value. Otherwise, the Solver application is started and the problem specifications are passed to it. When the solution process is complete, SOLVER.SOLVE returns an integer value indicating the stopping condition:

Value	Stopping condition
0	Solver found a solution. All constraints and optimality conditions are satisfied.
1	Solver has converged to the current solution. All constraints are satisfied.
2	Solver cannot improve the current solution. All constraints are satisfied.
3	Stop chosen when the maximum iteration limit was reached.
4	The Set Cells values do not converge.
5	Solver could not find a feasible solution.
6	Solver stopped at user's request.
7	The conditions for Assume Linear Model are not satisfied.
8	The problem is too large for Solver to solve.
9	Solver encountered an error value in a target or constraint cell.
10	Stop chosen when the maximum time limit was reached.
11	There is not enough memory available to solve the problem.
12	Another Excel instance is using SOLVER.DLL. Try again later.
13	Error in model. Please verify that all cells and constraints are valid.

Related Functions

[SOLVER.FINISH](#) Equivalent to choosing OK in the Solver Results dialog box that appears when the solution process is complete

List of [Command-Equivalent Functions](#)

SUBSCRIBE.TO

Macro Sheets Only

Inserts the contents of the edition into the active sheet at the point of the current selection. Use SUBSCRIBE.TO to incorporate editions published from other workbooks into your Microsoft Excel worksheets and macro sheets. SUBSCRIBE.TO returns TRUE if successful.

Syntax

SUBSCRIBE.TO(file_text, format_num)

Important This function is only available if you are using Microsoft Excel for the Macintosh with system software version 7.0 or later.

File_text is the name, as a text string, of the edition you want to insert into the active sheet. Unless file_text is in the current folder, supply the full path of the workbook. If file_text cannot be found, SUBSCRIBE.TO returns the #VALUE! error value and interrupts the macro.

Remarks

- If a single cell is selected, the data from the edition file is placed into as large a range of cells as is required by the data. Data already present in those cells is replaced. If the data is a picture, it is inserted from the upper-left corner of the selected cell.
- If a range of cells is selected, and the range is not big enough to contain the edition data, Microsoft Excel displays a dialog box asking if you want to clip the data to fit the range.

Format_num is the number 1 or 2 and specifies the format type of the file you are subscribing to.

Format_num	Format type
------------	-------------

1 or omitted	Picture
--------------	---------

2	Text (includes BIFF, VALU, TEXT, and CSV formats)
---	---

Related Functions

[CREATE.PUBLISHER](#)

Creates a publisher from the selection

[EDITION.OPTIONS](#)

Sets publisher and subscriber options

[GET.LINK.INFO](#)

Returns information about a link

List of [Command-Equivalent Functions](#)

TERMINATE

Macro Sheets Only

Closes a dynamic data exchange (DDE) channel previously opened with the INITIATE function. Use TERMINATE to close a channel after you have finished communicating with another application.

Syntax

TERMINATE(channel_num)

Important Microsoft Excel for the Macintosh requires system software version 7.0 or later for this function.

Channel_num is the number returned by a previously run INITIATE function. Channel_num identifies a DDE channel to close.

If TERMINATE is not successful, it returns the #VALUE! error value.

Related Functions

EXEC Starts another application
INITIATE Opens a channel to another application
List of DDE/External Functions

TEXTREF

Macro Sheets Only

Converts text to an absolute reference in either A1- or R1C1-style. Use TEXTREF to convert references stored as text to references so that you can use them with other functions, such as OFFSET.

Syntax

TEXTREF(text, a1)

Text is a reference in the form of text.

A1 is a logical value specifying the reference type of text. If a1 is TRUE, text is assumed to be an A1-style reference; if FALSE or omitted, text is assumed to be an R1C1-style reference.

Remarks

- If you use TEXTREF by itself in a cell, you will get the value contained in the cell specified by text, not the reference itself, because references are automatically converted into the contents of the referenced cell.
- If you use TEXTREF as a reference argument to a function, Microsoft Excel does not convert the reference to a value.

Tip You can convert a reference to text with REFTXT, manipulate it with the REPLACE and MID functions, and convert it back to a reference with TEXTREF.

Examples

TEXTREF("B7", TRUE) equals the reference value \$B\$7

TEXTREF("R5C5", FALSE) equals the reference value R5C5

TEXTREF("B7", FALSE) equals the #REF! error value, because "B7" can't be interpreted as an R1C1-style reference.

Related Functions

<u>DEREF</u>	Returns the values of the cells in a reference
<u>INDIRECT</u>	Returns a reference indicated by a text value
<u>MID</u>	Returns a specific number of characters from a text string starting at the position you specify
<u>OFFSET</u>	Returns a reference offset from a given reference
<u>REFTXT</u>	Converts a reference to text
<u>REPLACE</u>	Replaces characters within text
List of <u>Lookup & Reference Functions</u>	

UNDO

Macro Sheets Only

Equivalent to choosing the Undo command from the Edit menu. Reverses certain actions and commands. UNDO is available in the same situations as the Undo command.

Syntax

UNDO()

Related Functions

List of [Command-Equivalent Functions](#)

UNREGISTER

Macro Sheets Only

Unregisters a previously registered dynamic link library (DLL) or code resource. You can use UNREGISTER to free memory that was allocated to a DLL or code resource when it was registered. There are two syntax forms of this function. Use syntax 1 when you want Microsoft Excel to unregister a function or code resource according to its use count. Use syntax 2 when you want Microsoft Excel to unregister a function or code resource regardless of the use count.

Syntax 1

UNREGISTER(register_id)

Register_id is the register ID returned by the REGISTER or REGISTER.ID function, which corresponds to the function or code resource to be removed from memory.

Microsoft Excel counts the number of times you register a function or code resource. This number is called the use count. Each time you unregister a function or code resource, its use count is decremented by 1. When the use count equals 0, Microsoft Excel frees the allocated memory. Therefore, if you register a function or code resource more than once, you must use a corresponding number of UNREGISTER functions to ensure that it is completely unregistered.

Note Because Microsoft Excel for Windows and Microsoft Excel for the Macintosh use different types of code resources, UNREGISTER has a slightly different syntax form when used in each operating environment.

Syntax 2a

For Microsoft Excel for Windows

UNREGISTER(module_text)

Syntax 2b

For Microsoft Excel for the Macintosh

UNREGISTER(file_text)

Module_text or file_text is text specifying the name of the dynamic link library (DLL) that contains the function (in Microsoft Excel for Windows) or the name of the file that contains the code resource (in Microsoft Excel for the Macintosh).

If you use this syntax of UNREGISTER, all functions in the DLL (or all code resources in the file) are immediately unregistered, regardless of the use count.

Examples

Assuming that a REGISTER function in cell A5 of a macro sheet has already run (and has run only once), the following macro formula unregisters the corresponding function or code resource:

```
UNREGISTER(A5)
```

You could also use REGISTER.ID to return the register ID, instead of specifying a cell reference:

```
UNREGISTER(REGISTER.ID("User", "GetTickCount"))
```

Assuming that you have registered several different functions from the USER.EXE DLL of Microsoft Windows, the following macro formula unregisters all functions in that DLL:

```
UNREGISTER("User")
```

Tip If you register a function or code resource, and use the optional function_text argument to specify a custom name that will appear in the Paste Function dialog box, this custom name will not be removed by the UNREGISTER function. To remove the custom name, use the SET.NAME function without its second argument

Related Functions

<u>CALL</u>	Calls a procedure in a dynamic link library or code resource
<u>REGISTER</u>	Registers a code resource
<u>REGISTER.ID</u>	Returns the register ID of the resource

List of DDE/External Functions

VIEW.3D

Macro Sheets Only

Equivalent to choosing the 3-D View command from the Format menu in Microsoft Excel version 4.0, available when a chart is the active document. Adjusts the view of the active 3-D chart. Use VIEW.3D to emphasize different parts of your chart by viewing it from different angles and perspectives.

Syntax

VIEW.3D(elevation, perspective, rotation, axes, height%, autoscale)

VIEW.3D?(elevation, perspective, rotation, axes, height%, autoscale)

Elevation is a number from -90 to 90 specifying the viewing elevation of the chart and is measured in degrees. Elevation corresponds to the Elevation box in the 3-D View dialog box in Microsoft Excel version 4.0.

- If elevation is 0, you view the chart straight on. If elevation is 90, you view the chart from above (a "bird's eye view"). If elevation is -90, you view the chart from below.
- If elevation is omitted, it is assumed to be 25.
- Elevation is limited to 0 to 44 for 3-D bar charts and 0 to 80 for 3-D pie charts.

Perspective is a number from 0 to 100% specifying the perspective of the chart. Perspective corresponds to the Perspective box in the 3-D View dialog box in Microsoft Excel version 4.0.

- A higher perspective value simulates a closer view.
- If perspective is omitted, it is assumed to be 30.
- Perspective is ignored on 3-D bar and pie charts.

Rotation is a number from 0 to 360 specifying the rotation of the chart around the value (z) axis and is measured in degrees. Rotation corresponds to the Rotation box in the 3-D View dialog box in Microsoft Excel version 4.0. As you rotate the chart, the back and side walls are moved so that they do not block the chart.

- If rotation is omitted, it is assumed to be 30.
- Rotation is limited to 0 to 44 for 3-D bar charts.

Axes is a logical value specifying whether axes are fixed in the plane of the screen or can rotate with the chart. Axes corresponds to the Right Angle Axes check box in the 3-D View dialog box in Microsoft Excel version 4.0.

- If axes is TRUE, Microsoft Excel locks the axes.
- If axes is FALSE, Microsoft Excel allows the axes to rotate.
- If axes is omitted and the chart view is 3-D layout, axes is assumed to be FALSE.
- If axes is omitted and the chart view is not 3-D layout, axes is assumed to be TRUE.
- Axes is TRUE for 3-D bar charts and ignored for 3-D pie charts.

Height% is a number from 5 to 500 specifying the height of the chart as a percentage of the length of the base. Height% corresponds to the Height box in the 3-D View dialog box in Microsoft Excel version 4.0. Height% is useful for controlling the appearance of charts with many series or data points. If height% is omitted, it is assumed to be 100.

Autoscale is a logical value corresponding to the Auto Scaling check box in the 3-D View dialog box in Microsoft Excel version 4.0. If TRUE, automatic scaling is used; if FALSE, it is not; if omitted, the current setting is not changed.

Related Function

FORMAT.MAIN Formats a main chart

List of Command-Equivalent Functions

VIEW.DEFINE

Macro Sheets Only

Equivalent to choosing the Add button from the View Manager dialog box, which appears when you choose the View Manager command from the View menu. Creates or replaces a view.

If this function is not available, you must install the View Manager add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

VIEW.DEFINE(view_name, print_settings_log, row_col_log)

View_name is text enclosed in quotation marks and specifies the name of the view you want to define for the sheet. If the workbook already contains a view with view_name, the new view replaces the existing one.

Print_settings_log is a logical value that, if TRUE or omitted, includes current print settings in the view or, if FALSE, does not include current print settings in the view.

Row_col_log is a logical value that, if TRUE or omitted, includes current row and column settings in the view or, if FALSE, does not include current row and column settings in the view.

Related Functions

[VIEW.DELETE](#) Removes a view from the active workbook

[VIEW.SHOW](#) Shows a view

List of [Command-Equivalent Functions](#)

VIEW.DELETE

Macro Sheets Only

Equivalent to selecting a view and choosing the Delete button from the View Manager dialog box, which appears when you choose the View Manager command from the View menu. Removes a view from the active workbook.

If this function is not available, you must install the View Manager add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

VIEW.DELETE(view_name)

View_name is text enclosed in quotation marks and specifies the name of the view in the current workbook that you want to delete.

Remarks

VIEW.DELETE returns the #VALUE error value if view_name is invalid or if the workbook is protected.

Related Functions

[VIEW.DEFINE](#) Creates or replaces a view

[VIEW.SHOW](#) Shows a view

List of [Command-Equivalent Functions](#)

VIEW.GET

Macro Sheets Only

Equivalent to displaying a list of views in the View Manager dialog box, which appears when you choose the View Manager command from the View menu. Returns an array of views from the active workbook.

If this function is not available, you must install the View Manager add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

VIEW.GET(type_num, view_name)

Type_num is a number from 1 to 3 that specifies the type of information to return, as shown in the following table.

Type_num	Result
1	Returns an array of views from all the sheets in the active workbook or the #N/A error value if none are defined.
2	Returns TRUE if print settings are included in the specified view. Returns FALSE if print settings are not included. Returns the #VALUE! error value if the name is invalid or the workbook is protected.
3	Returns TRUE if row and column settings are included in the specified view. Returns FALSE if row and column settings are not included. Returns the #VALUE! error value if the name is invalid or the workbook is protected.

View_name is text enclosed in quotation marks and specifies the name of a view in the active workbook. View_name is required if type_num is 2 or 3.

Example

The following macro formula returns an array of views from the active workbook:

```
VIEW.GET(1)
```

Related Functions

- [VIEW.DEFINE](#) Creates or replaces a view
 - [VIEW.DELETE](#) Removes a view from the active workbook
 - [VIEW.SHOW](#) Shows a view
- List of [Information Functions](#)

VIEW.SHOW

Macro Sheets Only

Equivalent to selecting a view and choosing the Show button in the View Manager dialog box, which appears when you choose the View Manager command from the View menu. Shows a view.

If this function is not available, you must install the View Manager add-in macro. For more information, see "Installing Add-in Features" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

Syntax

VIEW.SHOW(view_name)

VIEW.SHOW?(view_name)

View_name is text enclosed in quotation marks and specifies the name of a view in the active workbook.

Remarks

VIEW.SHOW returns the #VALUE error value if view_name is invalid or the workbook is protected.

Related Functions

[VIEW.DEFINE](#) Creates or replaces a view

[VIEW.DELETE](#) Removes a view from the active workbook

List of [Command-Equivalent Functions](#)

SERIES

Charts Only

Represents a data series in the active chart. SERIES is used only in charts; you cannot enter it on a sheet or macro sheet. You normally create or change data series by using the Chart Wizard or EDIT.SERIES macro function, which is equivalent to the Edit Series command on the Chart menu in Microsoft Excel version 4.0. However, you can edit a data series manually by selecting it, switching to the formula bar, and typing the changes.

Syntax

SERIES(name_ref, categories, **values**, plot_order)

Name_ref is the name of the data series. It can be an external reference to a single cell or a name defined as a single cell. Name_ref can also be text enclosed in quotation marks (for example, "Projected Sales").

Categories is an external reference to the name of the workbook and to the cells that contain one of the following sets of data:

- Category labels for all charts except xy (scatter) charts
- X-coordinate data for xy (scatter) charts

Values is an external reference to the name of the workbook and to the cells that contain values (or y-coordinate data in scatter charts).

Plot_order is an integer specifying whether the series is plotted first, second, or third, and so on, in the chart. No two series can have the same plot_order.

Remarks

- Categories and values can be arrays or references to a multiple selection, although they cannot be names that refer to a multiple selection. If you specify a multiple selection for any of these arguments, make sure you include the necessary sets of parentheses so that Microsoft Excel does not treat the components of the references as separate arguments.
- If either categories or values is a multiple selection, then all areas in that selection must be either vertical (more rows than columns) or horizontal (more columns than rows).

Related Functions

<u>CHART.WIZARD</u>	Creates and formats a chart
<u>EDIT.SERIES</u>	Creates or changes a chart series

ADD.TOOL

Macro Sheets Only

Adds one or more buttons to a toolbar.

Syntax

ADD.TOOL(bar_id, position, tool_ref)

Bar_id is either a number specifying one of the built-in toolbars or the name of a custom toolbar.

Bar_id	Built-in toolbar
1	Standard
2	Formatting
3	Query and Pivot
4	Chart
5	Drawing
6	TipWizard
7	Forms
8	Stop Recording
9	Visual Basic
10	Auditing
11	WorkGroup
12	Microsoft
13	Full Screen

Position specifies the position of the button within the toolbar. Position starts with 1 at the left side (if horizontal) or at the top (if vertical).

Tool_ref is either a number specifying a built-in button or a reference to an area on the macro sheet that defines a custom button or set of buttons (or an array containing this information). For a complete list of built-in buttons and their corresponding numbers, see Appendix D, "Toolbar Buttons in Microsoft Excel," in the Microsoft Excel Visual Basic User's Guide.

For customized buttons, the following example shows the components of a button reference area on a macro sheet and defines custom tools. The range A1:I5 is a valid tool_ref. Row 1 refers to a built-in tool. Row 2 defines a gap. For this illustration, values are displayed instead of formulas so that text can wrap in cells.

	A	B	C	D	E	F	G	H	I
1	Tool_id	Macro	Down	Enabled	Face	Status_text	Ballon_text	Help_topic	Tip_text
2	14								
3	0								
4	220	ShiftCells	TRUE	TRUE	Rectangle1	Move Selected cells right one cell	Use the Shift Cells Right tool to move the selected cells to the right	Help!40	Shift Cells right
5	224	Multiply	TRUE	TRUE	Rectangle3	Multiply a column or row of numbers	Use the Multiply tool to multiply a column or row of numbers	Help!41	Multiply

- Tool_id is a number associated with the tool. A zero specifies a gap on the toolbar. To specify a custom button, use a name, or a number between 201 and 231. For a complete list of built-in buttons and their corresponding numbers, see Appendix D, "Toolbar Buttons in Microsoft Excel," in the Microsoft Excel Visual Basic User's Guide.
- Macro is the name of, or a quoted R1C1-style reference to, the macro you want to run when the

button is clicked.

- Down is a logical value specifying the default image of the tool. If down is TRUE, the button appears depressed into the screen; if FALSE or omitted, it appears normal (up).
- Enabled is a logical value specifying whether the button can be used. If enabled is TRUE, the button is enabled; if FALSE, it is disabled.
- Face specifies a face associated with the tool. Face must be a reference to a picture-type object, for example "Picture 1". If face is omitted, Microsoft Excel uses the default face for the tool.
- Status_text is the text, if any, that you want displayed in the status bar when the button is selected.
- Balloon_text is the balloon help text, if any, associated with the tool. Balloon_text is available only in Microsoft Excel for the Macintosh using system software version 7.0 or later.
- Help_topics is a reference to a topic in a Help file, in the form "filename!topic_number". Help_topics must be text. If help_topics is omitted, HELP displays the Contents topic for Microsoft Excel Help.
- Tip_text is the text, if any, that you want displayed as a ToolTip when the mouse pointer moves over a tool button.

To indicate that a particular component of tool_ref is not used, clear the contents of the corresponding cell.

Remarks

- If you do not want to reserve a section of your macro sheet to define the buttons, you can use an array as the tool_ref argument as shown in the following syntax:

ADD.TOOL(bar_id, position, {tool_id1, macro1, down1, enabled1, face1, help_text1, balloon_text1, help_topics1; tool_id2, macro2, down2, enabled2, face2, help_text2, balloon_text2, help_topics2;...})

- Picture objects can be created with the camera button or pasted in from another application. In Microsoft Excel for Windows, the graphic object must be either a Windows bitmap or picture object. In Microsoft Excel for the Macintosh, the object must be a picture object.

Examples

The following macro formula adds a button to Toolbar5. The cell range B6:I6 contains tool_ref.

```
ADD.TOOL("Toolbar5", 6, B6:I6)
```

The following macro formula adds the New Macro Sheet button to the fifth position on the Standard toolbar:

```
ADD.TOOL(1, 5, 6)
```

Related Functions

<u>ADD.COMMAND</u>	Adds a command to a menu
<u>ADD.TOOLBAR</u>	Creates a toolbar with the specified tools
<u>DELETE.TOOL</u>	Deletes a button from a toolbar
<u>DELETE.TOOLBAR</u>	Deletes custom toolbars
<u>RESET.TOOLBAR</u>	Resets a built-in toolbar to its default initial setting
List of <u>Customizing Functions</u>	

APP.SIZE

Macro Sheets Only

Equivalent to choosing the Size command from the Control menu for the application window. Changes the size of the Microsoft Excel window.

Syntax

APP.SIZE(x_num, y_num)

APP.SIZE?(x_num, y_num)

Note This function is only for Microsoft Excel for Windows. You can use this function in macros created with Microsoft Excel for the Macintosh, but it will return the #N/A error value.

X_num specifies the width of the Microsoft Excel window in points.

Y_num specifies the height of the Microsoft Excel window in points.

APP.SIZE?, the dialog-box form of the function, doesn't display a dialog box. Instead, it is equivalent to pressing ALT, SPACEBAR, S or to dragging a window border with the mouse. Using APP.SIZE?, you can size the window with the keyboard or mouse. If you specify x_num and/or y_num in the dialog-box form of the function, the window is sized according to the specified coordinates, and you are left in size mode.

Related Functions

<u>APP.ACTIVATE</u>	Switches to an application
<u>APP.MAXIMIZE</u>	Maximizes the Microsoft Excel application window
<u>APP.MINIMIZE</u>	Minimizes the Microsoft Excel application window
<u>APP.MOVE</u>	Moves the Microsoft Excel application window
<u>APP.RESTORE</u>	Restores the Microsoft Excel application window

List of Command-Equivalent Functions

CALCULATE.DOCUMENT

Macro Sheets Only

Equivalent to choosing the Calc Sheet button in the Calculation tab on the Options dialog, which appears when you choose the Options command from the Tools menu. Calculates only the active worksheet.

Syntax

CALCULATE.DOCUMENT()

Remarks

If a chart is active, CALCULATE.DOCUMENT returns the #VALUE! error value.

Related Functions

CALCULATE.NOW Calculates all open workbooks immediately

CALCULATION Controls calculation settings

List of Command-Equivalent Functions

CLEAR.ROUTING.SLIP

Equivalent to the Clear button in the Routing Slip dialog box. Clears the routing slip.

Syntax

CLEAR.ROUTING.SLIP(reset_only_logical)

Reset_only_logical is a logical value that specifies whether the routing slip should be cleared.

- This option is valid only after every recipient on the routing slip has received and forwarded the workbook. Setting reset_only_logical to TRUE in this case is equivalent to the Reset button in the routing slip dialog.
- If some recipients have not received or routed the workbook, reset_only_logical is ignored.
- If reset_only_logical is FALSE or omitted and the workbook has been routed to all recipients, then the routing slip is removed from the workbook. A new slip can be subsequently added using ROUTING.SLIP().

Related Functions

List of [Command-Equivalent Functions](#)

CONSTRAIN.NUMERIC

Macro Sheets Only

Equivalent to pressing the Constrain Numeric button in the Formula category of the Customize dialog box, which appears when you choose the Toolbars command from the View menu and then choose the customize button. Constrains handwriting recognition to numbers and punctuation only. Use this function in a macro to improve the accuracy of handwriting recognition when the user is entering a series of numbers or formulas.

Note This function is only available if you are using Microsoft Windows for Pen Computing.

Syntax

CONSTRAIN.NUMERIC(numeric_only)

Numeric_only is a logical value that turns the numeric constraint on or off. If numeric_only is TRUE, only numbers and digits are recognized; if FALSE, all characters are recognized as usual. if numeric_only is omitted, the numeric constraint is toggled.

Remarks

When the numeric constraint is on, Microsoft Excel recognizes only the following symbols:

0 1 2 3 4 5 6 7 8 9 \$ # @ % () - + = { } : < > , ? | .

Tip Use GET.WORKSPACE(45) to make sure you're running Microsoft Windows for Pen Computing.

Related Functions

List of Command-Equivalent Functions

COPY.CHART

Macro Sheets Only

Equivalent to choosing the Copy Chart command from the Edit menu in Microsoft Excel for the Macintosh version 1.5 or earlier. This function is included only for macro compatibility. You can copy a chart with the COPY.PICTURE function by omitting the appearance_num argument.

Syntax

COPY.CHART(size_num)

Size_num is a number describing how to copy the picture and is only available if the current selection is a chart.

Size_num	Action
1 or omitted	Copies the chart in the same size as the window on which it is displayed
2	Copies what you would see if you printed the chart

Related Function

[COPY.PICTURE](#) Creates a picture of the current selection for use in another program

List of [Command-Equivalent Functions](#)

COPY.PICTURE

Macro Sheets Only

Equivalent to choosing the Copy Picture command from the Edit menu. The Copy Picture command appears if you hold down SHIFT while choosing the Edit menu. It copies a chart or range of cells to the Clipboard as a graphic. Use COPY.PICTURE to create an image of the current selection or chart for use in another program.

Syntax

COPY.PICTURE(appearance_num, size_num, type_num)
COPY.PICTURE?(appearance_num, size_num, type_num)

Remarks

Graphics are created differently on screen and on a printer. Thus, the printed picture may look different from the one on screen.

Appearance_num is a number describing how to copy the picture.

Appearance_num	Action
1 or omitted	Copies a picture as closely as possible to the picture displayed on your screen
2	Copies what you would see if you printed the selection

Size_num is a number describing how to copy the picture and is only available if the current selection is a chart.

Size_num	Action
1 or omitted	Copies the chart in the same size as the window on which it is displayed
2	Copies what you would see if you printed the chart

Type_num is a number specifying the format of the picture. This argument is available only in Microsoft Excel for Windows.

Type_num	Format of the picture
1 or omitted	Picture
2	Bitmap

Related Functions

<u>COPY</u>	Copies and pastes data or objects
<u>CUT</u>	Cuts or moves data or objects
<u>PASTE</u>	Pastes cut or copied data
<u>PASTE.PICTURE.LINK</u>	Pastes a linked picture of the currently copied area
<u>PASTE.SPECIAL</u>	Pastes specific components of copied data
List of <u>Command-Equivalent Functions</u>	

CUSTOMIZE.TOOLBAR

Macro Sheets Only

Equivalent to choosing the Toolbars command from the View menu and choosing the Customize button. Displays the Customize Toolbar dialog box. This function has a dialog-box syntax only.

Syntax

CUSTOMIZE.TOOLBAR?(category)

Category is a number that specifies which category of tools you want displayed in the dialog box. If omitted, the previous setting is used.

Category	Category of tools
1	File
2	Edit
3	Formula
4	Formatting
5	Text Formatting
6	Drawing
7	Macro
8	Charting
9	Utility
10	Data
11	TipWizard
12	Auditing
13	Forms
14	Custom

Related Functions

ADD.TOOLBAR Creates a new toolbar with the specified tools

SHOW.TOOLBAR Hides or displays a toolbar

List of Customizing Functions

DATA.LABEL

Equivalent to choosing the Data Labels command on the Insert menu when a chart is active. Specifies label contents and position.

Syntax

DATA.LABEL(show_option, auto_text, show_key)

Show_option is a number that specifies what type of labels to display.

Show_option	Type displayed
1	none
2	Show value
3	Show percent
4	Show label
5	Show label and percent

Auto_text is a logical value that corresponds the Automatic Checkbox in the Data Labels dialog box. If TRUE, resets a chart's data labels back to their actual values. If FALSE, they are not reset. The Automatic Text checkbox appears only if the label has been selected and its value changed.

Show_key is a logical value that specified whether to show the legend key next to the label. If TRUE, displays the legend key. If FALSE or omitted, does not display the legend key.

Related Functions

List of Command-Equivalent Functions

DELETE.STYLE

Macro Sheets Only

Equivalent to choosing the Delete button from the Style dialog box, which appears when you choose the Style command from the Format menu. Deletes a style from a workbook. Cells formatted with the deleted style revert to the Normal style.

Syntax

DELETE.STYLE(style_text)

Style_text is the name of a style to be deleted. If style_text does not exist, DELETE.STYLE returns the #VALUE! error value and interrupts the macro.

Remarks

You can only delete styles from the active workbook. External references are not permitted as part of the style_text argument.

Related Functions

<u>APPLY.STYLE</u>	Applies a style to the selection
<u>DEFINE.STYLE</u>	Creates or changes a cell style
<u>MERGE.STYLES</u>	Merges styles from another document into the active document

List of Command-Equivalent Functions

FIND.FILE

Equivalent to choosing the Find File command from the File menu. Lets you search for files based on criteria such as author or creation date.

Syntax

FIND.FILE?()

Remarks

This function has a dialog-box form only.

Related Functions

List of Command-Equivalent Functions

FILTER

Filters lists of data one column at a time. Only one list can be filtered on any one sheet at a time.

Syntax

FILTER(field_num, criteria1, operation, criteria2)

FILTER?(field_num, criteria1, operation, criteria2)

Field_num is the number of the field that you want to filter. Fields are numbered from left to right starting with 1.

Criteria1 is a text string specifying criteria for filtering a list, such as ">2". If you want to include all items in the list, omit this argument.

Operation is a number that specifies how you want criteria2 used with criteria1:

Number	Operation Used
1 or omitted	AND
2	OR

Criteria2 is a text string specifying criteria for filtering a list, such as ">2". If you include this argument, operation is required.

Remarks

If you omit all arguments, FILTER toggles the display of filter arrows.

Related Functions

[FILTER.ADVANCED](#) Lets you set options for filtering a list

List of [Command-Equivalent Functions](#)

FILTER.ADVANCED

Equivalent to choosing the Advanced Filter command from the Filter submenu on the Data menu. Lets you set options for filtering a list.

Syntax

FILTER.ADVANCED(operation, list_ref, criteria_ref, copy_ref, unique)

FILTER.ADVANCED?(operation, list_ref, criteria_ref, copy_ref, unique)

Operation is a number specifying whether to copy the filter list to a new location. To filter a list without copying, use 1; to copy the filter list to a new location, use 2.

List_ref specifies the location of the list to be filtered. If operation is 1, then list_ref must be on the active sheet.

Criteria_ref is a reference to a range containing criteria for filtering the list. If omitted, uses "All" as the criteria.

Copy_ref is a reference on the active sheet where you want the filtered list copied. Ignored if operation is 1.

Unique is a logical value that specifies whether only unique records are displayed. To display only unique records, use TRUE. To display all records that match the criteria, use FALSE or omit this argument.

Related Functions

[FILTER](#) Filters lists of data one column at a time

List of [Command-Equivalent Functions](#)

FILTER.SHOW.ALL

Equivalent to choosing the Show All command from the Filter submenu on the Data menu. Displays all items in a filtered list.

Syntax

FILTER.SHOW.ALL()

Related Functions

List of Command-Equivalent Functions

FILL.GROUP

Macro Sheets Only

Equivalent to choosing the Across Worksheets command from the Fill submenu on the Edit menu.

Copies the contents of the active worksheet's selection to the same area on all other worksheets in the group. Use FILL.GROUP to fill a range of cells on all worksheets in a group at once.

Syntax

FILL.GROUP(type_num)

FILL.GROUP?(type_num)

Type_num is a number from 1 to 3 that corresponds to the choices in the Fill Group dialog box.

Type_num	Type of information filled
1	All
2	Contents
3	Formats

Related Functions

[NEW](#)

Creates a new workbook

[WORKBOOK.SELECT](#)

Selects one or more sheets in a workbook

List of [Command-Equivalent Functions](#)

FORMAT.CHART

Macro Sheets Only

Equivalent to choosing the Options button in the Chart Type dialog box, which is available when you choose the Chart Type command from the Format menu when a chart is active. Formats the chart according to the arguments you specify.

Syntax

FORMAT.CHART(layer_num, view, overlap, angle, gap_width, gap_depth, chart_depth, doughnut_size, axis_num, drop, hilo, up_down, series_line, labels, vary)

FORMAT.CHART?(layer_num, view, overlap, angle, gap_width, gap_depth, chart_depth, doughnut_size, axis_num, drop, hilo, up_down, series_line, labels, vary)

Several of the following arguments are logical values corresponding to check boxes in the Options tab of Format (chart type) Group dialog box. If an argument is TRUE, Microsoft Excel selects the corresponding check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the setting is unchanged.

Layer_num is a number specifying which chart you want to change.

View is a number specifying one of the subtypes in the Subtype tab of the Format (type) Group dialog box. The subtype varies depending on the type of chart.

Overlap is a number from -100 to 100 specifying how you want bars or columns to be positioned. It corresponds to the Overlap edit box in the Options tab on the Format Bar Group Dialog box, which appears when you choose the Bar Group from the Format menu. Overlap is ignored if type_num is not 2 or 3 (bar or column chart).

- If overlap is positive, it specifies the percentage of overlap you want for bars or columns. For example, 50 would cause one-half of a bar or column to be covered by an adjacent bar or column. A value of zero prevents bars or columns from overlapping.
- If overlap is negative, then bars or columns are separated by the specified percentage of the maximum available distance between any two bars or columns.
- If overlap is omitted, it is assumed to be 0 (bars or columns do not overlap), or it is unchanged if a value was previously set.

Angle is a number from 0 to 360 specifying the angle of the first pie or doughnut slice (in degrees) if the chart is a pie or doughnut chart. If angle is omitted, it is assumed to be 0, or it is unchanged if a value was previously set.

Gap_width is a number from 0 to 500 specifying the space between bar or column clusters as a percentage of the width of a bar or column. It corresponds to the Gap Width edit box in the Options tab on the Format Bar Group Dialog box, which appears when you choose the Bar Group from the Format menu.

- Gap_width is ignored if type_num is not 2, 3, 8, or 12 (bar or column chart).
- If Gap_width is omitted, it is assumed to be 50, or it is unchanged if a value was previously set.

The next two arguments are for 3-D charts only, and correspond to check boxes in the Options tab of Format (chart type) Group dialog box.

Gap_depth is a number from 0 to 500 specifying the depth of the gap in front of and behind a bar, column, area, or line as a percentage of the depth of the bar, column, area, or line.

- Gap_depth is ignored if the chart is a pie chart or if it is not a 3-D chart.
- If gap_depth is omitted and the chart is a 3-D chart, gap_depth is assumed to be 50, or it is unchanged if a value was previously set. If gap_depth is omitted and the view is side-by-side, stacked, or stacked 100%, gap_depth is assumed to be 0, or it is unchanged if a value was previously set.

Chart_depth is a number from 20 to 2000 specifying the visual depth of the chart as a percentage of the width of the chart.

- Chart_depth is ignored if the chart is not a 3-D chart.
- If Chart_depth is omitted, it is assumed to be 100, or it is unchanged if a value was previously set.

Doughnut_size specifies the size of the hole in a doughnut chart. Can be a value from 10% - 90%. Default is 50%.

Axis_num is a number specifying whether to plot the chart on the primary axis or the secondary axis.

Drop corresponds to the Drop Lines check box. Drop is available only for area and line charts.

Hilo corresponds to the Hi-Lo Lines check box. Hilo is available only for 2-D line charts.

The next four arguments are logical values corresponding to check boxes in the Options tab of the Format (chart type) Group dialog box. If an argument is TRUE, Microsoft Excel selects the corresponding check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the setting is unchanged.

Up_down corresponds to the Up/Down Bars check box. Up_down is available only for 2-D line charts.

Series_line corresponds to the Series Lines check box. Series_line is available only for 2D stacked bar and column charts.

Labels corresponds to the Radar Axis Labels check box. Labels is available only for radar charts.

Vary corresponds to the Vary Colors By Point check box. Vary applies only to charts with one data series and is not available for area charts.

Related Functions

[FORMAT.MAIN](#)

Formats a chart according to the arguments you specify

[FORMAT.OVERLAY](#)

Formats an overlay chart

List of [Command-Equivalent Functions](#)

FORMAT.FONT

Macro Sheets Only

Equivalent to choosing the Cells command from the Format menu, and then selecting Font tab from the Format Cells dialog box. This function is included for compatibility with Microsoft Excel 4.0. Use [FONT.PROPERTIES](#) to set various font properties. FORMAT.FONT has three syntax forms. Syntax 1 is for cells; syntax 2 is for text boxes and buttons; syntax 3 is used with all chart items (axes, labels, text, and so on).

Syntax 1

Cells

FORMAT.FONT(name_text, size_num, bold, italic, underline, strike, color, outline, shadow)

FORMAT.FONT?(name_text, size_num, bold, italic, underline, strike, color, outline, shadow)

Syntax 2

Text boxes and buttons on worksheets and macro sheets

FORMAT.FONT(name_text, size_num, bold, italic, underline, strike, color, outline, shadow, object_id_text, start_num, char_num)

FORMAT.FONT?(name_text, size_num, bold, italic, underline, strike, color, outline, shadow, object_id_text, start_num, char_num)

Syntax 3

Chart items including unattached chart text

FORMAT.FONT(color, backgd, apply, name_text, size_num, bold, italic, underline, strike, outline, shadow, object_id_text, start_num, char_num)

FORMAT.FONT?(color, backgd, apply, name_text, size_num, bold, italic, underline, strike, outline, shadow, object_id_text, start_num, char_num)

Arguments correspond to check boxes and list box items in the Font tab on the Format Cells dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the format is not changed.

Name_text is the name of the font as it appears in the Font tab. For example, Courier is a font name.

Size_num is the font size, in points.

Bold corresponds to the Bold item in the Font Style list box. Makes the selection bold, if applicable.

Italic corresponds to the Italic item in the Font Style list box. Makes the selection italic, if applicable.

Underline corresponds to the Underline check box.

Strike corresponds to the Strikethrough check box.

Color is a number from 0 to 56 corresponding to the colors in the Font tab; 0 corresponds to automatic color.

Outline corresponds to the Outline check box. Outline fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, this argument is ignored by Microsoft Excel for Windows.

Shadow corresponds to the Shadow check box. Shadow fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, this argument is ignored by Microsoft Excel for Windows.

Note For macro compatibility with Microsoft Excel for the Macintosh, the presence of the outline and shadow arguments do not prevent the macro from working on Microsoft Excel for Windows, nor does their absence prevent it from working on the Macintosh.

Object_id_text identifies the text box you want to format (for example, "Text 1", "Text 2", and so on). You can also use the object number alone without the text identifier. For compatibility with earlier versions of Microsoft Excel. This argument is ignored in Microsoft Excel 5.0.

Start_num specifies the first character to be formatted. If start_num is omitted, it is assumed to be 1 (the first character in the text box).

Char_num specifies how many characters to format. If char_num is omitted, Microsoft Excel formats all characters in the text box starting at start_num.

Backgd is a number from 1 to 3 specifying which type of background to apply to text in a chart.

Backgd	Type of background applied
1	Automatic
2	Transparent
3	Opaque

Apply corresponds to the Apply To All check box. This argument applies to data labels only.

Remarks

Some extended TrueType styles do not have corresponding arguments to FORMAT.FONT. To access an extended TrueType font style, append the style name to the font name in name_text. For example, the font Taipei can be formatted in an upside-down style by specifying "Taipei Upside-down" as the name_text argument. For more information about TrueType, see your Microsoft Windows documentation.

Related Functions

ALIGNMENT

Aligns or wraps text in cells

FONT.PROPERTIES

Sets various font attributes

FORMAT.NUMBER

Applies a number format to the selection

FORMAT.TEXT

Formats a worksheet text box or a chart text item

List of Command-Equivalent Functions

FORMAT.TEXT

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu, and then choosing the Alignment tab, when a text box or button is selected, or when a chart is made active. Formats the selected worksheet text box or button or any text item on a chart.

Syntax

FORMAT.TEXT(x_align, y_align, orient_num, auto_text, auto_size, show_key, show_value, add_indent)

FORMAT.TEXT?(x_align, y_align, orient_num, auto_text, auto_size, show_key, show_value, add_indent)

Arguments correspond to check boxes or options in the various tabs on Format Object dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box; if omitted, the current setting is used.

X_align is a number from 1 to 4 specifying the horizontal alignment of the text.

X_align	Horizontal alignment
1	Left
2	Center
3	Right
4	Justify

Y_align is a number from 1 to 4 specifying the vertical alignment of the text.

Y_align	Vertical alignment
1	Top
2	Center
3	Bottom
4	Justify

Orient_num is a number from 0 to 3 specifying the orientation of the text.

Orient_num	Text orientation
0	Horizontal
1	Vertical
2	Upward
3	Downward

Auto_text corresponds to the Automatic Text check box. If the selected text was created with the Data Labels command from the Insert menu and later edited, this option restores the original text. Auto_text is ignored for text boxes on worksheets and macro sheets.

Auto_size corresponds to the Automatic Size check box. If you have changed the size of the border around the selected text, this option restores the border to automatic size. Automatic size makes the border fit exactly around the text no matter how you change the text.

Show_key corresponds to the Show Legend Key Next to Label check box in the Format Data Labels dialog box. This argument applies only if the selected text is an attached data label on a chart.

Show_value corresponds to the Show Value option button in the Format Data Labels dialog box. This argument applies only if the selected text is an attached data label on a chart.

The following list summarizes which arguments apply to each type of text item.

Add_indent This argument is for only Far East versions of Microsoft Excel.

Text item	Arguments that apply
Worksheet text box or button	X_align, y_align, orient_num, auto_size
Attached data label	All arguments
Unattached text label	X_align, y_align, orient_num, auto_size
Tickmark label	Orient_num

Related Functions

CREATE.OBJECT Creates an object

FONT.PROPERTIES Applies a font to the selection

FORMULA Enters values into a cell or range or onto a chart

List of Command-Equivalent Functions

CHART.TREND

Equivalent to choosing Trendline from the Insert menu when a chart is active. A trendline can be added to only to the these chart types: bar, column, stacked column, scatter, line, and area.

Syntax

CHART.TREND(type, ord_per, forecast, backcast, intercept, equation, r_squared, name)

Type is the type of trend or regression.

Number	Type used
1	Linear
2	Logarithmic
3	Polynomial
4	Power
5	Exponential
6	Moving Average

Ord_per depends on type. If type is 3, then ord_per is the order of the polynomial. If type is 6, ord_per is the number of periods for the moving average. If type is neither 3 nor 6, then ord_per is ignored.

Forecast is the number of periods or units to extrapolate the trendline in the positive or forward direction. This argument is ignored for moving averages (type 6). The default is zero.

Backcast is a number specifying the number of periods or units to extrapolate the trendline in the negative or backward direction. This argument is ignored for moving averages (type 6). The default is zero.

Intercept is a number specifying the value of the the y-intercept of the trendline, if it is already known. If FALSE or omitted, Microsoft Excel will calculate the y-intercept . This argument is ignored for moving averages.

Equation is a logical value specifying whether the trend equation should be displayed on the chart. If TRUE, the equation will be displayed on the chart. If FALSE or omitted, the equation will not be displayed on the chart.

R_squared is a logical value specifying whether the r-squared equation should be displayed on the chart. If TRUE, the value will be displayed on the chart. If FALSE or omitted, the equation will not be displayed on the chart.

Name is a text string specifying the custom name of the trendline. Can also be a logical value. If TRUE or omitted, the automatic name will be used instead.

Remarks

- A trendline can not be added to a 3-D chart, a stacked chart, or an 100% chart.
- the linear model calculates the least squares fit for a line represented by the equation $y = mx + b$, where m is the sope and b is the intercept.
- The logarithmic model calculates the least squares fit through points using the equation $y = c \cdot \ln(x) + b$, where c and b are constants.
- The exponential model calculates the least squares fit through points using the following equation:

$$y = c * e^{(b*x)}$$

where c and b are constants.

- The polynomial model calculates the least squares fit through points using the following equation:

$$y = b + (c_1 * x) + (c_2 * x^2) + (c_3 * x^3) + \dots (c_6 * x^6)$$

where b, c1, c2, c3, etc. are constants.

- The power model calculates the least squares fit through points using the following equation:

$$y = cx^b$$

where b and c are constants.

Related Functions

CHART.WIZARD Equivalent to choosing the ChartWizard button on the standard or chart toolbar

List of Command-Equivalent Functions

FULL.SCREEN

Equivalent to choosing the Full Screen command from the View menu.

Syntax

FULL.SCREEN(logical)

Logical switches to fullscreen if TRUE or omitted; exits fullscreen mode if FALSE.

Related Functions

List of [Command-Equivalent Functions](#)

GET.CELL

Macro Sheets Only

Returns information about the formatting, location, or contents of a cell. Use GET.CELL in a macro whose behavior is determined by the status of a particular cell.

Syntax

GET.CELL(type_num, reference)

Type_num is a number that specifies what type of cell information you want. The following list shows the possible values of type_num and the corresponding results.

Type_num	Returns
1	Absolute reference of the upper-left cell in reference, as text in the current workspace reference style.
2	Row number of the top cell in reference.
3	Column number of the leftmost cell in reference.
4	Same as TYPE(reference).
5	Contents of reference.
6	Formula in reference, as text, in either A1 or R1C1 style depending on the workspace setting.
7	Number format of the cell, as text (for example, "m/d/yy" or "General").
8	Number indicating the cell's horizontal alignment: 1 = General 2 = Left 3 = Center 4 = Right 5 = Fill 6 = Justify 7 = Center across cells
9	Number indicating the left-border style assigned to the cell: 0 = No border 1 = Thin line 2 = Medium line 3 = Dashed line 4 = Dotted line 5 = Thick line 6 = Double line 7 = Hairline
10	Number indicating the right-border style assigned to the cell. See type_num 9 for descriptions of the numbers returned.
11	Number indicating the top-border style assigned to the cell. See type_num 9 for descriptions of the numbers returned.
12	Number indicating the bottom-border style assigned to the cell. See type_num 9 for descriptions of the numbers returned.
13	Number from 0 to 18, indicating the pattern of the selected cell as displayed in the Patterns tab of the Format Cells dialog box, which appears when you choose the Cells command from the Format menu. If no pattern is selected, returns 0.
14	If the cell is locked, returns TRUE; otherwise, returns FALSE.
15	If the cell's formula is hidden, returns TRUE; otherwise, returns FALSE.
16	A two-item horizontal array containing the width of the active cell and a logical value indicating whether the cell's width is set to change as the standard width changes (TRUE) or is a custom width (FALSE).
17	Row height of cell, in points.
18	Name of font, as text.
19	Size of font, in points.

- 20 If all the characters in the cell, or only the first character, are bold, returns TRUE; otherwise, returns FALSE.
- 21 If all the characters in the cell, or only the first character, are italic, returns TRUE; otherwise, returns FALSE.
- 22 If all the characters in the cell, or only the first character, are underlined, returns TRUE; otherwise, returns FALSE.
- 23 If all the characters in the cell, or only the first character, are struck through, returns TRUE; otherwise, returns FALSE.
- 24 Font color of the first character in the cell, as a number in the range 1 to 56. If font color is automatic, returns 0.
- 25 If all the characters in the cell, or only the first character, are outlined, returns TRUE; otherwise, returns FALSE. Outline font format is not supported by Microsoft Excel for Windows.
- 26 If all the characters in the cell, or only the first character, are shadowed, returns TRUE; otherwise, returns FALSE. Shadow font format is not supported by Microsoft Excel for Windows.
- 27 Number indicating whether a manual page break occurs at the cell:
0 = No break
1 = Row
2 = Column
3 = Both row and column
- 28 Row level (outline).
- 29 Column level (outline).
- 30 If the row containing the active cell is a summary row, returns TRUE; otherwise, returns FALSE.
- 31 If the column containing the active cell is a summary column, returns TRUE; otherwise, returns FALSE.
- 32 Name of the workbook and sheet containing the cell. If the window contains only a single sheet that has the same name as the workbook without its extension, returns only the name of the book, in the form BOOK1.XLS. Otherwise, returns the name of the sheet in the form "[Book1]Sheet1".
- 33 If the cell is formatted to wrap, returns TRUE; otherwise, returns FALSE.
- 34 Left-border color as a number in the range 1 to 56. If color is automatic, returns 0.
- 35 Right-border color as a number in the range 1 to 56. If color is automatic, returns 0.
- 36 Top-border color as a number in the range 1 to 56. If color is automatic, returns 0.
- 37 Bottom-border color as a number in the range 1 to 56. If color is automatic, returns 0.
- 38 Shade foreground color as a number in the range 1 to 56. If color is automatic, returns 0.
- 39 Shade background color as a number in the range 1 to 56. If color is automatic, returns 0.
- 40 Style of the cell, as text.
- 41 Returns the formula in the active cell without translating it (useful for international macro sheets).
- 42 The horizontal distance, measured in points, from the left edge of the active window to the left edge of the cell. May be a negative number if the window is scrolled beyond the cell.
- 43 The vertical distance, measured in points, from the top edge of the active window to the top edge of the cell. May be a negative number if the window is scrolled beyond the cell.
- 44 The horizontal distance, measured in points, from the left edge of the active window to the right edge of the cell. May be a negative number if the window is scrolled beyond the cell.
- 45 The vertical distance, measured in points, from the top edge of the active window to the bottom edge of the cell. May be a negative number if the window is scrolled beyond the cell.
- 46 If the cell contains a text note, returns TRUE; otherwise, returns FALSE.
- 47 If the cell contains a sound note, returns TRUE; otherwise, returns FALSE.
- 48 If the cells contains a formula, returns TRUE; if a constant, returns FALSE.

- 49 If the cell is part of an array, returns TRUE; otherwise, returns FALSE.
- 50 Number indicating the cell's vertical alignment:
 1 = Top
 2 = Center
 3 = Bottom
 4 = Justified
- 51 Number indicating the cell's vertical orientation:
 0 = Horizontal
 1 = Vertical
 2 = Upward
 3 = Downward
- 52 The cell prefix (or text alignment) character, or empty text ("") if the cell does not contain one.
- 53 Contents of the cell as it is currently displayed, as text, including any additional numbers or symbols resulting from the cell's formatting.
- 54 Returns the name of the PivotTable view containing the active cell.
- 55 Returns the position of a cell within the PivotTableView.
- 56 Returns the name of the field containing the active cell reference if inside a PivotTable view.
- 57 Returns TRUE if all the characters in the cell, or only the first character, are formatted with a superscript font; otherwise, returns FALSE.
- 58 Returns the font style as text of all the characters in the cell, or only the first character as displayed in the Font tab of the Format Cells dialog box: for example, "Bold Italic".
- 59 Returns the number for the underline style:
 1 = none
 2 = single
 3 = double
 4 = single accounting
 5 = double accounting
- 60 Returns TRUE if all the characters in the cell, or only the first character, are formatted with a subscript font; otherwise, it returns FALSE.
- 61 Returns the name of the PivotTable item for the active cell, as text.
- 62 Returns the name of the workbook and the current sheet in the form "[book1]sheet1".
- 63 Returns the fill (background) color of the cell.
- 64 Returns the pattern (foreground) color of the cell.
- 65 Returns TRUE if the Add Indent alignment option is on (Far East versions of Microsoft Excel only); otherwise, it returns FALSE.
- 66 Returns the book name of the workbook containing the cell in the form BOOK1.XLS.

Reference is a cell or a range of cells from which you want information.

- If reference is a range of cells, the cell in the upper-left corner of the first range in reference is used.
- If reference is omitted, the active cell is assumed.

Tip Use GET.CELL(17) to determine the height of a cell and GET.CELL(44) - GET.CELL(42) to determine the width.

Examples

The following macro formula returns TRUE if cell B4 on sheet Sheet1 is bold:

```
GET.CELL(20, Sheet1!$B$4)
```

You can use the information returned by GET.CELL to initiate an action. The following macro formula runs a custom function named BoldCell if the GET.CELL formula returns FALSE:

```
IF(GET.CELL(20, Sheet1!$B$4), , BoldCell())
```


Related Functions

<u>ABSREF</u>	Returns the absolute reference of a range of cells to another range
<u>ACTIVE.CELL</u>	Returns the reference of the active cell
<u>GET.FORMULA</u>	Returns the contents of a cell
<u>GET.NAME</u>	Returns the definition of a name
<u>GET.NOTE</u>	Returns characters from a note
<u>RELREF</u>	Returns a relative reference

List of [Information Functions](#)

GET.DOCUMENT

Macro Sheets Only

Returns information about a sheet in a workbook.

Syntax

GET.DOCUMENT(type_num, name_text)

Type_num is a number that specifies what type of information you want. The following lists show the possible values of type_num and the corresponding results.

Type_num	Returns
1	If there is more than one sheet in the workbook, returns the name of the worksheet, as text, in the form "[book1]sheet1". Otherwise, returns only the name of the workbook. The workbook name does not include the drive, directory or folder, or window number. It is usually best to use GET.DOCUMENT(76) and GET.DOCUMENT(88) to return the name of the active worksheet and the active workbook.
2	Path of the directory or folder containing name_text, as text. If the workbook name_text hasn't been saved yet, returns the #N/A error value.
3	Number indicating the type of sheet. If name_text is a sheet, then the return value is one of the following numbers. If name_text is a book, then the return value is always 5. If name_text is omitted, then the sheet type is returned. If the book has one sheet that is named the same as the book, then the sheet type is returned. 1 = Worksheet 2 = Chart 3 = Macro sheet 4 = Info window if active 5 = Reserved 6 = Module 7 = Dialog
4	If changes have been made to the sheet since it was last saved, returns TRUE; otherwise, returns FALSE.
5	If the sheet is read-only, returns TRUE; otherwise, returns FALSE.
6	If the sheet is password protected, returns TRUE; otherwise, returns FALSE.
7	If cells in a sheet, the contents of a sheet, or the series in a chart are protected, returns TRUE; otherwise, returns FALSE.
8	If the workbook windows are protected, returns TRUE; otherwise, returns FALSE.

The next four values of type_num apply only to charts.

Type_num	Returns
9	Number indicating the type of the main chart: 1 = Area 2 = Bar 3 = Column 4 = Line 5 = Pie 6 = XY (scatter) 7 = 3-D area 8 = 3-D column 9 = 3-D line 10 = 3-D pie 11 = Radar 12 = 3-D bar 13 = 3-D surface 14 = Donut
10	Number indicating the type of the overlay chart. Same as 1, 2, 3, 4, 5, 6, 11, and 14 for main chart above. If there is no overlay chart, returns the #N/A error value.
11	Number of series in the main chart.
12	Number of series in the overlay chart.

The next values of type_num apply to worksheets and macro sheets and to charts when appropriate.

Type_num	Returns
9	Number of the first used row. If the document is empty, returns 0.
10	Number of the last used row. If the document is empty, returns 0.
11	Number of the first used column. If the document is empty, returns 0.
12	Number of the last used column. If the document is empty, returns 0.
13	Number of windows.
14	Number indicating calculation mode: 1 = Automatic 2 = Automatic except tables 3 = Manual
15	If the Iteration check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
16	Maximum number of iterations.
17	Maximum change between iterations.
18	If the Update Remote References check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
19	If the Precision As Displayed check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
20	If the 1904 Date System check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.

Type_num values of 21 through 29 correspond to the four default fonts in previous versions of Microsoft Excel. These values are provided only for macro compatibility.

The next values of type_num apply to worksheets and macro sheets, and to charts if indicated.

Type_num	Returns
30	Horizontal array of consolidation references for the current sheet, in the form of text. If the list is empty, returns the #N/A error value.
31	Number from 1 to 11, indicating the function used in the current consolidation. The function that corresponds to each number is listed under the CONSOLIDATE function. The default function is SUM.
32	Three-item horizontal array indicating the status of the check boxes in the Data Consolidate dialog box. An item is TRUE if the check box is selected or FALSE if the check box is cleared. The first item indicates the Top Row check box, the second the Left Column check box, and the third the Create Links To Source Data check box.
33	If the Recalculate Before Saving check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
34	If the workbook is read-only recommended, returns TRUE; otherwise, returns FALSE.
35	If the workbook is write-reserved, returns TRUE; otherwise, returns FALSE.
36	If the document has a write-reservation password and it is opened with read/write permission, returns the name of the user who originally saved the file with the write-reservation password. If the file is opened as read-only, or if a password has not been added to the document, returns the name of the current user.
37	Number corresponding to the file type of the document as displayed in the Save As dialog box. See the <u>SAVE.AS</u> function for a list of all the file types that Microsoft Excel recognizes.
38	If the Summary Rows Below Detail check box is selected in the Outline dialog box, returns TRUE; otherwise, returns FALSE.
39	If the Summary Columns To Right Of Detail check box is selected in the Outline dialog box, returns TRUE; otherwise, returns FALSE.
40	If the Create Backup File check box is selected in the Save As dialog box, returns TRUE; otherwise, returns FALSE.

- 41 Number from 1 to 3 indicating whether objects are displayed:
 1 = All objects are displayed
 2 = Placeholders for pictures and charts
 3 = All objects are hidden
- 42 Horizontal array of all objects in the sheet. If there are no objects, returns the #N/A error value.
- 43 If the Save External Link Values check box is selected in the Calculation tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
- 44 If objects in a document are protected, returns TRUE; otherwise, returns FALSE.
- 45 A number from 0 to 3 indicating how windows are synchronized:
 0 = Not synchronized
 1 = Synchronized horizontally
 2 = Synchronized vertically
 3 = Synchronized horizontally and vertically
- 46 A seven-item horizontal array of print settings that can be set by the LINE.PRINT macro function:
- Setup text
 - Left margin
 - Right margin
 - Top margin
 - Bottom margin
 - Page length
 - A logical value indicating whether output will be formatted (TRUE) or unformatted (FALSE) when printed
- 47 If the Transition Expression Evaluation check box is selected in the Transition tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
- 48 The standard column width setting.

The next values of type_num correspond to printing and page settings.

Type_num	Returns
49	The starting page number, or the #N/A error value if none is specified or if "Auto" is entered in the First page Number text box on the Page tab of the Page Setup dialog box.
50	The total number of pages that would be printed based on current settings, excluding notes, or 1 if the document is a chart.
51	The total number of pages that would be printed if you print only notes, or the #N/A error value if the document is a chart.
52	Four-item horizontal array indicating the margin settings (left, right, top, bottom) in the currently specified units.
53	A number indicating the orientation: 1 = Portrait 2 = Landscape
54	The header as a text string, including formatting codes.
55	The footer as a text string, including formatting codes.
56	Horizontal array of two logical values corresponding to horizontal and vertical centering.
57	If row or column headings are to be printed, returns TRUE; otherwise, returns FALSE.
58	If gridlines are to be printed, returns TRUE; otherwise, returns FALSE.
59	If the sheet is printed in black and white only, returns TRUE; otherwise, returns FALSE.
60	A number from 1 to 3 indicating how the chart will be sized when it's printed: 1 = Size on screen 2 = Scale to fit page 3 = Use full page
61	A number indicating the pagination order:

1 = Down, then Over
 2 = Over, then Down
 Returns the #N/A error value if the document is a chart.

- 62 Percentage of reduction or enlargement, or 100% if none is specified. Returns the #N/A error value if not supported by the current printer or if the document is a chart.
- 63 A two-item horizontal array indicating the number of pages to which the printout should be scaled to fit, with the first item equal to the width (or #N/A if no width scaling is specified) and the second item equal to the height (or #N/A if no height scaling is specified). #N/A is also returned if the document is a chart.
- 64 An array of row numbers corresponding to rows that are immediately below a manual or automatic page break.
- 65 An array of column numbers corresponding to columns that are immediately to the right of a manual or automatic page break.

Note GET.DOCUMENT(62) and GET.DOCUMENT(63) are mutually exclusive. If one returns a value, then the other returns the #N/A error value.

The next values of type_num correspond to various document settings.

Type_num	Returns
66	In Microsoft Excel for Windows, if the Transition Formula Entry check box is selected in the Transition tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
67	Microsoft Excel 5.0 always returns TRUE here.
68	Microsoft Excel 5.0 always returns the book name.
69	Returns TRUE if Automatic Page Breaks is chosen in the View tab of the Options dialog box; otherwise, returns FALSE.
70	Returns the names of all the PivotTables in the document.
71	Returns an horizontal array of all the styles in a document.
72	Returns an horizontal array of all chart types displayed on the current sheet.
73	Returns an array of the number of series in each chart of the current sheet.
74	Returns the object id of the control that currently has the focus on a running user-defined dialog (based on the dialog sheet).
75	Returns the object id of the object that is the current default button on a running user-defined dialog (based on the dialog sheet).
76	Returns the name of the active sheet or macro sheet in the form [Book1]Sheet1.
77	Returns the paper size, as integer: 1 = Letter 8.5 x 11 in 2 = Letter Small 8.5 x 11 in 5 = Legal 8.5 x 14 in 9 = A4 210 x 297 mm 10 = A4 Small 210 x 297 mm 13 = B5 182 x 257 mm 18 = Note 8.5 x 11 in
78	Returns the print resolution, as a horizontal array of two numbers.
79	Returns TRUE if the Draft Quality check box has been selected from the sheet tab in the Page Setup dialog box; otherwise, returns FALSE.
80	Returns TRUE if the Notes checkbox has been selected on the Sheet tab in the Page Setup dialog box; otherwise, returns FALSE.
81	Returns the print area from the Sheet tab of the Page Setup dialog box as a cell reference.
82	Returns the print titles from the Sheet tab of the Page Setup dialog box as an array of cell references.
83	Returns TRUE if the worksheet is protected for scenarios; otherwise, returns FALSE.
84	Returns the value of the first circular reference on the sheet, or #N/A if there are no

circular references.

- 85 Returns the advanced filter mode state of the sheet. This is the mode without drop-down arrows on top. Returns TRUE if the list has been filtered by choosing Filter, then Advanced Filter from the Data menu. Otherwise, returns FALSE.
- 86 Returns the automatic filter mode state of the sheet. This is the mode with drop-down arrows on top. Returns TRUE if you have chosen Filter, then AutoFilter from the Data menu and the filter drop-down arrows are displayed. Otherwise, returns FALSE.
- 87 Returns the position number of the sheet. The first sheet is position 1. Hidden sheet are included in the count.
- 88 Returns the name of the active workbook in the form "Book1".

Name_text is the name of an open document. If name_text is omitted, it is assumed to be the active document.

Examples

The following macro formula returns TRUE if the contents of the active document are protected:

```
GET.DOCUMENT (7)
```

In Microsoft Excel for Windows, the following macro formula returns the number of windows in SALES.XLS:

```
GET.DOCUMENT (13, "SALES.XLS")
```

In Microsoft Excel for the Macintosh, the following macro formula returns 3 if the overlay chart on SALES CHART is a column chart:

```
GET.DOCUMENT (10, "SALES CHART")
```

To find out if SHEET1 is password-protected and if its contents and windows are protected, enter the following formula in a three-cell horizontal array:

```
GET.DOCUMENT ({6, 7, 8}, "SHEET1")
```

Related Functions

GET.CELL Returns information about the specified cell

GET.WINDOW Returns information about a window

GET.WORKSPACE Returns information about the workspace

List of Information Functions

GET.PIVOT.FIELD

Macro Sheets Only

Returns information about a field in a PivotTable.

Syntax

GET.PIVOT.FIELD(type_num, pivot_field_name, pivot_table_name)

Type_num is a value from 1 to 17 that returns the following types of information:

Type_num	Value
1	Returns an array of all the items which make up pivot_field_name. The array is made up of text constants, dates or numbers depending on the field.
2	Returns an array of all items which are set to show with the pivot_field_name. The array is made up of text constants, dates or numbers depending on the field. The array is returned in the order that the items are displayed in the PivotTable. If pivot_field_name is a page field, then the array contains only one element, the value corresponding to the active page (this could be all if the All item is showing).
3	Returns an array of all items which are hidden in the pivot_field_name. The array is made up of text constants, dates or numbers depending on the field. If pivot_field_name is a data field or the data header name, this function returns the #N/A! error value.
4	Returns an integer describing where the field is displayed in the active PivotTable (either row or column): 0 = Hidden 1 = Row 2 = Col 3 = Page 4 = Data
5	Returns an array of all items in pivot_field_name that are group parents. The array is made up of text constants, dates or numbers depending on the field. The array is returned in the order which these items appear in the PivotTable. Returns #N/A if there are no group parents and if the pivot_field_name is a data field or the data field header.
6	Returns a number between 0 and 4095 which describes the subtotals attached to the field. The number is the sum of the values associated with each subtotal function. See PIVOT.FIELD.PROPERTIES for a list of all the values associated with subtotal calculations. If the field is showing as a data field or data field header, #N/A! is returned.
7	Returns an integer describing the type of data contained in the field: 0 = Text 1 = Number 2 = Date
8	Returns an array five columns wide and one row high describing the summary function's custom calculation shown with the specified field (Data field) in the PivotTable. The array will look as follows: {function, calculation, base Field, base item, number format}. If pivot_field_name is not showing in the active PivotTable as a data field, #N/A! is returned.
9	Returns a reference to all of pivot_field_name's items currently showing in the active PivotTable. If pivot_field_name is hidden, #N/A! is returned. If pivot_field_name is a page field, the reference to the currently showing page item is returned. If pivot_field_name is a data field, a reference to all the data for this field in the PivotTable body is returned. The references are returned as text.
10	Returns a reference to the header cell for pivot_field_name. If pivot_field_name is a data field, a reference to all the headers in the data row or column is returned. If pivot_field_name is hidden, #N/A! is returned. The reference is returned as text.
11	Returns the number of grouped fields in the grouped field set which includes pivot_field_name. If pivot_field_name is neither a parent field nor a child field, 1 is returned.
12	Returns the level of pivot_field_name in the grouped field set which includes pivot_field_name. Returns 1 for the highest level parent field, 2 for its child field, and so on. If pivot_field_name is neither a parent field nor a child field, 1 is returned.

- 13 Returns the name of the parent field for pivot_field_name as a text constant. If pivot_field_name is not a child field, #N/A! is returned.
- 14 Returns the name of the child field for pivot_field_name as a text constant. If pivot_field_name is not a parent field, #N/A! is returned.
- 15 Returns a text constant representing the original name of the field in the data source.
- 16 Returns the position of the field among all the other fields in its orientation. For instance, a 1 would be returned if the field was the first row field.
- 17 Returns an array of all items in pivot_field_name that are group children. The array is made up of text constants, dates or numbers depending on the field. The array is returned in the order which these items appear in the PivotTable. Returns #N/A if there are no group children, and if the Pivot_field_name is a data field or the data field header.

Pivot_field_name is the name of the field that you want information about. If there is no field named Pivot_field_name in the PivotTable, returns #VALUE!.

Pivot_table_name is the name of a PivotTable containing the field that you want information about. If omitted, the PivotTable containing the active cell is used. If the active cell is not in a PivotTable, the #VALUE! error value is returned.

Related Functions

GET.PIVOT.ITEM Returns information about an item in a PivotTable.

GET.PIVOT.TABLE Returns information about a PivotTable.

List of Information Functions

GET.PIVOT.ITEM

Macro Sheets Only

Returns information about an item in a PivotTable.

Syntax

GET.PIVOT.ITEM(type_num, pivot_item_name, pivot_field_name, pivot_table_name)

Type_num is a value from 1 to 9 that represents the type of information you want about an item in a PivotTable.

Type_num	Information
1	Returns the position of the item in its field. Returns #N/A if pivot_field_name is a data field. Returns #N/A! if the item is hidden.
2	Returns the reference to all the cells in the PivotTable header currently containing pivot_item_name. This reference is returned as text. If pivot_item_name is currently not showing in the PivotTable, #N/A! is returned.
3	Returns the reference to all the data in the PivotTable body which is qualified by pivot_item_name. This reference is returned as text. If pivot_item_name is currently not showing in the PivotTable, #N/A! is returned.
4	Returns an array of text constants representing the children of pivot_item_name if pivot_item_name is a parent. Otherwise the function returns #N/A!.
5	Returns a text constant representing the parent of pivot_item_name, if pivot_item_name exists as part of a group. Otherwise the function returns #N/A!.
6	Returns TRUE if pivot_item_name is a member of a group which is currently expanded to show detail. Returns FALSE if pivot_item_name is a member of a group currently collapsed to hide detail. If item_name is not a member of a group, the function returns #N/A!.
7	Returns TRUE if pivot_item_name is expanded to show detail. Returns FALSE if pivot_item_name is collapsed to hide detail.
8	Returns TRUE if the item pivot_item_name is currently visible, FALSE if it is hidden.
9	Returns the name of the item as it appeared in the original data source. This will differ from the current item name only if the user changes the name of the item after creating the PivotTable.

Pivot_item_name is the name of the item that you want information about. If there is no item named pivot_item_name in the PivotTable, returns #VALUE!.

Pivot_field_name is the name of the field that you want information about. If there is no field named pivot_field_name in the PivotTable, returns #VALUE!.

Pivot_table_name is the name of a PivotTable containing the field that you want information about. If omitted, uses the PivotTable containing the active cell. If the active cell is not in a PivotTable, the #VALUE! error value is returned.

Related Functions

GET.PIVOT.FIELD Returns information about an item in a PivotTable.

GET.PIVOT.TABLE Returns information about a PivotTable.

List of Information Functions

GET.PIVOT.TABLE

Macro Sheets Only

Returns information about a PivotTable.

Syntax

GET.PIVOT.TABLE(type_num,pivot_table_name)

Type_num is a value from 1 to 21 that represents a type of information you want about a PivotTable.

Type_num	Information
1	Returns the name of the person who last updated the PivotTable, as a text constant.
2	Returns the date the PivotTable was last updated, as a serial number.
3	Returns a horizontal array of text constants representing all the fields in the PivotTable.
4	Returns an integer representing the number of fields in the PivotTable.
5	Returns a horizontal array of text constants representing all the visible fields in the PivotTable (rows, columns, pages or data)
6	Returns a horizontal array of text constants representing all the hidden fields in the PivotTable. Return #N/A if no hidden fields.
7	Returns a horizontal array of text constants representing the names of all the fields currently showing in the PivotTable as row fields. Returns #N/A if there are no row fields.
8	Returns a horizontal array of text constants representing all the fields currently showing in the PivotTable as column fields. Returns #N/A if no column fields exist.
9	Returns a horizontal array of text constants representing all the fields currently showing in the PivotTable as page fields. Return #N/A if no page fields exist.
10	Returns a horizontal array of text constants representing all the fields currently showing in the PivotTable as data fields. Returns #N/A if there are no data fields.
11	Returns the smallest rectangular reference which bounds the PivotTable and all headers (not including the page header). This reference is returned as text.
12	Returns the smallest rectangular reference which bounds the PivotTable and all headers (including the page headers). This reference is returned as text.
13	Returns the reference to the row header area as text. The row header area includes each row field header along with all the items in each row field. Returns #N/A if there are no row headers.
14	Returns the reference to the column header area as text. The column header area includes each column field header along with all the items in each column field. Returns #N/A if there are no column headers.
15	Returns the reference to the data header area as text. The data header area includes the data field header along with all the headers in the data row/col. Returns #N/A if there is no data field.
16	Returns a reference to all the page headers as text.
17	Returns the reference to the PivotTable data area as text.
18	Returns TRUE if the PivotTable is set to show row grand totals.
19	Returns TRUE if the PivotTable is set to show column grand totals.
20	Returns TRUE if the user is saving data with the PivotTable.
21	Returns TRUE if the PivotTable is set up to Autoformat on pivoting.
22	Returns the data source of the PivotTable. The kind of information returned depends on the data source: If the data source is a Microsoft Excel list or database, the cell reference is returned as text. If the data source is an external data source, then an array is returned. Each row consists of a SQL connection string with the remaining elements as the query string broken down into 200 character segments. If the data source is Multiple Consolidation ranges, then a two dimensional array is returned, each row of which consists of a reference and associated page field items. If the data source is another PivotTable, then one of the above three kinds of information

is returned.

Pivot_table_name is the name of a PivotTable containing the field that you want information about. If omitted, uses the PivotTable containing the active cell.

Remarks

Returns #VALUE! error value when pivot_table_name is not a valid PivotTable name on the active sheet and the active cell is not within a PivotTable.

Related Functions

GET.PIVOT.FIELD Returns information about an item in a PivotTable.

GET.PIVOT.ITEM Returns information about a PivotTable.

List of Information Functions

GET.WINDOW

Macro Sheets Only

Returns information about a window. Use GET.WINDOW in a macro that requires the status of a window, such as its name, size, position, and display options.

Syntax

GET.WINDOW(type_num, window_text)

Type_num is a number that specifies what type of window information you want. The following list shows the possible values of type_num and the corresponding results:

type_num	Returns
1	Name of the workbook and sheet in the window as text. For compatibility with Microsoft Excel 4.0, if the window contains only a single sheet that has the same name as the workbook without its extension, returns only the name of the book. Otherwise, returns the name of the sheet in the form "[Book1]Sheet1".
2	Number of the window.
3	X position, measured in points from the left edge of the workspace (in Microsoft Excel for Windows) or screen (in Microsoft Excel for the Macintosh) to the left edge of the window.
4	Y position, measured in points from the bottom edge of the formula bar to the top edge of the window.
5	Width, measured in points.
6	Height, measured in points.
7	If window is hidden, returns TRUE; otherwise, returns FALSE.

The rest of the values for type_num apply only to worksheets and macro sheets, except where indicated:

type_num	Returns
8	If formulas are displayed, returns TRUE; otherwise, returns FALSE.
9	If gridlines are displayed, returns TRUE; otherwise, returns FALSE.
10	If row and column headings are displayed, returns TRUE; otherwise, returns FALSE.
11	If zeros are displayed, returns TRUE; otherwise, returns FALSE.
12	Gridline and heading color as a number in the range 1 to 56, corresponding to the colors in the View tab of the Options dialog box; if color is automatic, returns 0.

Values 13 to 16 for type_num return arrays that specify which rows or columns are at the top and left edges of the panes in the window and the widths and heights of those panes.

- The first number in the array corresponds to the first pane, the second number to the second pane, and so on.
- If the edge of the pane occurs at the boundary between rows or columns, the number returned is an integer.
- If the edge of the pane occurs within a row or column, the number returned has a fractional part that represents the fraction of the row or column visible within the pane.
- The numbers can be used as arguments to the SPLIT function to split a window at specific locations.

type_num	Returns
13	Leftmost column number of each pane, in a horizontal numeric array
14	Top row number of each pane, in a horizontal numeric array.
15	Number of columns in each pane, in a horizontal numeric array.
16	Number of rows in each pane, in a horizontal numeric array.
17	Number indicating the active pane: 1 = Upper, left, or upper-left 2 = Right or upper-right 3 = Lower or lower-left 4 = Lower-right
18	If window has a vertical split, returns TRUE; otherwise, returns FALSE.

- 19 If window has a horizontal split, returns TRUE; otherwise, returns FALSE.
- 20 If window is maximized, returns TRUE; otherwise, returns FALSE.
- 21 Reserved
- 22 If the Outline Symbols check box is selected in the View tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
- 23 Number indicating the size of the window (including charts):
 1 = Restored
 2 = Minimized (displayed as an icon)
 3 = Maximized
- 24 If panes are frozen on the active window, returns TRUE; otherwise, returns FALSE.
- 25 The numeric magnification of the active window (as a percentage of normal size) as set in the Zoom dialog box, or 100 if none is specified.
- 26 Returns TRUE if horizontal scrollbars are displayed in the active window; otherwise, returns FALSE.
- 27 Returns TRUE if vertical scrollbars are displayed in the active window; otherwise, returns FALSE.
- 28 Returns the tab ratio of workbook tabs to horizontal scrollbar, from 0 to 1. The default is .6.
- 29 Returns TRUE if workbook tabs are displayed in the active window; otherwise, returns FALSE.
- 30 Returns the title of the active sheet in the window in the form "[book1]sheet1".
- 31 Returns the name of a workbook only, without read/write indicated. For example, if Book1.xls is read only, then "Book.xls" will be returned without "[Read Only]" appended.

Window_text is the name that appears in the title bar of the window that you want information about. If window_text is omitted, it is assumed to be the active window.

Examples

If the active window contains the sheet book Book1, then:

`GET.WINDOW(1)` equals "Book1"

If the title of the active window is Macro1:3, then:

`GET.WINDOW(2)` equals 3

In Microsoft Excel for Windows, the following macro formula returns the gridline and heading color of REPORT.XLS:

`GET.WINDOW(12, "REPORT.XLS")`

In Microsoft Excel for the Macintosh, the following macro formula returns the gridline and heading color of REPORT MASTER:

`GET.WINDOW(12, "REPORT MASTER")`

Related Functions

[GET.DOCUMENT](#) Returns information about a document

[GET.WORKSPACE](#) Returns information about the workspace

List of [Information Functions](#)

GET.WORKBOOK

Macro Sheets Only

Returns information about a workbook.

Syntax

GET.WORKBOOK(type_num, name_text)

Type_num is a number that specifies what type of workbook information you want.

Type_num	Returns
1	The names of all sheets in the workbook, as a horizontal array of text values.
2	This will always return the #N/A error value.
3	The names of the currently selected sheets in the workbook, as a horizontal array of text values.
4	The number of sheets in the workbook.
5	TRUE if the workbook has a routing slip; otherwise, FALSE.
6	The names of all of the document routing recipients who have not received the document, as a horizontal array of text values.
7	The subject line for the current routing slip, as text.
8	The message text for the routing slip, as text.
9	If the document is to be routed to recipients one after another, returns 1. If it is to be routed all at once, returns 2.
10	TRUE, if the Return When Done check box in the Routing Slip dialog box is selected; otherwise, FALSE.
11	TRUE, if the current recipient has already forwarded the current document; otherwise, FALSE.
12	TRUE, if the Track Status checkbox in the Routing Slip dialog box is selected; otherwise, FALSE.
13	Status of the workbook routing slip: 0 = unrouted 1 = routing in progress, or the workbook has been routed to a user 2 = routing is finished
14	TRUE, if the workbook structure is protected; otherwise, FALSE.
15	TRUE, if the workbook windows are protected; otherwise, FALSE.
16	Name of the workbook as text. The workbook name does not include the drive, directory or folder, or window number. This is the equivalent of GET.DOCUMENT(1).
17	TRUE if the document is read only; otherwise, FALSE. This is the equivalent of GET.DOCUMENT(34).
18	TRUE if sheet is write-reserved; otherwise, FALSE. This is the equivalent of GET.DOCUMENT(35).
19	Name of the user with current write permission for the document. This is the equivalent of GET.DOCUMENT(36).
20	Number corresponding to the file type of the document as displayed in the Save As dialog box. This is the equivalent of GET.DOCUMENT(37).
21	TRUE if the Create Backup File check box is selected in the Save As dialog box; otherwise, FALSE. This is the equivalent of GET.DOCUMENT(40).
22	TRUE if the Save External Link Values check box is selected in the Calculation tab of the Options dialog box. This is the equivalent of GET.DOCUMENT(43).
23	TRUE if the workbook has a PowerTalk mailer; otherwise, FALSE. Returns #N/A if no OCE mailer is installed.
24	TRUE if changes have been made to the workbook since the last time it was saved; FALSE if book is unchanged (or when closed, will not prompt to be saved).
25	The recipients on the To line of a PowerTalk mailer, as a horizontal array of text.
26	The recipients on the Cc line of a PowerTalk mailer, as a horizontal array of text.

- 27 The recipients on the Bcc line of a PowerTalk mailer, as a horizontal array of text.
- 28 The subject of the PowerTalk mailer, as text.
- 29 The enclosures of the PowerTalk mailer, as a horizontal array of text.
- 30 TRUE, if the PowerTalk mailer has been received from another user (as opposed to just being added but not sent). FALSE, if the mailer has not been received from another user.
- 31 The date and time the PowerTalk mailer was sent, as a serial number. Returns the #N/A error value if the mailer has not yet been sent.
- 32 The sender name of the PowerTalk mailer, as text. Returns the #N/A error value if the mailer has not yet been sent.
- 33 The title of the document as displayed in the Summary Info dialog box, as text.
- 34 The subject of the document as displayed in the Summary Info dialog box, as text.
- 35 The author of the document as displayed in the Summary Info dialog box, as text.
- 36 The keywords for the document as displayed in the Summary Info dialog box, as text.
- 37 The comments for the document as displayed in the Summary Info dialog box, as text.
- 38 The name of the active worksheet.

Name_text is the name of an open workbook. If name_text is omitted, it is assumed to be the active workbook.

Example

The following macro formula returns the name of the active sheet in the workbook named SALES.XLS:

```
GET.WORKBOOK (2, "SALES.XLS")
```

Related Functions

GET.DOCUMENT Returns information about a document

WORKBOOK.SELECT Selects the specified documents in a workbook

List of Information Functions

GET.WORKSPACE

Macro Sheets Only

Returns information about the workspace. Use GET.WORKSPACE in a macro that depends on the status of the workspace, such as the environment, version number, and available memory.

Syntax

GET.WORKSPACE(type_num)

Type_num is a number specifying the type of workspace information you want. The following list shows the type_num values and their corresponding results.

Type_num	Returns
1	Name of the environment in which Microsoft Excel is running, as text, followed by the environment's version number.
2	The version number of Microsoft Excel, as text (for example, "5.0").
3	If fixed decimals are set, returns the number of decimals; otherwise, returns 0.
4	If in R1C1 mode, returns TRUE; if in A1 mode, returns FALSE.
5	If scroll bars are displayed, returns TRUE; otherwise, returns FALSE. See also GET.WINDOW(26) and GET.WINDOW(27).
6	If the status bar is displayed, returns TRUE; otherwise, returns FALSE.
7	If the formula bar is displayed, returns TRUE; otherwise, returns FALSE.
8	If remote DDE requests are enabled, returns TRUE; otherwise, returns FALSE.
9	Returns the alternate menu key as text; if no alternate menu key is set, returns the #N/A error value.
10	Number indicating special modes: 1 = Data Find 2 = Copy 3 = Cut 4 = Data Entry 5 = Unused 6 = Copy and Data Entry 7 = Cut and Data Entry If no special mode is set, returns 0.
11	X position of the Microsoft Excel workspace window, measured in points from the left edge of the screen to the left edge of the window. In Microsoft Excel for the Macintosh, always returns 0.
12	Y position of the Microsoft Excel workspace window, measured in points from the top edge of the screen to the top edge of the window. In Microsoft Excel for the Macintosh, always returns 0.
13	Usable workspace width, in points.
14	Usable workspace height, in points.
15	Number indicating maximized or minimized status of Microsoft Excel: 1 = Neither 2 = Minimized 3 = Maximized Microsoft Excel for the Macintosh always returns 3.
16	Amount of memory free (in kilobytes).
17	Total memory available to Microsoft Excel (in kilobytes).
18	If a math coprocessor is present, returns TRUE; otherwise, returns FALSE.
19	If a mouse is present, returns TRUE; otherwise, returns FALSE. In Microsoft Excel for the Macintosh, always returns TRUE.
20	If a group is present in the workspace, returns a horizontal array of sheets in the group; otherwise returns the #N/A error value.
21	If the Standard toolbar is displayed, returns TRUE; otherwise, returns FALSE.
22	DDE-application-specific error code.

- 23 Full path of the default startup directory or folder.
- 24 Full path of the alternate startup directory or folder; returns the #N/A error value if no alternate path has been specified.
- 25 If Microsoft Excel is set for relative recording, returns TRUE; if set for absolute recording, returns FALSE.
- 26 Name of user.
- 27 Name of organization.
- 28 If Microsoft Excel menus are switched to by the transition menu or help key, returns 1; if Lotus 1-2-3 Help is switched to, returns 2.
- 29 If transition navigation keys are enabled, returns TRUE.
- 30 A nine-item horizontal array of global (default) print settings that can be set by the LINE.PRINT function:
- Setup text
 - Left margin
 - Right margin
 - Top margin
 - Bottom margin
 - Page length
 - Logical value indicating whether to wait after printing each page (TRUE) or use continuous form feeding (FALSE)
 - Logical value indicating whether the printer has automatic line feeding (TRUE) or requires line feed characters (FALSE)
 - The number of the printer port
- 31 If a currently running macro is in single step mode, returns TRUE; otherwise, returns FALSE.
- 32 The current location of Microsoft Excel as a complete path.
- 33 A horizontal array of the names in the New list, in the order they appear.
- 34 A horizontal array of template files (with complete paths) in the New list, in the order they appear (returns the names of custom template files and the #N/A error value for built-in document types).
- 35 If a macro is paused, returns TRUE; FALSE otherwise.
- 36 If the Allow Cell Drag And Drop check box is selected in the Edit tab of the Options dialog box that appears when you choose the Options command from the Tools menu, returns TRUE; otherwise, returns FALSE.
- 37 A 45-item horizontal array of the items related to country versions and settings. Use the following macro formula to return a specific item, where number is a number in the list below:
- ```
INDEX(GET.WORKSPACE(37), number)
```
- These values apply to country codes:
- 1 Number corresponding to the country version of Microsoft Excel.
  - 2 Number corresponding to the current country setting in the Microsoft Windows Control Panel or the country number as determined by your Apple system software
- These values apply to number separators:
- 3 Decimal separator
  - 4 Zero (or 1000) separator
  - 5 List separator
- These values apply to R1C1-style references:
- 6 Row character
  - 7 Column character
  - 8 Lowercase row character
  - 9 Lowercase column character
  - 10 Character used instead of the left bracket ([])
  - 11 Character used instead of the right bracket ([])

These values apply to array characters:

- 12 Character used instead of the left bracket ({} )
- 13 Character used instead of the right bracket ({} )
- 14 Column separator
- 15 Row separator
- 16 Alternate array item separator to use if the current array separator is the same as the decimal separator

These values apply to format code symbols:

- 17 Date separator
- 18 Time separator
- 19 Year symbol
- 20 Month symbol
- 21 Day symbol
- 22 Hour symbol
- 23 Minute symbol
- 24 Second symbol
- 25 Currency symbol
- 26 "General" symbol

These values apply to format codes:

- 27 Number of decimal digits to use in currency formats
- 28 Number indicating the current format for negative currencies
  - 0 = (\$currency) or (currency\$)
  - 1 = -\$currency or -currency\$
  - 2 = \$-currency or currency-\$
  - 3 = \$currency- or currency\$-where currency is any number and the \$ represents the current currency symbol.
- 29 Number of decimal digits to use in noncurrency number formats
- 30 Number of characters to use in month names
- 31 Number of characters to use in weekday names
- 32 Number indicating the date order
  - 0 = Month-Day-Year
  - 1 = Day-Month-Year
  - 2 = Year-Month-Day

These values apply to logical format values:

- 33 TRUE if using 24-hour time; FALSE if using 12-hour time.
- 34 TRUE if not displaying functions in English; otherwise, returns FALSE.
- 35 TRUE if using the metric system; FALSE if using the English measurement system.
- 36 TRUE if a space is added before the currency symbol; otherwise, returns FALSE.
- 37 TRUE if currency symbol precedes currency values; FALSE if it follows currency values.
- 38 TRUE if using minus sign for negative numbers; FALSE if using parentheses.
- 39 TRUE if trailing zeros are displayed for zero currency values; otherwise, returns FALSE.
- 40 TRUE if leading zeros are displayed for zero currency values; otherwise, returns FALSE.
- 41 TRUE if leading zero is displayed in months (when months are displayed as numbers); otherwise, returns FALSE.
- 42 TRUE if leading zero is shown in days (when days are displayed as numbers); otherwise, returns FALSE.
- 43 TRUE if using four-digit years; FALSE if using two-digit years.
- 44 TRUE if date order is month-day-year when displaying dates in long form; FALSE if date order is day-month-year.
- 45 TRUE if leading zero is shown in the time; otherwise, returns FALSE.

38 The number 0, 1, or 2 indicating the type of error-checking as set by the ERROR function. For more information, see ERROR.

39 A reference to the currently defined error-handling macro (set by the ERROR function), or the #N/A error value if none is specified.

- 40 If screen updating is turned on (set by the ECHO function), returns TRUE; otherwise, returns FALSE.
- 41 A horizontal array of cell ranges, as R1C1-style text, that were previously selected with the Goto command from the Edit menu or the FORMULA.GOTO macro function.
- 42 If your computer is capable of playing sounds, returns TRUE; otherwise, returns FALSE.
- 43 If your computer is capable of recording sounds, returns TRUE; otherwise, returns FALSE.
- 44 A three-column array of all currently registered procedures in dynamic link libraries (DLLs). The first column contains the names of the DLLs that contain the procedures (in Microsoft Excel for Windows) or the names of the files that contain the code resources (in Microsoft Excel for the Macintosh). The second column contains the names of the procedures in the DLLs (in Microsoft Excel for Windows) or code resources (in Microsoft Excel for the Macintosh). The third column contains text strings specifying the data types of the return values, and the number and data types of the arguments. For more information about DLLs and code resources and data types, see the "Using the CALL and REGISTER Functions" in the Appendix for the Microsoft Excel Worksheet Function Reference, or Using the CALL and REGISTER Functions in online Help.
- 45 If Microsoft Windows for Pen Computing is running, returns TRUE; otherwise, returns FALSE.
- 46 If the Move Selection After Enter check box is selected in the Edit tab of the Options dialog box, returns TRUE; otherwise, returns FALSE.
- 47 Reserved.
- 48 Path to the library subdirectory for Microsoft Excel, as text.
- 49 MAPI session currently in use, returned as a string of hex digits encoding the mail session value.
- 50 If the Full Screen mode is on, returns TRUE; otherwise, FALSE.
- 51 If the the formula bar is displayed in Full Screen mode, returns TRUE; otherwise, FALSE.
- 52 If the status bar is displayed in Full Screen mode, returns TRUE; otherwise, FALSE.
- 53 The name of the topmost custom dialog sheet currently running in a modal window, or #N/A if no dialog sheet is currently running.
- 54 If the Edit Directly In Cell check box is selected on the Edit tab in the Options dialog box, returns TRUE; otherwise, returns FALSE.
- 55 TRUE if the Alert Before Overwriting Cells check box in the Edit tab on Options dialog box is selected; otherwise, FALSE.
- 56 Standard font name in the General tab in the Options dialog box, as text.
- 57 Standard font size in the General tab in the Options dialog box, as a number
- 58 If the Recently Used File list check box in the General tab on the Options dialog box is selected, returns TRUE; otherwise, FALSE.
- 59 If the Display Old Menus check box in the General tab on the Options dialog box is selected, returns TRUE; otherwise, FALSE.
- 60 If the Tip Wizard is enabled, returns TRUE; otherwise, FALSE.
- 61 Number of custom list entries listed in the Custom Lists tab of the Options dialog box.
- 62 Returns information about available file converters.
- 63 Returns the type of mail system in use by Excel:  
0 = no mail transport detected  
1 = MAPI based transport  
2 = PowerTalk based transport (Macintosh only)
- 64 If the Ask to Update Automatic Links check box in the Edit tab of the Options dialog box is selected, returns TRUE; otherwise, FALSE.
- 65 If the Cut, Copy, and Sort Objects with Cells check box in the Edit tab on the Options dialog box is selected, returns TRUE; otherwise, FALSE.
- 66 Default number of sheets in a new workbook, as a number, from the Edit tab on Options dialog box.

- 67 Default file directory location, as text, from the General tab in the Options dialog box.
- 68 If the Show ToolTips check box on the Toolbars dialog box is selected, returns TRUE; otherwise, FALSE.
- 69 If the Large Buttons check box in the Toolbars dialog box is selected, returns TRUE; otherwise, FALSE.
- 70 If the Prompt for Summary Info check box in the General tab on the Options dialog box is selected, returns TRUE; otherwise, FALSE.
- 71 TRUE if Microsoft Excel is open for in-place object editing (OLE). If FALSE, it is opened normally.
- 72 TRUE if the Color Toolbars check box is selected in the Toolbars dialog box. FALSE if the Color Toolbars check box is not selected.

#### **Related Functions**

GET.DOCUMENT Returns information about a document

GET.WINDOW Returns information about a window

List of Information Functions

## INSERT.PICTURE

Macro Sheets Only

Equivalent to choosing the Picture command from the Insert menu. This function is available for Microsoft Excel for Windows only .

### Syntax

**INSERT.PICTURE**(file\_name, filter\_number)

**INSERT.PICTURE?**(file\_name, filter\_number)

File\_name is the name, as text, of the file containing the picture that you want to insert into your workbook.

Filter\_number is a number specifying which converter Microsoft Excel will use to open the file.

| Convert_type | Converter and filename extension |
|--------------|----------------------------------|
| 1            | Windows Bitmaps (bmp)            |
| 2            | Windows Metafile (wmf)           |
| 3            | DrawPerfect (wpg)                |
| 4            | Micrografix Designer/Draw (drw)  |
| 5            | AutoCAD Format 2-D (dxf)         |
| 6            | HP Graphics Language (hgl)       |
| 7            | Computer Graphics Metafile (cgm) |
| 8            | Encapsulated Postscript (eps)    |
| 9            | Tagged Image Format (tif)        |
| 10           | PC PaintBrush (pcx)              |
| 11           | Lotus 1-2-3 Graphic (pic)        |
| 12           | AutoCAD Plot Files (plt)         |
| 13           | Macintosh PICT (pct)             |

### Related Functions

List of [Information Functions](#)

## MAIL.ADD.MAILER

Macro Sheets Only

Equivalent to choosing the Add Mailer command from the Mail submenu on the File menu. Adds a new PowerTalk mailer to the active document. Use this command to add addressing or subject information to a document that you want to send to another user.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.ADD.MAILER( )**

### Remarks

If there is already a mailer, this command fails and returns the #VALUE! error value.

### Related Functions

MAIL.DELETE.MAILER Deletes an existing mailer from the active document

List of Command-Equivalent Functions

## **MAIL.DELETE.MAILER**

Macro Sheets Only

Equivalent to choosing the Delete Mailer command from the Mail submenu on the File menu. Deletes an existing mailer from the active document.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### **Syntax**

**MAIL.DELETE.MAILER( )**

### **Remarks**

If there is no mailer, returns the #VALUE! error value.

### **Related Functions**

MAIL.ADD.MAILER Adds a new PowerTalk mailer to the active document

List of Command-Equivalent Functions

## MAIL.EDIT.MAILER

Macro Sheets Only

Equivalent to choosing the Mailer button when mailer is attached to the current document. Allows you to edit a PowerTalk mailer attached to the active document

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.EDIT.MAILER**(to\_recipients, cc\_recipients, bcc\_recipients, subject, enclosures, which\_address)

**MAIL.EDIT.MAILER?**(to\_recipients, cc\_recipients, bcc\_recipients, subject, enclosures, which\_address)

To\_recipients is the name of the person to whom you want to send the mail. The name should be given as text. To specify more than one name, give the list of names as an array.

Cc\_recipients is the name of those recipients to be carbon copied. A single name should be given as text. To specify more than one name, give the list of names as an array.

Bcc\_recipients is the name of the recipients to be added as blind carbon copies.

Subject is a text string containing the subject text for the mail messages.

Enclosures is an array of strings specifying enclosures as file names.

Which\_address indicates which type of address to use, as a text string, specifying the address type for all recipients. For example, "Fax".

### Remarks

If there is no mailer, returns the #VALUE! error value.

### Related Functions

MAIL.DELETE.MAILER Adds a new PowerTalk mailer to the active document

MAIL.ADD.MAILER Adds a new PowerTalk mailer to the active document

List of Command-Equivalent Functions



## MAIL.FORWARD

Macro Sheets Only

Equivalent to choosing the Forward command from the Mail submenu on the File menu. Creates a new mailer to replace the previous version and brings up the mailer dialog.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.FORWARD( )**

### Remarks

- Returns the #VALUE! error value or #N/A if the current document has no mailer.
- This function is available only when the current document is open and has been received by PowerTalk with a piece of mail to forward.

### Related Functions

MAIL.EDIT.MAILER Allows you to edit a PowerTalk mailer attached to the active document

MAIL.DELETE.MAILER Deletes a new PowerTalk mailer to the active document

MAIL.ADD.MAILER Adds a new PowerTalk mailer to the active document

List of Command-Equivalent Functions

## MAIL.NEXT.LETTER

Macro Sheets Only

Equivalent to choosing the Next Letter command from the Mail submenu on the File menu. Opens the oldest unread Microsoft Excel document from the In Tray as a new window.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.NEXT.LETTER( )**

### Remarks

Returns #VALUE! on error, and #N/A if there are no more letters in the In Tray to open.

### Related Functions

MAIL.EDIT.MAILER Allows you to edit a PowerTalk mailer attached to the active document

MAIL.DELETE.MAILER Adds a new PowerTalk mailer to the active document

MAIL.ADD.MAILER Adds a new PowerTalk mailer to the active document

List of Command-Equivalent Functions

## MAIL.REPLY

Macro Sheets Only

Equivalent to choosing the Reply command from the Mail submenu on the File menu. Replies to the sender of the current letter.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.REPLY( )**

### Remarks

- Returns the #VALUE! error value or #N/A if the current document has no mailer.
- The letter must currently be open.

### Related Functions

MAIL.EDIT.MAILER Allows you to edit a PowerTalk mailer attached to the active document

MAIL.DELETE.MAILER Deletes a new PowerTalk mailer to the active document

MAIL.ADD.MAILER Adds a new PowerTalk mailer to the active document

List of Command-Equivalent Functions

## MAIL.REPLY.ALL

Macro Sheets Only

Equivalent to choosing the Reply All command from the Mail submenu on the File menu. Replies to the sender and all recipients of the current letter.

---

**Note** This function is available on Macintosh computers with Microsoft Excel and Apple PowerTalk only.

---

### Syntax

**MAIL.REPLY.ALL( )**

### Remarks

Returns the #VALUE! error value or #N/A if the current document has no mailer.

### Related Functions

|                                             |                                                                       |
|---------------------------------------------|-----------------------------------------------------------------------|
| <u>MAIL.EDIT.MAILER</u>                     | Allows you to edit a PowerTalk mailer attached to the active document |
| <u>MAIL.DELETE.MAILER</u>                   | Deletes a new PowerTalk mailer to the active document                 |
| <u>MAIL.ADD.MAILER</u>                      | Adds a new PowerTalk mailer to the active document                    |
| List of <u>Command-Equivalent Functions</u> |                                                                       |

## OPEN.LINKS

Macro Sheets Only

Equivalent to choosing the Links command from the Edit menu. Use OPEN.LINKS with the LINKS function to open workbooks linked to a particular sheet.

### Syntax

**OPEN.LINKS**(document\_text1, document\_text2, ..., read\_only, type\_of\_link)

**OPEN.LINKS?**(document\_text1, document\_text2, ..., read\_only, type\_of\_link)

Document\_text1, document\_text2, ... are 1 to 12 arguments that are the names of supporting documents in the form of text, or arrays or references that contain text.

Read\_only is a logical value corresponding to the read/write status of the linked worksheet. If read\_only is TRUE, the sheet can be modified but changes cannot be saved; if FALSE or omitted, changes to the sheet can be saved. Read\_only applies only to Microsoft Excel, WKS, and SYLK documents.

Type\_of\_link is a number from 1 to 6 that specifies what type of link you want to get information about.

| Type_of_link | Link document type   |
|--------------|----------------------|
| 1            | Microsoft Excel link |
| 2            | DDE link             |
| 3            | Reserved             |
| 4            | Not applicable       |
| 5            | Subscriber           |
| 6            | Publisher            |

### Remarks

You can generate an array of the names of linked workbooks with the LINKS function.

### Related Functions

|                                             |                                          |
|---------------------------------------------|------------------------------------------|
| <u>CHANGE.LINK</u>                          | Changes supporting workbook links        |
| <u>GET.LINK.INFO</u>                        | Returns information about a link         |
| <u>LINKS</u>                                | Returns the name of all linked workbooks |
| <u>UPDATE.LINK</u>                          | Updates a link to another document       |
| List of <u>Command-Equivalent Functions</u> |                                          |

## OPTIONS.CALCULATION

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu, and selecting the Calculation tab in the Options dialog box. Sets various worksheet calculation settings.

### Syntax

**OPTIONS.CALCULATION**(type\_num, iter, max\_num, max\_change, update, precision, date\_1904, calc\_save, save\_values)

**OPTIONS.CALCULATION?**(type\_num, iter, max\_num, max\_change, update, precision, date\_1904, calc\_save, save\_values)

Arguments correspond to check boxes and options in the Calculation tab in the Options dialog box.

Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Type\_num is a number from 1 to 3 indicating the type of calculation.

| Type_num | Type of calculation     |
|----------|-------------------------|
| 1        | Automatic               |
| 2        | Automatic except tables |
| 3        | Manual                  |

Iter corresponds to the Iteration check box. The default is FALSE.

Max\_num is the maximum number of iterations. The default is 100.

Max\_change is the maximum change of each iteration. The default is 0.001.

Update corresponds to the Update Remote References check box. The default is TRUE.

Precision corresponds to the Precision As Displayed check box. The default is FALSE.

Date\_1904 corresponds to the 1904 Date System check box. The default is FALSE in Microsoft Excel for Windows and TRUE in Microsoft Excel for the Macintosh.

Calc\_save corresponds to the Recalculate Before Save check box. If calc\_save is FALSE, the workbook is not recalculated before saving when in manual calculation mode. The default is TRUE.

Save\_values corresponds to the Save External Link Values check box. The default is TRUE.

---

**Note** Microsoft Excel for Windows and Microsoft Excel for the Macintosh use different date systems as their default. For more information, see NOW.

---

### Related Functions

List of Command-Equivalent Functions

## OPTIONS.CHART

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu and then selecting the Chart Tab in the Options dialog box when a chart is activated for editing. Sets various chart settings.

### Syntax

**OPTIONS.CHART**(display\_blanks, plot\_visible, size\_with\_window)

**OPTIONS.CHART?**( display\_blanks, plot\_visible, size\_with\_window)

Display\_Blanks is a number indicating how blank cells are plotted.

| Number           | Blanks are displayed as                                                                                            |
|------------------|--------------------------------------------------------------------------------------------------------------------|
| 1                | Not plotted (gaps are shown)                                                                                       |
| 2                | Zero values                                                                                                        |
| 3                | interpolated                                                                                                       |
| Plot_Visible     | is a logical value that if TRUE plots only visible data. If FALSE, all cells in the selection are plotted.         |
| Size_With_Window | is a logical value that if TRUE allows the chart to resize with window. If FALSE, chart will not size with window. |

### Remarks

If any of the arguments are omitted, then that setting is unchanged within the Options dialog box.

### Related Functions

PREFERRED Changes the format of the active chart

SET.PREFERRED Changes the default chart format

List of Command-Equivalent Functions

## OPTIONS.EDIT

Macro Sheets Only

Equivalent to choosing the Options command from the tools menu and then selecting the Edit tab in the Options dialog box. Sets various worksheet editing options.

### Syntax

**OPTIONS.EDIT**(incell\_edit, drag\_drop, alert, entermove, fixed, decimals, copy\_objects, update\_links)

**OPTIONS.EDIT?**(incell\_edit, drag\_drop, alert, entermove, fixed, decimals, copy\_objects, update\_links)

**incell\_edit** is a logical value corresponding to the Edit Directly In Cell check box, which if TRUE allows In Cell Editing. If FALSE, editing directly in cells is not allowed. If omitted, the dialog box setting is not changed.

**drag\_drop** is a logical value corresponding to the Allow Cell Drag And Drop check box, which if TRUE allows drag and dropping on sheets. If FALSE, drag and drop is not allowed. If omitted, the dialog box setting is not changed.

**alert** is a logical value corresponding to the Alert Before Overwriting Cells check box, which if TRUE displays an alert message warning you that cells containing values are about to be overwritten. If FALSE, an alert will not be displayed if your cells are about to be overwritten. If omitted, the dialog box setting is not changed.

**entermove** is a logical value corresponding to the Move Selection After Enter check box, which if TRUE moves the selection after the ENTER key is pressed. If FALSE, the selection is not moved. If omitted, the dialog box setting is not changed.

**fixed** is a logical value corresponding to the Fixed Decimal check box, which if TRUE fixes the decimal place according to decimals. If FALSE, the decimal places are not fixed. If omitted, the dialog box setting is not changed.

**decimals** is a number specifying the number of decimal places. Decimals is ignored if fixed is FALSE or omitted.

**copy\_objects** is a logical value corresponding to the Cut, Copy, and Sort objects with Cells check box. If TRUE allows objects to be cut, copied and sorted with their cells. If FALSE, objects are not cut, copied or sorted with cells. If omitted, the dialog box setting is not changed.

**update\_links** is a logical value corresponding to the Ask to Update Automatic Links check box, which if TRUE will prompt the user when the workbook is opened that has links to other documents. If FALSE, the prompt will not be displayed. If omitted, the dialog box setting is not changed.

### Related Functions

OPTIONS.GENERAL Sets various general Microsoft Excel settings

List of Command-Equivalent Functions



## OPTIONS.GENERAL

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu and then selecting the General tab from the Options dialog box. Sets various general Microsoft Excel settings.

### Syntax

**OPTIONS.GENERAL**(R1C1\_mode, dde\_on, sum\_info, tips, recent\_files, old\_menus, user\_info, font\_name, font\_size, default\_location, alternate\_location, sheet\_num, enable\_under)

**OPTIONS.GENERAL?**(R1C1\_mode, dde\_on, sum\_info, tips, recent\_files, old\_menus, user\_info, font\_name, font\_size, default\_location, alternate\_location, sheet\_num, enable\_under)

Arguments correspond to option buttons, check boxes and text boxes in the General dialog box.

Arguments corresponding to check boxes are logical values. If an argument is TRUE, the check box is selected; if FALSE, the check box is cleared; if omitted, the current setting is not changed.

R1C1\_mode is a number specifying the reference style. Use 1 for A1 style references; 2 for R1C1 style references.

Dde\_on is a logical value corresponding to the Ignore Other Applications check box, which if TRUE ignores DDE request from other applications. If FALSE, DDE requests from other applications are allowed to happen.

Sum\_info is a logical value corresponding to the Prompt for Summary Info check box, which if TRUE displays the summary information dialog box when a workbook is initially saved. If FALSE, the dialog box is not displayed.

Tips is a logical value corresponding to the Show ToolTips check box, which if TRUE allows ToolTips to be displayed as the user scrolls the cursor over toolbar buttons. If FALSE, ToolTips will not be displayed.

Recent\_files is a logical value corresponding to the Recently Used File List check box, which if TRUE displays the four last opened files from the File menu. If FALSE, the file list will not be displayed.

Old\_menus is a logical value corresponding to the Microsoft Excel 4.0 Menus check box, which if TRUE replaces the current menu bar with the Microsoft Excel 4.0 menu bar. If FALSE, the menu bar will not be replaced.

User\_info corresponds to the Name text box, and is the name of the user of this copy of Microsoft Excel. By default it is the registered user, but can be changed to work on a network.

Font\_name corresponds to the Standard Font text box, and is the name of the default font.

Font\_size corresponds to the Size drop-down edit box, and is the size of the default font.

Default\_location corresponds to the Default File Location text box, and is the default location that the File Open command displays. The Default is where Microsoft Excel is installed.

Alternate\_location corresponds to the Alternate Startup File Location text box, and is the alternate startup directory.

Sheet\_num corresponds to the Sheets in New Workbook spin control, and is the number of sheets in a new workbook. Default is 16. Can go up to 255.

Enable\_under enables underlining of the menus. Used for only Microsoft Excel for the Macintosh. Ignored in Microsoft Excel for Windows.

### Related Functions

OPTIONS.LISTS.DELETE Deletes a custom list

OPTIONS.LISTS.GET Returns contents of custom AutoFill lists

OPTIONS.VIEW Sets various view settings

List of Command-Equivalent Functions

## OPTIONS.LISTS.ADD

Macro Sheets Only

This is the equivalent to choosing the Options command from the Tools menu and then selecting the Custom Lists tab in the Options dialog box. Used to add a new custom list.

### Syntax

**OPTIONS.LISTS.ADD(string\_array)**  
**OPTIONS.LISTS.ADD(import\_ref, by\_row)**  
**OPTIONS.LISTS.ADD?(import\_ref, list\_num)**

**String\_array** This is an array of strings or cell reference that contains the custom items in the list, a named cell reference, or an external reference containing the items of the custom list to add.

**Import\_ref** is the reference to the cells that contain the members of the custom list. If A1:A12 contains the twelve signs of the Zodiac starting with Aquarius, then this function will add the contents of these twelve cells as a custom list.

**By\_row** is a logical value that if TRUE, and if importing from cells, assumes that the list items are in sequential rows. If FALSE, assumes that the list items are in columns. If omitted, Microsoft Excel will try to determine the order of the custom lists according to the layout of the sheet.

**List\_num** is a number specifying which list to activate. If omitted, then New List will be activated.

### Remarks

- To replace an existing custom list, you must first delete it and then add the new list to the end.
- If the list already exists, then this function will do nothing. The list is not case sensitive, so "Scorpio" and "scorpio" are treated the same in custom lists.

### Related Functions

|                                             |                                           |
|---------------------------------------------|-------------------------------------------|
| <u>OPTIONS.VIEW</u>                         | Sets various view settings                |
| <u>OPTIONS.LISTS.GET</u>                    | Returns contents of custom AutoFill lists |
| <u>OPTIONS.LISTS.DELETE</u>                 | Deletes a custom list                     |
| List of <u>Command-Equivalent Functions</u> |                                           |

## **OPTIONS.LISTS.DELETE**

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu and then selecting the Delete button on the Custom Lists tab when a custom list is selected.

### **Syntax**

**OPTIONS.LISTS.DELETE(list\_num)**

List\_num is the number of the custom list to delete. The first five lists (numbered zero through 4) cannot be deleted. If list\_num doesn't exist, then FALSE is returned.

### **Related Functions**

OPTIONS.LISTS.GET Returns contents of custom AutoFill lists

OPTIONS.LISTS.ADD Used to add a new custom list

List of Command-Equivalent Functions

## OPTIONS.LISTS.GET

Macro Sheets Only

Returns contents of custom AutoFill lists as an array of text strings.

### Syntax

**OPTIONS.LISTS.GET(list\_num)**

List\_num is a number specifying which list to return, as a horizontal string array. If list\_num is zero, then FALSE is returned.

### Example

OPTIONS.LIST.GET(3) returns the twelve months of the year in the form {"Jan", "Feb", "Mar"}

### Remarks

If list\_num is zero or omitted, then FALSE is returned.

### Related Functions

|                                      |                            |
|--------------------------------------|----------------------------|
| <u>OPTIONS.LISTS.ADD</u>             | Adds a new custom list     |
| <u>OPTIONS.LISTS.DELETE</u>          | Deletes a custom list      |
| <u>OPTIONS.VIEW</u>                  | Sets various view settings |
| List of <u>Information Functions</u> |                            |

## OPTIONS.TRANSITION

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu and then selecting the Transition tab in the Options dialog box. Sets options relating to compatibility with other spreadsheets.

### Syntax

**OPTIONS.TRANSITION**(menu\_key, menu\_key\_action, nav\_keys, trans\_eval, trans\_entry )

**OPTIONS.TRANSITION?**(menu\_key, menu\_key\_action, nav\_keys, trans\_eval, trans\_entry )

Menu\_key is text specifying which alternate menu key to use.

Menu\_key\_action is the number 1 or 2 specifying options for the alternate menu or Help key. In Microsoft Excel for the Macintosh, menu\_key\_action is ignored.

| Menu_key_action | Alternate menu or Help key activates |
|-----------------|--------------------------------------|
|-----------------|--------------------------------------|

|              |                       |
|--------------|-----------------------|
| 1 or omitted | Microsoft Excel menus |
|--------------|-----------------------|

|   |                  |
|---|------------------|
| 2 | Lotus 1-2-3 Help |
|---|------------------|

Nav\_keys is a logical value that corresponds to the Transition Navigation Keys check box, which if TRUE uses alternate navigation keys that correspond to the navigation keys for Lotus 1-2-3. In Microsoft Excel for the Macintosh, nav\_keys is ignored.

Trans\_eval is a logical value that corresponds to the Transition Formula Evaluation check box.

- If trans\_eval is TRUE, Microsoft Excel uses a set of rules compatible with that of Lotus 1-2-3 when calculating formulas. Text is treated as 0. TRUE and FALSE are treated as 1 and 0. Certain characters in database criteria ranges are interpreted the same way Lotus 1-2-3 interprets them.

- If trans\_eval is FALSE or omitted, Microsoft Excel calculates normally.

Trans\_entry is a logical value that corresponds to the Transition Formula Entry check box.

- This argument is available only in Microsoft Excel for Windows.

- If trans\_entry is TRUE, Microsoft Excel accepts formulas entered in Lotus 1-2-3 style.

- If trans\_entry is FALSE or omitted, Microsoft Excel only accepts formulas entered in Microsoft Excel style.

### Related Functions

OPTIONS.LISTS.DELETE Deletes a custom list

OPTIONS.LISTS.GET Returns contents of custom AutoFill lists

OPTIONS.VIEW Sets various view settings

List of Command-Equivalent Functions

## OPTIONS.VIEW

Macro Sheets Only

Equivalent to choosing the Options command from the Tools menu and then selecting the View tab in the Options dialog box. Sets various view settings.

### Syntax

**OPTIONS.VIEW**(formula, status, notes, show\_info, object\_num, page\_breaks, formulas, gridlines, color\_num, headers, outline, zeros, hor\_scroll, vert\_scroll, sheet\_tabs)

**OPTIONS.VIEW?**(formula, status, notes, show\_info, object\_num, page\_breaks, formulas, gridlines, color\_num, headers, outline, zeros, hor\_scroll, vert\_scroll, sheet\_tabs)

Arguments correspond to check boxes and text boxes in the view tab on the options dialog box.

Arguments corresponding to check boxes are logical values. If an argument is TRUE, the check box is selected; if FALSE, the check box is cleared; if omitted, the current setting is not changed.

Formula is a logical value corresponding to the Formula Bar check box. If TRUE, displays the formula bar. If FALSE, the formula bar is not displayed.

Status is a logical value corresponding to the Status Bar check box. If TRUE, the status bar is displayed. If FALSE, the status bar is not displayed.

Notes is a logical value corresponding to the Note Indicator check box. If TRUE, note indicators will be displayed. If FALSE, note indicators will not be displayed.

Object\_num is a number from 1 to 3 corresponding to the display options in the Objects box.

| Object_num   | Corresponds to    |
|--------------|-------------------|
| 1 or omitted | Show All          |
| 2            | Show Placeholders |
| 3            | Hide              |

Page\_breaks is a logical value corresponding to the Automatic Page Breaks check box. If TRUE, automatic page breaks will not appear. If FALSE, automatic page breaks will be appear. The default is FALSE.

Formulas is a logical value corresponding to the Formulas check box. If TRUE, formulas will appear in the cells. If FALSE, formulas will not appear in the cells. The default is FALSE on worksheets and TRUE on macro sheets.

displayed. If FALSE, gridlines will not appear. The default is TRUE.

Color\_num is a number from 0 to 56 corresponding to the gridline and heading. Zero corresponds to automatic color and is the default value.

Headings is a logical value corresponding to the Row & Column Headers check box. If TRUE, row and column headers will be displayed. If FALSE, they will not be displayed. The default is TRUE.

Outline is a logical value corresponding to the Outline Symbols check box. If TRUE, outline symbols will appear. If FALSE, they will not appear. The default is TRUE.

Zeros is a logical value corresponding to the Zero Values check box. If TRUE, zero values will appear, If FALSE, zero values will not appear. The default is TRUE.

Hor\_scroll is a logical value corresponding to the Horizontal Scroll Bar checkbox. If TRUE, the horizontal scroll bar will be displayed. If FALSE, it will not be displayed. The default is TRUE.

Vert\_scroll is a logical value corresponding to the Vertical Scroll Bar checkbox. If TRUE, the vertical scroll bar will be displayed. If FALSE, it will not be displayed. The default is TRUE.

Sheet\_tabs is a logical value corresponding to the Sheet Tabs check box. If TRUE, sheet tabs will be displayed. If FALSE, sheet tabs will not be displayed. The default is TRUE.

### Related Functions

OPTIONS.LISTS.GET Returns contents of custom AutoFill lists

OPTIONS.LISTS.DELETE Deletes a custom list

List of Command-Equivalent Functions

## PASTE.LINK

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu, and then choosing the Paste Link option from the Paste Special dialog box. Pastes copied data or objects and establishes a link to the source of the data or object. The source can be either another Microsoft Excel workbook or another application. Use PASTE.LINK when you want Microsoft Excel to automatically update the paste area with any changes that occur in the source.

### Syntax

**PASTE.LINK( )**

---

**Note** To work properly, the application you are linking to must support dynamic data exchange (DDE) or object linking and embedding (OLE).

---

### Related Functions

|                      |                                           |
|----------------------|-------------------------------------------|
| <u>COPY</u>          | Copies and pastes data or objects         |
| <u>CUT</u>           | Cuts or moves data or objects             |
| <u>PASTE</u>         | Pastes cut or copied data                 |
| <u>PASTE.SPECIAL</u> | Pastes specific components of copied data |

List of Command-Equivalent Functions

## **PASTE.SPECIAL**

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu. Pastes the specified components from the copy area into the current selection. The PASTE.SPECIAL function has four syntax forms.

|                 |                                                |
|-----------------|------------------------------------------------|
| <u>Syntax 1</u> | Pasting into a sheet or macro sheet            |
| <u>Syntax 2</u> | Copying from a sheet and pasting into a chart. |
| <u>Syntax 3</u> | Copying and pasting between charts             |
| <u>Syntax 4</u> | Pasting information from another application.  |



## PASTE.SPECIAL Syntax 1

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu. Pastes the specified components from the copy area into the current selection. The PASTE.SPECIAL function has four syntax forms. Use syntax 1 if you are pasting into a sheet or macro sheet.

### Syntax

**PASTE.SPECIAL**(paste\_num, operation\_num, skip\_blanks, transpose)

**PASTE.SPECIAL?**(paste\_num, operation\_num, skip\_blanks, transpose)

Paste\_num is a number from 1 to 5 specifying what to paste. Paste\_num can also be quoted text of the object you want to paste.

| Paste_num | Pastes   |
|-----------|----------|
| 1         | All      |
| 2         | Formulas |
| 3         | Values   |
| 4         | Formats  |
| 5         | Notes    |

Operation\_num is a number from 1 to 5 specifying which operation to perform when pasting.

| Operation_num | Action   |
|---------------|----------|
| 1             | None     |
| 2             | Add      |
| 3             | Subtract |
| 4             | Multiply |
| 5             | Divide   |

Skip\_blanks is a logical value corresponding to the Skip Blanks check box in the Paste Special dialog box.

- If skip\_blanks is TRUE, Microsoft Excel skips blanks in the copy area when pasting.
- If skip\_blanks is FALSE, Microsoft Excel pastes normally.

Transpose is a logical value corresponding to the Transpose check box in the Paste Special dialog box.

- If transpose is TRUE, Microsoft Excel transposes rows and columns when pasting.
- If transpose is FALSE, Microsoft Excel pastes normally.

### Related Functions

|                                             |                                                         |
|---------------------------------------------|---------------------------------------------------------|
| <u>FORMULA</u>                              | Enters values into a cell or range or onto a chart      |
| <u>PASTE</u>                                | Pastes cut or copied data                               |
| <u>PASTE.LINK</u>                           | Pastes copied data and establishes a link to its source |
| <u>Syntax 2</u>                             | Copying from a sheet and pasting into a chart.          |
| <u>Syntax 3</u>                             | Copying and pasting between charts                      |
| <u>Syntax 4</u>                             | Pasting information from another application.           |
| List of <u>Command-Equivalent Functions</u> |                                                         |

## PASTE.SPECIAL Syntax 2

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu on the Chart menu bar. Pastes the specified components from the copy area into a chart. The PASTE.SPECIAL function has four syntax forms. Use syntax 2 if you have copied from a sheet and are pasting into a chart.

### Syntax

**PASTE.SPECIAL**(rowcol, titles, categories, replace, series)

**PASTE.SPECIAL?**(rowcol, titles, categories, replace, series)

Rowcol is the number 1 or 2 and specifies whether the values corresponding to a particular data series are in rows or columns. Enter 1 for rows or 2 for columns.

Titles is a logical value corresponding to the Series Names In First Column check box (or First Row, depending on the value of rowcol) in the Paste Special dialog box.

- If series is TRUE, Microsoft Excel selects the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the name of the data series in that row (or column).

- If series is FALSE, Microsoft Excel clears the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the first data point of the data series.

Categories is a logical value corresponding to the Categories (X Labels) In First Row (or First Column, depending on the value of rowcol) check box in the Paste Special dialog box.

- If categories is TRUE, Microsoft Excel selects the check box and uses the contents of the first row (or column) of the selection as the categories for the chart.

- If categories is FALSE, Microsoft Excel clears the check box and uses the contents of the first row (or column) as the first data series in the chart.

Replace is a logical value corresponding to the Replace Existing Categories check box in the Paste Special dialog box.

- If replace is TRUE, Microsoft Excel selects the check box and applies categories while replacing existing categories with information from the copied cell range.

- If replace is FALSE, Microsoft Excel clears the check box and applies new categories without replacing any old ones.

Series is a number specifying how cells are added to a chart.

| series | Added as     |
|--------|--------------|
| 1      | New series   |
| 2      | New point(s) |

### Related Functions

[Syntax 1](#) Pasting into a sheet or macro sheet

[Syntax 3](#) Copying and pasting between charts

[Syntax 4](#) Pasting information from another application

List of [Command-Equivalent Functions](#)

## PASTE.SPECIAL Syntax 3

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu on the Chart menu bar. Pastes the specified components from the copy area into a chart. The PASTE.SPECIAL function has four syntax forms. Use syntax 3 if you have copied from a chart and are pasting into a chart.

### Syntax

**PASTE.SPECIAL**(paste\_num)

**PASTE.SPECIAL?**(paste\_num)

Paste\_num is a number from 1 to 3 specifying what to paste.

| Paste_num | Pastes                        |
|-----------|-------------------------------|
| 1         | All (formats and data series) |
| 2         | Formats only                  |
| 3         | Formulas (data series) only   |

### Related Functions

[Syntax 1](#) Pasting into a sheet or macro sheet

[Syntax 2](#) Copying from a sheet and pasting into a chart

[Syntax 4](#) Pasting information from another application

List of [Command-Equivalent Functions](#)

## PASTE.SPECIAL Syntax 4

Macro Sheets Only

Equivalent to choosing the Paste Special command from the Edit menu when pasting data from another application into Microsoft Excel. Use syntax 4 when pasting information from another application.

### Syntax

**PASTE.SPECIAL**(format\_text, pastelink\_logical, display\_icon\_logical, icon\_file, icon\_number, icon\_label)

**PASTE.SPECIAL?**(format\_text, pastelink\_logical, display\_icon\_logical, icon\_file, icon\_number, icon\_label)

Format\_text is text specifying the type of data you want to paste from the Clipboard.

- The valid data types vary depending on the application from which the data was copied. For example, if you're copying data from Microsoft Word, some of the data types are "Microsoft Document Object", "Picture", and "Text".
- For more information about object classes, see your Microsoft Windows or Apple system software documentation.

Pastelink\_logical is a logical value specifying whether to link the pasted data to its source application.

- If pastelink\_logical is TRUE, Microsoft Excel updates the pasted information whenever it changes in the source application.
- If pastelink\_logical is FALSE or omitted, the information is pasted without a link.
- If Microsoft Excel or the source application does not support linking for the specified format\_text, then pastelink\_logical is ignored.

Display\_icon\_logical is a logical value that specifies whether you want an application's icon to be displayed on the worksheet instead of the linked data. Equivalent to the Display as Icon check box on the Paste Special dialog box. If TRUE, the application's icon will be displayed. If FALSE or omitted, the application's icon will not be displayed.

Icon\_file is the name of the file (with an .EXE or .DLL extension) that contains the icon. If display\_icon\_logical is FALSE, this argument is ignored.

Icon\_number is the number associated with the icon and corresponds to the icon's relative position within the Icon Drop Down list box on the Change Icon Dialog box, which appears when you choose the Change Icon button from the Paste Special dialog box. If display\_icon\_logical is FALSE, this argument is ignored.

Icon\_label is the caption that you want to appear below the icon, and is equivalent to the Caption text box on the Change Icon dialog box, which appears when you choose the Change Icon button from the Paste Special dialog box. If display\_icon\_logical is FALSE, this argument is ignored.

### Related Functions

|                                             |                                               |
|---------------------------------------------|-----------------------------------------------|
| <u>Syntax 1</u>                             | Pasting into a sheet or macro sheet           |
| <u>Syntax 2</u>                             | Copying from a sheet and pasting into a chart |
| <u>Syntax 3</u>                             | Copying and pasting between charts            |
| List of <u>Command-Equivalent Functions</u> |                                               |

## PIVOT.ADD.DATA

Macro Sheets Only

Adds a field to a PivotTable.

### Syntax

**PIVOT.ADD.DATA**(name, pivot\_field\_name, new\_name, position, function, calculation, base\_field, base\_item, format\_text)

Name is the name of the PivotTable to which the user wants to add as a data field. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Pivot\_field\_name is the name of a field which the user would like to add to his PivotTable as data or as text.

New\_name is the name you would like to give to the new field once it is added to your PivotTable. If this argument is omitted, Microsoft Excel will pick a default name for you. This function returns new\_name or the name Microsoft Excel chooses for the field.

Position is the position within all the Data fields you would like to place the new data field. If position is omitted, the field will be added as the last data field.

Function is a number from 2 to 2048 specifying how the new field is to be calculated. To compute the value to place in this column choose one value from the following table. If function is omitted, SUM will be used. If the field is a numeric field or text field, COUNTA will be used.

| Value | Function |
|-------|----------|
| 2     | SUM      |
| 4     | COUNTA   |
| 8     | COUNT    |
| 16    | AVERAGE  |
| 32    | MAX      |
| 64    | MIN      |
| 128   | PRODUCT  |
| 256   | STDEV    |
| 512   | STDEVP   |
| 1024  | VAR      |
| 2048  | VARP     |

Calculation is a number between 1 and 9 representing which custom calculation you would like to apply to this data field. This corresponds to the Show Data As drop-down box on the PivotTable Field dialog box. If this argument is omitted, no special calculation will be applied to the data field.

| Value | Calculation       |
|-------|-------------------|
| 1     | Normal            |
| 2     | Difference From   |
| 3     | % Of Item         |
| 4     | % Difference From |
| 5     | Running Total In  |
| 6     | % of Row          |
| 7     | % of Column       |
| 8     | % of Total        |
| 9     | Index             |

Base\_Field is the field on which you want to base the Calculation.

Base\_Item is the item within base\_field on which you want to base the Calculation.

Format\_text is the type of number format you want to apply to the PivotTable data. Corresponds to the number button in the PivotTable Field dialog box, which appears when you choose the PivotTable Field command from the Data menu when the selection is in a data field.

### Remarks

- If name is not a valid PivotTable name, then the #VALUE! error value is returned.
- If field\_name is not a valid field for the current PivotTable then the #VALUE! error value is returned.

#### **Related Functions**

|                                             |                                                               |
|---------------------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>                          | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u>               | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>                  | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>                           | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>                     | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                                   |
| List of <u>Command-Equivalent Functions</u> |                                                               |

## PIVOT.ADD.FIELDS

Macro Sheets Only

Add fields onto a PivotTable.

### Syntax

**PIVOT.ADD.FIELDS**(name, row\_array, column\_array, page\_array, add\_to\_table)

Name is the name of the PivotTable to which the user wants to add fields. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Row\_array is an array of text constants consisting of the names of the fields which the user would like to add to the PivotTable as Row Fields.

Column\_array is an array of text constants consisting of the names of the fields which the user would like to add to the PivotTable as Column Fields.

Page\_array is an array of text constants consisting of the names of the fields which the user would like to add to the PivotTable as Page Fields.

Add\_to\_table is a logical value which if TRUE adds the fields specified by row\_array, column\_array and page\_array to the existing fields on the PivotTable. If add\_bool is FALSE, Microsoft Excel will replace the fields already along the rows, columns and pages with the fields specified by row\_array, column\_array and page\_array.

### Remarks

If name is not a valid PivotTable name, then the #VALUE! error value is returned.

### Related Functions

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>         | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.FIELD</u>            | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>      | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u> | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>    | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>             | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>  | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>          | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>       | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>     | Creates an empty PivotTable                                   |

List of Command-Equivalent Functions

## PIVOT.FIELD

Macro Sheets Only

Pivots a field within a PivotTable.

### Syntax

**PIVOT.FIELD**(name, pivot\_field\_name, orientation, position)

Name is the name of the PivotTable in which the user wants to pivot fields. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Pivot\_field\_name is the name of the field which the user wishes to pivot to another part of the PivotTable. This argument is given as a text constant or a reference to a text constant. If field\_name is omitted, Microsoft Excel uses the field containing the active cell.

Orientation is an integer representing the destination of the field which is being pivoted. If this argument is omitted, then the orientation remains unchanged. The integers refer to orientations as follows:

| Value | Orientation |
|-------|-------------|
| 0     | Hidden      |
| 1     | Row         |
| 2     | Column      |
| 3     | Page        |
| 4     | Data        |

Position is an integer representing where in the orientation the fields will be positioned. Position 1 is the leftmost header position in the row header and the topmost position in the column header. This argument is ignored if orientation is set to 0. If the position argument is omitted, it will default to the last position in the field.

### Remarks

- The function returns TRUE if successful.
- If name is not a valid PivotTable name then the #VALUE! error value is returned.
- If pivot\_field\_name is not a text constant or contains text which is not a valid field name for the PivotTable then the #VALUE! error value is returned.
- If destination is not an integer between 0 and 4, then the #VALUE! error value is returned.

### Related Functions

|                                             |                                                               |
|---------------------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>                       | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u>               | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>                  | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>                           | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>                     | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                                   |
| List of <u>Command-Equivalent Functions</u> |                                                               |



## PIVOT.FIELD.GROUP

Macro Sheets Only

Creates groups within a PivotTable.

### Syntax

**PIVOT.FIELD.GROUP**(start, end, by, periods)

**PIVOT.FIELD.GROUP?**(start, end, by, periods)

**Start** is the beginning date of the range to be grouped. If start is TRUE or omitted, it is assumed to be the first value in the field.

**End** is the ending date of the range to be grouped. If end is TRUE or omitted it is assumed to be the last value in the field.

**By** is the size of the groups to be created. If by is omitted, Microsoft Excel chooses a default group size. If grouping a date field and if periods is not 8(days) then by is ignored.

**Periods** is a number between 1 and 127. It is calculated by summing the values in the following table corresponding to the periods into which you want to group your dates. This argument is ignored if the field is not a date field. This argument takes precedence over By if they are both specified for a date field.

| Value | Periods  |
|-------|----------|
| 1     | Seconds  |
| 2     | Minutes  |
| 4     | Hours    |
| 8     | Days     |
| 16    | Months   |
| 32    | Quarters |
| 64    | Years    |

### Remarks

- This function returns TRUE if the grouping is successful. The #N/A error value is returned if the grouping failed.
- If no arguments are specified and multiple items within the header field are selected then this function groups those selected items.
- If no arguments are specified and a single item within the header field is selected then the #VALUE! error value is returned.

### Related Functions

|                                               |                                                               |
|-----------------------------------------------|---------------------------------------------------------------|
| <u><a href="#">PIVOT.ADD.DATA</a></u>         | Adds a field to a PivotTable as a Data Field                  |
| <u><a href="#">PIVOT.ADD.FIELDS</a></u>       | Adds fields to a PivotTable                                   |
| <u><a href="#">PIVOT.FIELD</a></u>            | Pivots fields within a PivotTable                             |
| <u><a href="#">PIVOT.FIELD.PROPERTIES</a></u> | Changes the properties of a field inside a PivotTable         |
| <u><a href="#">PIVOT.FIELD.UNGROUP</a></u>    | Ungroups all selected groups within a PivotTable              |
| <u><a href="#">PIVOT.ITEM</a></u>             | Moves an item within a PivotTable                             |
| <u><a href="#">PIVOT.ITEM.PROPERTIES</a></u>  | Changes the properties of an item within a header field       |
| <u><a href="#">PIVOT.REFRESH</a></u>          | Refreshes a PivotTable                                        |
| <u><a href="#">PIVOT.SHOW.PAGES</a></u>       | Creates new sheets in the workbook containing the active cell |
| <u><a href="#">PIVOT.TABLE.WIZARD</a></u>     | Creates an empty PivotTable                                   |

List of [Command-Equivalent Functions](#)

## PIVOT.FIELD.PROPERTIES

Macro Sheets Only

Changes the properties of a field inside a PivotTable.

### Syntax

**PIVOT.FIELD.PROPERTIES**(name, pivot\_field\_name, new\_name, orientation, function, formats)

**PIVOT.FIELD.PROPERTIES?**(name, pivot\_field\_name, new\_name, orientation, function, formats)

Name is the name of the PivotTable containing the field which the user wants to edit. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Pivot\_field\_name is the name of a field in the PivotTable which the user would like to edit, as text. If it is omitted, Microsoft Excel uses the field containing the active cell.

New\_name is the name which you would like to rename the current field. If it is omitted, the name of the current field will not change.

Orientation is a number between 0 and 4 specifying which area will show the field containing the active cell. If zero, then the field is deleted and all other arguments to this function are ignored. If this argument is omitted, the orientation of the field will not change.

| Value | Orientation       |
|-------|-------------------|
| 0     | Delete            |
| 1     | Display as Row    |
| 2     | Display as Column |
| 3     | Display as Page   |
| 4     | Display as Data   |

Function is a number between 0 and 4094 specifying which calculation or subtotals to apply to the field. If you will be showing the field in the header (orientation 1, 2, or 3), add up the values from the table corresponding to the subtotals you would like to show. If you will be showing the field as a data field (orientation 4), choose one value from the table. If an entry in this column is left blank, Microsoft Excel will not change the calculation or subtotal which are currently attached to the field.

| Value | Function     |
|-------|--------------|
| 0     | NO SUBTOTALS |
| 1     | AUTOMATIC    |
| 2     | SUM          |
| 4     | COUNTA       |
| 8     | COUNT        |
| 16    | AVERAGE      |
| 32    | MAX          |
| 64    | MIN          |
| 128   | PRODUCT      |
| 256   | STDEV        |
| 512   | STDEVP       |
| 1024  | VAR          |
| 2048  | VARP         |

Formats is either a one or a two dimension array, depending on whether the field is a header field or a data field.

- If the active field is a header field (orientation argument is 1, 2 or 3) then this is a two dimensional array. Each row of the array should consist of two entries. The first is a text string corresponding to the item whose property is being changed. The second element specifies whether the item will be hidden. If this argument is TRUE, the item will be hidden and therefore will not be displayed in the PivotTable. If the argument is FALSE, then the item will be displayed in the PivotTable.
- If the active field is a data field, then the array is a one dimensional array with four elements. The first element is a number between 1 and 9 specifying which calculation you wish to apply to the current data field. This corresponds to the Show Data As drop-down box on the PivotTable Field dialog box.

| Value | Format           |
|-------|------------------|
| 1     | Normal           |
| 2     | Difference From  |
| 3     | %Of Item         |
| 4     | %Difference From |
| 5     | Running Total In |
| 6     | %Of Row          |
| 7     | %Of Column       |
| 8     | %Of Subtotal     |
| 9     | Index            |

- The second element contains a text string representing the field to which your Data Field is related. This argument is not necessary for the Normal calculation. If omitted, Microsoft Excel will use the first field that would appear in the Base Field list box.
- The third element must contain a text string representing an item in the base field on which to base your calculation. Note that this argument is not necessary for calculations like Running Total In which relies only on a Base Field. If omitted, Microsoft Excel will use the first item that would appear in the Base Item list box.
- The fourth element is a text string representing the number format you wish to apply to the data field.

#### Remarks

- If pivot\_field\_name is not a valid field name for the PivotTable then the #VALUE! error value is returned.
- If name is not a valid PivotTable name, then the #VALUE! error value is returned.
- If the orientation and function arguments do not contain numbers or if these arguments contain numbers which are out of range then the #VALUE! error value is returned.

#### Related Functions

|                                             |                                                               |
|---------------------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>                       | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>                          | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.UNGROUP</u>                  | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>                           | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>                     | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                                   |
| List of <u>Command-Equivalent Functions</u> |                                                               |

## PIVOT.FIELD.UNGROUP

Macro Sheets Only

Ungroups all selected groups within a PivotTable.

### Syntax

**PIVOT.FIELD.UNGROUP( )**

### Remark

If the active cell is on a field header, then all the groups in that field are ungrouped and the field will be removed from the PivotTable. Similarly, if the last group in a Parent field is ungrouped, the entire field will be removed from the PivotTable.

### Related Functions

|                                             |                                                               |
|---------------------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>                       | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>                          | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u>               | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.ITEM</u>                           | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>                     | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                                   |
| List of <u>Command-Equivalent Functions</u> |                                                               |

## PIVOT.ITEM

Macro Sheets Only

Moves an item within a PivotTable.

### Syntax

**PIVOT.ITEM**(name, pivot\_field\_name, pivot\_item\_name, position)

Name is the name of the PivotTable within which an item will be repositioned. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Pivot\_field\_name is the name of the field within which an item will be repositioned, given as a text string. If pivot\_field\_name is omitted, Microsoft Excel will use the field containing the active cell. If the active cell is not within a field, then this argument is required.

Pivot\_item\_name is the name of the item to be repositioned in its field (given as a text constant). If it is omitted, Microsoft Excel uses the item containing the active cell. If the active cell is not contained within an item, then this argument is required.

Position is a number representing where in the field the items will be moved. Position 1 is the topmost position within the row field and the leftmost position within the column field and the highest position within the page field. If the position argument is omitted, it will default to the last position in the field.

### Remarks

- If an item is set to be visible, but its display is suppressed because there is no data, this item still occupies a valid position.
- If name is not a valid PivotTable name then the #VALUE! error value is returned.
- If pivot\_field\_name is not a text string, or if pivot\_field\_name is not a text string within a valid field name, then #VALUE! is returned.
- If pivot\_item\_name is an item which is not currently showing in the PivotTable because it does not exist in the field pivot\_field\_name, the #VALUE! error value is returned.

### Related Functions

|                                             |                                                               |
|---------------------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>                       | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>                          | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u>               | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>                  | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>                     | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                                   |
| List of <u>Command-Equivalent Functions</u> |                                                               |

## PIVOT.ITEM.PROPERTIES

Macro Sheets Only

Changes the properties of an item within a header field.

### Syntax

**PIVOT.ITEM.PROPERTIES**(name, pivot\_field\_name, pivot\_item\_name, new\_name, position, show, active\_page)

Name is the name of the PivotTable containing the item which the user wants to edit.

Pivot\_field\_name is the name of a field in the PivotTable containing the item which the user would like to edit. If it is omitted, Microsoft Excel uses the field containing the active cell.

Pivot\_item\_name is the name of the item which the user would like to edit. If it is omitted, Microsoft Excel uses the item containing the active cell.

New\_name is the name which you would like to rename the current item. If it is omitted, then the name of the current item will not change.

Position is a number representing where in the field the item will appear. Position 1 is the topmost position within the row field, the leftmost position within the column field, and the highest position within the page field. If the position argument is omitted, it will default to the last position in the field.

Show is a logical value which if TRUE causes the item to appear in the PivotTable. If FALSE, the item will be hidden.

Active\_page is a logical value specifying whether item\_name will become the active item in the page field. If TRUE then item\_name will become the active item in the page field. If FALSE or omitted, the active item in the page field does not change. Applies to only page fields.

### Remarks

- If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.
- If pivot\_field\_name is not a header field, then this function will return the #VALUE! error value.

### Related Functions

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>         | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>       | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>            | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>      | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u> | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>    | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>             | Moves an item within a PivotTable                             |
| <u>PIVOT.REFRESH</u>          | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>       | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>     | Creates an empty PivotTable                                   |

List of Command-Equivalent Functions

## PIVOT.REFRESH

Macro Sheets Only

Refreshes a PivotTable.

### Syntax

**PIVOT.REFRESH**(name)

Name is the name of the PivotTable the user would like to refresh with current data. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

### Remarks

- If the function is successful, it returns TRUE; otherwise, it returns the #VALUE! error value.
- If name is not a valid PivotTable name, then the #VALUE! error value is returned.

### Related Functions

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>         | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>       | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>            | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>      | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u> | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>    | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>             | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>  | Changes the properties of an item within a header field       |
| <u>PIVOT.SHOW.PAGES</u>       | Creates new sheets in the workbook containing the active cell |
| <u>PIVOT.TABLE.WIZARD</u>     | Creates an empty PivotTable                                   |

List of Command-Equivalent Functions

## PIVOT.SHOW.PAGES

Macro Sheets Only

Creates new sheets in the workbook containing the active cell. The function will iterate through each item in page\_field and create a new PivotTable on a new sheet with the page field set to that particular item.

### Syntax

**PIVOT.SHOW.PAGES**(name, page\_field)

Name is the name of the target PivotTable. If name is omitted, Microsoft Excel will use the PivotTable containing the active cell.

Page\_field is the name of a page field in the PivotTable specified by the name argument.

### Remarks

- If the function is successful, it returns TRUE; otherwise, it returns the #VALUE! error value.
- If name is not a valid PivotTable name then the #VALUE! error value is returned.

### Related Functions

|                                             |                                                         |
|---------------------------------------------|---------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>                       | Adds a field to a PivotTable as a Data Field            |
| <u>PIVOT.ADD.FIELDS</u>                     | Adds fields to a PivotTable                             |
| <u>PIVOT.FIELD</u>                          | Pivots fields within a PivotTable                       |
| <u>PIVOT.FIELD.GROUP</u>                    | Creates groups within a PivotTable                      |
| <u>PIVOT.FIELD.PROPERTIES</u>               | Changes the properties of a field inside a PivotTable   |
| <u>PIVOT.FIELD.UNGROUP</u>                  | Ungroups all selected groups within a PivotTable        |
| <u>PIVOT.ITEM</u>                           | Moves an item within a PivotTable                       |
| <u>PIVOT.ITEM.PROPERTIES</u>                | Changes the properties of an item within a header field |
| <u>PIVOT.REFRESH</u>                        | Refreshes a PivotTable                                  |
| <u>PIVOT.TABLE.WIZARD</u>                   | Creates an empty PivotTable                             |
| List of <u>Command-Equivalent Functions</u> |                                                         |



## PIVOT.TABLE.WIZARD

Macro Sheets Only

Creates an empty PivotTable.

### Syntax

**PIVOT.TABLE.WIZARD**(type, source, destination, name, row\_grand, col\_grand, save\_data, apply\_auto\_format, autopage )

**PIVOT.TABLE.WIZARD?**(type, source, destination, name, row\_grand, col\_grand, save\_data, apply\_auto\_format, autopage)

Type is a number specifying the type of source data used to create the PivotTable.

| Value | Type of source data              |
|-------|----------------------------------|
| 1     | Microsoft Excel List or Database |
| 2     | External data source             |
| 3     | Multiple consolidation ranges    |
| 4     | Another PivotTable               |

Source can be one of four things. If type is one, then source is a cell reference or name to the range to be used as the PivotTable source. If type is two, then source is a one- dimensional array describing the external database to be used as the PivotTable source. If type is three, then source is a multi-dimensional array listing the cell ranges and associated page field items describing the consolidation PivotTable source. If type is four, then source is the name of another PivotTable with which to share its source.

Destination is a cell reference or name. The upper left corner of this range will act as the upper left corner of the PivotTable which will be created. If destination is omitted, Microsoft Excel will choose a default location for the PivotTable.

Name is the name of the PivotTable to be created given as a text. If name is omitted, Microsoft Excel will choose a default name.

Row\_grand is a logical value which if TRUE displays a Grand Total for each row on the PivotTable. If FALSE, a Grand Total for each row is not displayed.

Col\_grand is a logical value which if TRUE displays a Grand Total for each column. If FALSE, a Grand Total for each column is not displayed.

Save\_data is a logical value which if TRUE causes the data for the PivotTable to be saved along with the PivotTable definition. If FALSE, the data is not saved along with the PivotTable definition.

Apply\_auto\_format is a logical value which if TRUE causes autoformatting upon pivoting or refreshing. If FALSE, the PivotTable will not be formatted automatically upon pivoting or refreshing.

Autopage Applies only to type 3. This argument is a logical value which if TRUE or omitted causes Microsoft Excel to create a page field automatically. If FALSE, the page field must be created manually.

### Remarks

- The function will return TRUE if successful; otherwise, returns the #VALUE! error value.
- If destination is not a valid Microsoft Excel reference, then #VALUE! error value is returned.
- If name is not a valid PivotTable name, then the #VALUE! error value is returned.

### Related Functions

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <u>PIVOT.ADD.DATA</u>         | Adds a field to a PivotTable as a Data Field                  |
| <u>PIVOT.ADD.FIELDS</u>       | Adds fields to a PivotTable                                   |
| <u>PIVOT.FIELD</u>            | Pivots fields within a PivotTable                             |
| <u>PIVOT.FIELD.GROUP</u>      | Creates groups within a PivotTable                            |
| <u>PIVOT.FIELD.PROPERTIES</u> | Changes the properties of a field inside a PivotTable         |
| <u>PIVOT.FIELD.UNGROUP</u>    | Ungroups all selected groups within a PivotTable              |
| <u>PIVOT.ITEM</u>             | Moves an item within a PivotTable                             |
| <u>PIVOT.ITEM.PROPERTIES</u>  | Changes the properties of an item within a header field       |
| <u>PIVOT.REFRESH</u>          | Refreshes a PivotTable                                        |
| <u>PIVOT.SHOW.PAGES</u>       | Creates new sheets in the workbook containing the active cell |

## List of Command-Equivalent Functions

## QUERY.GET.DATA

Macro Sheets Only

Builds a new query using the supplied information. The application Microsoft Query nor any dialog boxes are displayed. Equivalent to choosing the Get External Data command from the Data menu when the Microsoft Query add-in (XLQUERY.XLA) is installed. For more information on add-ins, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**QUERY.GET.DATA**(connection\_string, query\_text, keep\_query\_def, field\_names, row\_numbers, destination)

**QUERY.GET.DATA?**(connection\_string, query\_text, keep\_query\_def, field\_names, row\_numbers, destination)

Connection\_string supplies information, such as the data source name, user ID, and passwords, necessary to making a SQL connection to an external data source. For example: "DSN=Myserver; Server=server1; UID=dbayer; PWD=buyer1; Database=nwind".

- You must define the data source name (DSN) used in connection\_string before you try to connect to it.
- You can enter connection\_string as an array or a string. If connection\_string exceeds 250 characters, you must enter it as an array.
- If QUERY.GET.DATA is unable to access the data source using connection\_string, it returns the #N/A error value.

Query\_text is the SQL language query to be executed on the data source.

Keep\_query\_def is a logical value that, if TRUE or omitted, preserves the query definition. If FALSE, the query definition is lost and the data from the query no longer constitutes a data range.

Field\_names is a logical value that, if TRUE or omitted, places field names from Microsoft Query into the first row of the data range. If FALSE, the field names are discarded.

Row\_numbers is a logical value that, if TRUE, places row numbers from Microsoft Query into the first column in the data range. If FALSE or omitted, the row numbers are discarded.

Destination is the location as a cell reference where you want the data placed. If destination is in a current data range then that data range is changed to reflect the new SQL statement. The default destination is the currently selected cell or range.

### Remarks

- If the information provided is not sufficient to create the query then the error value #REF! is returned.
- If the Microsoft Query Tool is unavailable or can not be found, #NA is returned.
- If connection string is longer than 255 characters, the string will be truncated at the last semi-colon.

### Related Functions

[QUERY.REFRESH](#) Refreshes the data in a data range returned by the Microsoft Query tool

List of [Command-Equivalent Functions](#)

## QUERY.REFRESH

Macro Sheets Only

Refreshes the data in a data range returned by the Microsoft Query tool. This function acts exactly the same as the Refresh tool.

### Syntax

**QUERY.REFRESH**(reference)

Reference is the reference to a single cell inside a data range. If reference is not in a data range then the error value #REF! is returned.

### Related Functions

[QUERY.GET.DATA](#) Builds a new query using the supplied information

List of [Command-Equivalent Functions](#)

## **RENAME.OBJECT**

Macro Sheets Only

Renames the selected object or group. This is useful for giving objects names more relevant to their usage. This is also useful if it is uncertain how the object may have been named.

### **Syntax**

**RENAME.OBJECT**(new\_name)

New\_name is the new name to be given to the selected object.

### **Related Functions**

GET.OBJECT Returns information about a specified object

INSERT.OBJECT Equivalent to choosing the Object command from the Insert menu

SELECT Syntax 2 Selects objects on worksheets

List of Command-Equivalent Functions

## ROUTING.SLIP

Macro Sheets Only

Equivalent to choosing the Add Routing Slip command from the File menu. Adds or Edits the routing slip attached to the current workbook.

### Syntax

**ROUTING.SLIP**(recipients,subject, message, route\_num, return\_logical, status\_logical)

**ROUTING.SLIP?**(recipients,subject, message, route\_num, return\_logical, status\_logical)

**Recipients** is the name of the person to whom you want to send the mail. The name should be given as text.

- To specify more than one name, give the list of names as an array. For example, ROUTING.SLIP({"John", "Paul", "George", "Ringo"}) would send the active workbook to the four names in the array. You can also refer to a range on a sheet or macro sheet that contains a list of names to whom you want the mail to be sent.

- Specifying recipients while a routing is in progress only modifies the non-grayed recipients (that is, those recipients who have not received the message yet). Recipients who have already received, reviewed and forwarded the routed document cannot be modified.

**Subject** is a text string containing the subject text used for the mail messages used to route the document. If omitted, the default subject line is "Routing: name", where name is the file name or title as displayed in the Summary Info dialog box, if available.

**Message** is a text string containing the body text used for the mail messages used to route the document.

**Route\_num** is a number indicating the type of routing method.

| Route_num    | Method                    |
|--------------|---------------------------|
| 1 or omitted | One after another routing |
| 2            | All at once routing       |

**Return\_logical** is a logical value which, if TRUE or omitted, indicates that the routing should be returned to the originator when the routing is complete. If FALSE, the routing will end with the last recipient in the To list box in the Routing Slip Dialog box.

**Status\_logical** is a logical value corresponding to the Track Status check box in the Routing Slip dialog box. If TRUE or omitted, status tracking messages for the routing are sent. FALSE means that no status tracking is performed.

### Remarks

- If this function is used on a document that is already being routed, the route\_num, status\_logical and return\_logical arguments are ignored (they cannot be changed).

- When arguments are omitted and a routing slip already exists, the omitted arguments are replaced by the current values of the routing slip.

### Related Functions

ROUTE.DOCUMENT Routes the document using the defined routing slip information

SEND.MAIL Sends the active workbook using email

List of Command-Equivalent Functions

## **ROUTE.DOCUMENT**

Macro Sheets Only

Routes the document using the defined routing slip information.

### **Syntax**

**ROUTE.DOCUMENT()**

### **Remarks**

If there is no routing slip, returns #N/A. If an error occurs or routing is not enabled for the system, returns #VALUE!.

### **Related Functions**

SEND.MAIL Sends the active workbook using email

ROUTING.SLIP Adds or Edits the routing slip attached to the current workbook

List of Command-Equivalent Functions

## **SAVE.WORKSPACE**

Macro Sheets Only

Equivalent to choosing the Save Workspace command from the File menu. Saves the currently opened workbook or workbooks as a workspace.

### **Syntax**

**SAVE.WORKSPACE**(name\_text)

**SAVE.WORKSPACE?**(name\_text)

Name\_text is the name of the workspace to save.

### **Related Function**

[SAVE.AS](#) Specifies a new filename.

List of [Command-Equivalent Functions](#)



## SCALE

Macro Sheets Only

Equivalent to choosing the Scale command from the Format menu, which is available when a chart is active. There are five syntax forms of this function.

|                 |                                                                                  |
|-----------------|----------------------------------------------------------------------------------|
| <u>Syntax 1</u> | Changes the position, formatting, and scaling of the category axis in 2-D charts |
| <u>Syntax 2</u> | Changes the position, formatting, and scaling of the value axis in 2-D charts    |
| <u>Syntax 3</u> | Changes the position, formatting, and scaling of the category axis in 3-D charts |
| <u>Syntax 4</u> | Changes the position, formatting, and scaling of the series axis in 3-D charts   |
| <u>Syntax 5</u> | Changes the position, formatting, and scaling of the value axis in 3-D charts    |

## SCALE

Macro Sheets Only

Equivalent to choosing the Selected Axis command from the Format menu when a chart's axis is selected, and then choosing the Scale tab. There are five syntax forms of this function. Syntax 1 of SCALE applies if the selected axis is a category (x) axis on a 2-D chart and the chart is not an xy (scatter) chart. Use this syntax of SCALE to change the position, formatting, and scaling of the category axis.

### Syntax 1

**SCALE**(cross, cat\_labels, cat\_marks, between, max, reverse)

**SCALE?**(cross, cat\_labels, cat\_marks, between, max, reverse)

Arguments correspond to text boxes and check boxes in the Scale tab on the Format Axis dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Cross is a number corresponding to the Value (Y) Axis Crosses At Category number text box. The default is 1. Cross is ignored if max is set to TRUE.

Cat\_labels is a number corresponding to the Number Of Categories Between Tick Mark Labels text box. The default is 1.

Cat\_marks is a number corresponding to the Number Of Categories Between Tick Marks text box. The default is 1.

Between corresponds to the Value (Y) Axis Crosses Between Categories check box. This argument only applies if cat\_labels is set to a number other than 1.

Max corresponds to the Value (Y) Axis Crosses At Maximum Category check box. If max is TRUE, it overrides any setting for cross.

Reverse corresponds to the Categories In Reverse Order check box.

### Related Functions

|                                             |                                                                                  |
|---------------------------------------------|----------------------------------------------------------------------------------|
| <u>AXES</u>                                 | Controls whether axes on a chart are visible                                     |
| <u>GRIDLINES</u>                            | Controls whether chart gridlines are visible                                     |
| <u>Syntax 2</u>                             | Changes the position, formatting, and scaling of the value axis in 2-D charts    |
| <u>Syntax 3</u>                             | Changes the position, formatting, and scaling of the category axis in 3-D charts |
| <u>Syntax 4</u>                             | Changes the position, formatting, and scaling of the series axis in 3-D charts   |
| <u>Syntax 5</u>                             | Changes the position, formatting, and scaling of the value axis in 3-D charts    |
| List of <u>Command-Equivalent Functions</u> |                                                                                  |

## SCALE

Macro Sheets Only

Equivalent to choosing the Selected Axes command from the Format menu when a chart's axis is selected, and then choosing the Scale tab. There are five syntax forms of this function. Syntax 2 of SCALE applies if the selected axis is a value (y) axis on a 2-D chart, or either axis on an xy (scatter) chart. Use this syntax of SCALE to change the position, formatting, and scaling of the value axis.

### Syntax 2

**SCALE**(min\_num, max\_num, major, minor, cross, logarithmic, reverse, max)

**SCALE?**(min\_num, max\_num, major, minor, cross, logarithmic, reverse, max)

The first five arguments correspond to the five range variables on the Scale tab. Each argument can be either the logical value TRUE or a number:

- If an argument is TRUE, Microsoft Excel selects the Auto check box.
- If an argument is a number, that number is used for the variable.

Min\_num corresponds to the Minimum check box and is the minimum value for the value axis.

Max\_num corresponds to the Maximum check box and is the maximum value for the value axis.

Major corresponds to the Major Unit check box and is the major unit of measure.

Minor corresponds to the Minor Unit check box and is the minor unit of measure.

Cross corresponds to the Category (X) Axis Crosses At text box for the value (y) axis of a 2-D chart or the Value (Y) Axis Crosses At text box for the category (x) axis of an xy (scatter) chart.

The last three arguments are logical values corresponding to check boxes in the Scale tab. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Logarithmic corresponds to the Logarithmic Scale check box.

Reverse corresponds to the Values In Reverse Order check box.

Max corresponds to the Category (X) Axis Crosses At Maximum Value/Category check box.

### Related Functions

[Syntax 1](#) Changes the position, formatting, and scaling of the category axis in 2-D charts

[Syntax 3](#) Changes the position, formatting, and scaling of the category axis in 3-D charts

[Syntax 4](#) Changes the position, formatting, and scaling of the series axis in 3-D charts

[Syntax 5](#) Changes the position, formatting, and scaling of the value axis in 3-D charts

List of [Command-Equivalent Functions](#)

## SCALE

Macro Sheets Only

Equivalent to choosing the Selected Axes command from the Format menu when a chart's axis is selected, and then choosing the Scale tab. There are five syntax forms of this function. Syntax 3 of SCALE applies if the selected axis is a category (x) axis on a 3-D chart. Use this syntax of SCALE to change the position, formatting, and scaling of the category axis.

### Syntax 3

**SCALE**(cat\_labels, cat\_marks, reverse, between)

**SCALE?**(cat\_labels, cat\_marks, reverse, between)

Cat\_labels is a number corresponding to the Number Of Categories Between Tick Labels text box. The default is 1. Cat\_labels can also be a logical value. If TRUE, an automatic setting will be used. If FALSE, or omitted, the number will be used.

Cat\_marks is a number corresponding to the Number Of Categories Between Tick Marks text box. The default is 1. Cat\_marks can also be a logical value. If TRUE, an automatic setting will be used. If FALSE, or omitted, the number will be used.

Reverse corresponds to the Categories in Reverse Order check box in the Scale tab. If reverse is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Between corresponds to the Value (Z) Axis Crosses Between Categories check box. If between is TRUE, Microsoft Excel selects the check box and the data points appear between categories. If between is FALSE or omitted, Microsoft Excel clears the check box.

### Related Functions

Syntax 1 Changes the position, formatting, and scaling of the category axis in 2-D charts

Syntax 2 Changes the position, formatting, and scaling of the value axis in 2-D charts

Syntax 4 Changes the position, formatting, and scaling of the series axis in 3-D charts

Syntax 5 Changes the position, formatting, and scaling of the value axis in 3-D charts

List of Command-Equivalent Functions

## SCALE

Macro Sheets Only

Equivalent to choosing the Selected Axes command from the Format menu when a chart's axis is selected, and then choosing the Scale tab. There are five syntax forms of this function. Syntax 4 of SCALE applies if the selected axis is a series (y) axis on a 3-D chart. Use this syntax of SCALE to change the position, formatting, and scaling of the series axis.

### Syntax 4

Series (y) axis, 3-D chart

**SCALE**(series\_labels, series\_marks, reverse)

**SCALE?**(series\_labels, series\_marks, reverse)

Series\_labels is a number corresponding to the Number Of Series Between Tick Labels text box. The default is 1. Series\_labels can also be a logical value. If TRUE, and automatic setting will be used. If FALSE, or omitted, the number will be used.

Series\_marks is a number corresponding to the Number Of Series Between Tick Marks text box. The default is 1. Series\_marks can also be a logical value. If TRUE, and automatic setting will be used. If FALSE, or omitted, the number will be used.

Reverse is a logical value that corresponds to the Series In Reverse Order check box in the Scale tab. If reverse is TRUE, Microsoft Excel displays the series in reverse order; if FALSE or omitted, Microsoft Excel displays the series normally.

### Related Functions

|                 |                                                                                  |
|-----------------|----------------------------------------------------------------------------------|
| <u>Syntax 1</u> | Changes the position, formatting, and scaling of the category axis in 2-D charts |
| <u>Syntax 2</u> | Changes the position, formatting, and scaling of the value axis in 2-D charts    |
| <u>Syntax 3</u> | Changes the position, formatting, and scaling of the category axis in 3-D charts |
| <u>Syntax 5</u> | Changes the position, formatting, and scaling of the value axis in 3-D charts    |

List of Command-Equivalent Functions

## SCALE

Macro Sheets Only

Equivalent to choosing the Selected Axes command from the Format menu when a chart's axis is selected, and then choosing the Scale tab. There are five syntax forms of this function. Syntax 5 of SCALE applies if the selected axis is a value (z) axis on a 3-D chart. Use this syntax of SCALE to change the position, formatting, and scaling of the value axis.

### Syntax 5

**SCALE**(min\_num, max\_num, major, minor, cross, logarithmic, reverse, min)

**SCALE?**(min\_num, max\_num, major, minor, cross, logarithmic, reverse, min)

The first five arguments correspond to the five range variables in the Format Axis dialog box, as shown in the following list. Each argument can be either the logical value TRUE or a number.

- If TRUE or omitted, the Auto check box is selected.
- If a number, that number is used.

Min\_num corresponds to the Minimum box and is the minimum value for the value axis.

Max\_num corresponds to the Maximum box and is the maximum value for the value axis.

Major corresponds to the Major Unit box and is the major unit of measure.

Minor corresponds to the Minor Unit box and is the minor unit of measure.

Cross corresponds to the Floor (XY Plane) Crosses At box.

The last three arguments are logical values corresponding to check boxes in the Scale tab. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box.

Logarithmic corresponds to the Logarithmic Scale check box.

Reverse corresponds to the Values In Reverse Order check box.

Min corresponds to the Floor (XY Plane) Crosses At Minimum Value check box.

### Related Functions

[Syntax 1](#) Changes the position, formatting, and scaling of the category axis in 2-D charts

[Syntax 2](#) Changes the position, formatting, and scaling of the value axis in 2-D charts

[Syntax 3](#) Changes the position, formatting, and scaling of the category axis in 3-D charts

[Syntax 4](#) Changes the position, formatting, and scaling of the series axis in 3-D charts

List of [Command-Equivalent Functions](#)

## SHOW.BAR

Macro Sheets Only

Displays the specified menu bar. Use SHOW.BAR to display a menu bar you have created with the ADD.BAR function or to display a built-in Microsoft Excel menu bar.

### Syntax

**SHOW.BAR**(bar\_num)

Bar\_num is the number of the menu bar you want to display. It can be the number of one of the Microsoft Excel built-in menu bars, the number returned by a previously executed ADD.BAR function, or a reference to a cell containing a previously executed ADD.BAR function.

If bar\_num is omitted, Microsoft Excel displays the appropriate menu bar for the active workbook as shown in the following table.

| Bar_num | Bar displayed                                                                                                     |
|---------|-------------------------------------------------------------------------------------------------------------------|
| 1       | A sheet or macro sheet (Microsoft Excel 4.0)                                                                      |
| 2       | A chart (Microsoft Excel 4.0)                                                                                     |
| 3       | No active window                                                                                                  |
| 4       | The Info window                                                                                                   |
| 5       | A sheet or macro sheet (short menus)                                                                              |
| 6       | A chart (short menus)                                                                                             |
| 7       | Shortcut menus 1 (for Cells, Workbook tabs, Toolbars, VB Windows)                                                 |
| 8       | Shortcut menus 2 (for objects)                                                                                    |
| 9       | Shortcut menus 3 (for chart elements)                                                                             |
| 10      | A sheet or macro sheet (Microsoft Excel 5.0)                                                                      |
| 11      | A chart (Microsoft Excel 5.0)                                                                                     |
| 12      | A Visual Basic module                                                                                             |
| 13-35   | Reserved for use by shortcut menus. These numbers will return an error if a macro tries to do anything with them. |
| 36-50   | Custom menu bar for macro use                                                                                     |

### Remarks

- When displaying a built-in menu bar, you can display only bars 1 or 5 if a sheet or macro sheet is active, bars 2 or 6 if a chart is active, and so on. If you try to display a chart menu bar while a sheet or macro sheet is active, SHOW.BAR returns an error and interrupts the current macro.
- Displaying a custom menu bar disables automatic menu-bar switching when different types of sheets are selected. For example, if a custom menu bar is displayed and you switch to a chart, neither of the two chart menus is automatically displayed as it would be when you are using the built-in menu bars. Automatic menu-bar switching is reenabled when a built-in bar is displayed using SHOW.BAR.

### Example

The following macro formula displays short menus on a worksheet or macro sheet:

SHOW.BAR(5)

### Related Functions

- [ADD.BAR](#) Adds a menu bar
- [DELETE.BAR](#) Deletes a menu bar
- [SHOW.TOOLBAR](#) Hides or displays a toolbar
- List of [Customizing Functions](#)

## SHOW.TOOLBAR

Macro Sheets Only

Equivalent to selecting the check box corresponding to a toolbar in the Toolbars dialog box, which appears when you select the Toolbars command from the View menu. Hides or displays a toolbar.

### Syntax

**SHOW.TOOLBAR**(bar\_id, visible, dock, x\_pos, y\_pos, width, protect, tool\_tips, large\_buttons, color\_buttons)

Bar\_id is a number or name of a toolbar corresponding to the toolbars you want to display. For detailed information about bar\_id, see ADD.TOOL.

Visible is a logical value that, if TRUE, specifies that the toolbar is visible or, if FALSE, specifies that the toolbar is hidden.

Dock specifies the docking location of the toolbar.

| Dock | Position of toolbar     |
|------|-------------------------|
| 1    | Top of workspace        |
| 2    | Left edge of workspace  |
| 3    | Right edge of workspace |
| 4    | Bottom of workspace     |
| 5    | Floating (not docked)   |

X\_pos specifies the horizontal position of the toolbar.

- If the toolbar is docked (not floating), x\_pos is measured horizontally from the left edge of the toolbar to the left edge of the toolbar's docking area.
- If the toolbar is floating, x\_pos is measured horizontally from the left edge of the toolbar to the right edge of the rightmost toolbar in the left docking area.
- X\_pos is measured in points. A point is 1/72nd of an inch.

Y\_pos specifies the vertical position of the toolbar.

- If the toolbar is docked, y\_pos is measured vertically from the top edge of the toolbar to the top edge of the toolbar's docking area.
- If the toolbar is floating, y\_pos is measured vertically from the top edge of the toolbar to the top edge of the Microsoft Excel workspace.
- Y\_pos is measured in points.

Width specifies the width of the toolbar and is measured in points. If you omit width, Microsoft Excel uses the existing width setting.

Protect is a number specifying the degree to which you can modify a toolbar and its buttons. Each succeeding protect number retains the protection status of its previous numbers. For example, a protect status of 3 (a toolbar cannot become docked if it is floating) assumes the protection status of 0, 1, and 2 as well.

| Protect | Description                                                                                                                                  |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------|
| 0       | Default. Toolbars can be re-shaped, docked, and floating. Toolbar buttons can be removed from and moved to the toolbar.                      |
| 1       | Toolbars can be re-shaped, docked, and floating. Toolbar buttons can not be removed from nor moved to the toolbar.                           |
| 2       | A floating toolbar cannot be re-shaped. It can be docked.                                                                                    |
| 3       | A floating toolbar cannot be docked. If it is already docked, it cannot become floating.                                                     |
| 4       | The toolbar cannot be moved at all. If it is already floating, it cannot be re-shaped or moved. If it is docked, it cannot become un-docked. |

Tool\_tips is a logical value that corresponds to the Show ToolTips check box on the Toolbar dialog box. If TRUE, ToolTips will be displayed. If FALSE, ToolTips will not be displayed.

Large Buttons is a logical value that corresponds to the Large Buttons check box on the Toolbar dialog box. If TRUE, large buttons will be displayed. If FALSE, large buttons will not be displayed.

Color\_buttons is a logical value that corresponds to the Color Toolbars check box. If TRUE, the



toolbar buttons will be displayed in color. If FALSE, the toolbar buttons will not be displayed in color.

**Related Function**

ADD.TOOLBAR      Creates a new toolbar with the specified tools

List of Customizing Functions

## SORT

Macro Sheets Only

Equivalent to choosing the Sort command from the Data menu. Sorts the rows or columns of the selection according to the contents of a key row or column within the selection. Use SORT to rearrange information into ascending or descending order.

### Syntax 1

For Worksheet and macro sheets

**SORT**(orientation, key1, order1, key2, order2, key3, order3, header, custom, case)

**SORT?**(orientation, key1, order1, key2, order2, key3, order3, header, custom, case)

### Syntax 2

For PivotTables

**SORT**(orientation, key1, order1, type, custom)

**SORT?**(orientation, key1, order1, type, custom)

Orientation is a number specifying whether to sort by rows or columns. Enter 1 to sort top to bottom or 2 to sort left to right.

Key1 is a reference to the cell or cells you want to use as the first sort key. The sort key identifies which column to sort by when sorting rows or which row to sort by when sorting columns. For a PivotTable, if type is 1, then key1 is a cell reference which indicates what value to sort by. There are two ways to specify sort keys:

| Type of key                                                                                                                                  | Examples                  |
|----------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|
| An external reference to the active worksheet.                                                                                               | !B:B or !Price            |
| An R1C1-style reference in the form of text. If the reference is relative, it is assumed to be relative to the active cell in the selection. | "C2" or "C[1]" or "Price" |

Order1 specifies whether to sort the row or column containing key1 in ascending or descending order. Enter 1 to sort in ascending order or 2 to sort in descending order.

Key2, order2, key3, and order3 are similar to key1 and order1. Key2 specifies the second sort key, and order2 specifies whether to sort the row or column containing key2 in ascending or descending order. Key3 and order3 work similarly.

Header is a number indicating how Microsoft Excel is to handle headers on list.

| Header       | Defined                                             |
|--------------|-----------------------------------------------------|
| 0            | Microsoft Excel will guess if there is a header     |
| 1            | Forces Microsoft Excel to assume there is a header  |
| 2 or omitted | Forces Microsoft Excel to assume there is no header |

Type is a number specifying whether to sort the field by labels or values. Use one to sort by values or two to sort by labels.

Custom is a number that specifies what kind of custom sorting you want. This corresponds to the First Key Sort Order drop-down box in the Sort Options dialog box. For a PivotTable, custom is a number indicating what custom sort order to use when sorting labels.

| Number | Type of sort                                           |
|--------|--------------------------------------------------------|
| 1      | Normal                                                 |
| 2      | Weekdays in abbreviated form ("Sun", "Mon", and so on) |
| 3      | Weekdays                                               |
| 4      | Months in abbreviated form ("Jan" "Feb", and so on)    |
| 5      | Months                                                 |

Case is a logical value that determines whether the sort is case sensitive. If TRUE, the sort is case sensitive. If FALSE or omitted, the sort will not be case sensitive.

**Tip** If you want to sort using more than three keys, then sort the data three keys at a time, starting with the least important group of keys and progressing to the most important group, but listing the most important key first within each group.

---

**Remarks**

In the dialog box form of this function, if the header argument is omitted, then Microsoft Excel will guess whether or not there are headers.

**Related Functions**

List of [Command-Equivalent Functions](#)

## SPELLING

Macro Sheets Only

Equivalent to choosing the Spelling command from the Tools menu. Checks the spelling of words in the current selection.

### Syntax

**SPELLING**(custom\_dic, ignore\_uppercase, always\_suggest)

Custom\_dic is the filename of the custom dictionary to examine if words are not found in the main dictionary. If custom\_dic is omitted, the currently specified dictionary is used.

Ignore\_uppercase is a logical value corresponding to the Ignore Words In Uppercase check box.

| <b>If ignore_uppercase is</b> | <b>Microsoft Excel will</b> |
|-------------------------------|-----------------------------|
|-------------------------------|-----------------------------|

|      |                                       |
|------|---------------------------------------|
| TRUE | Ignore words in all uppercase letters |
|------|---------------------------------------|

|       |                                      |
|-------|--------------------------------------|
| FALSE | Check words in all uppercase letters |
|-------|--------------------------------------|

|         |                         |
|---------|-------------------------|
| Omitted | Use the current setting |
|---------|-------------------------|

Always\_suggest is a logical value corresponding to the Always Suggest check box.

| <b>If always_suggest is</b> | <b>Microsoft Excel will</b> |
|-----------------------------|-----------------------------|
|-----------------------------|-----------------------------|

|      |                                                                                     |
|------|-------------------------------------------------------------------------------------|
| TRUE | Display a list of suggested alternate spellings when an incorrect spelling is found |
|------|-------------------------------------------------------------------------------------|

|       |                                             |
|-------|---------------------------------------------|
| FALSE | Wait for user to input the correct spelling |
|-------|---------------------------------------------|

|         |                         |
|---------|-------------------------|
| Omitted | Use the current setting |
|---------|-------------------------|

### Related Function

SPELLING.CHECK Checks the spelling of a word

List of Command-Equivalent Functions

## SQL.BIND

Macro Sheets Only

Specifies where results from a SQL query are placed when they are retrieved with SQL.RETRIEVE. If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.BIND**(connection\_num, column, reference)

Connection\_num is the unique connection ID of the data source for which you want to define storage.

- Connection\_num was returned by a previously executed SQL.OPEN function.
- If connection\_num is not valid, then SQL.BIND returns the #VALUE! error value.

Column is the number of the result column that is to be bound. Result columns are numbered from left to right starting with 1. If column is omitted then all bindings for connection\_num are removed. Column number 0 contains row numbers for the result set. If column number 0 is bound then SQL.RETRIEVE will return row numbers in the bound location.

Reference is a single cell reference on the worksheet where the results should be placed. If reference is omitted, the binding is removed for the column.

When SQL.RETRIEVE is called, the result rows in this column will be placed in the reference cell and the cells immediately below reference. The number of rows that will be retrieved is one of the SQL.RETRIEVE arguments.

### Remarks

- If SQL.BIND is completed successfully then it will return a vertical array listing the bound columns on the current connection. If SQL.BIND is unable to bind the result column then it will return the error value #N/A. In such a case SQL.BIND will place error information in memory for the SQL.ERROR function, if such information is available.
- SQL.BIND tells the ODBC interface where to place results when they are retrieved using SQL.RETRIEVE. Binding is not necessary but can be useful if you want the results from different columns to be placed in disjoint worksheet locations.
- If bindings are used, SQL.BIND must be called once for each column in the result set. If a result column is not bound then it will not be returned. A binding remains valid for as long as connection\_num is open.
- SQL.BIND may be called whenever there is a valid connection\_num. Calls to SQL.BIND will not affect results that have already been retrieved.

### Example

SQL.BIND(conn1,1,"[Book1]Sheet1!C1") stores data obtained from the data source conn1 on Sheet1 from left to right in cell C1, starting with column1.

### Related Functions

|                                      |                                                   |
|--------------------------------------|---------------------------------------------------|
| <a href="#">SQL.OPEN</a>             | Establishes a connection with a data source       |
| <a href="#">SQL.EXEC.QUERY</a>       | Sends a query to a data source                    |
| <a href="#">SQL.RETRIEVE.TO.FILE</a> | Retrieves query results and places them in a file |
| <a href="#">SQL.RETRIEVE</a>         | Retrieves query results                           |
| <a href="#">SQL.GET.SCHEMA</a>       | Gets information about a connected data source.   |
| <a href="#">SQL.CLOSE</a>            | Closes a connection to a data source.             |
| <a href="#">SQL.ERROR</a>            | Returns detailed error information                |

List of [Command-Equivalent Functions](#)

## SQL.CLOSE

Macro Sheets Only

Terminates a connection to an external data source. If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.CLOSE(connection\_num)**

Connection\_num is the unique connection ID of the data source from which you wish to disconnect.

- Connection\_num is returned by a previously executed SQL.OPEN function.
- If connection\_num is not valid, SQL.CLOSE returns the #VALUE! error value.

### Remarks

- If the connection is successfully terminated SQL.CLOSE will return zero and the connection ID number is then no longer valid.
- If SQL.CLOSE is unable to disconnect with the data source then it will return the error value the #N/A error value. In such a case SQL.CLOSE will place error information in memory for the SQL.ERROR function, if such information is available.
- SQL.CLOSE works with data sources in much the same manner as FCLOSE works with files. If the call is successful then SQL.CLOSE will terminate the specified data source connection.

### Example

SQL.CLOSE (conn1) will close the connection with connection\_num conn1

### Related Functions

|                                             |                                                   |
|---------------------------------------------|---------------------------------------------------|
| <u>SQL.OPEN</u>                             | Establishes a connection with a data source       |
| <u>SQL.EXEC.QUERY</u>                       | Sends a query to a data source                    |
| <u>SQL.BIND</u>                             | Specifies storage for a result column             |
| <u>SQL.RETRIEVE.TO.FILE</u>                 | Retrieves query results and places them in a file |
| <u>SQL.RETRIEVE</u>                         | Retrieves query results                           |
| <u>SQL.GET.SCHEMA</u>                       | Gets information about a connected data source.   |
| <u>SQL.ERROR</u>                            | Returns detailed error information                |
| List of <u>Command-Equivalent Functions</u> |                                                   |

## SQL.ERROR

Macro Sheets Only

Returns detailed error information when it is called after a previous XLODBC.XLA function call has failed. If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

#### SQL.ERROR()

Calling SQL.ERROR returns detailed error information in a two dimensional array. Each row in the array describes exactly one error. If a function call generates multiple errors, a row will be created for each error. When SQL.ERROR is processed successfully, all SQL.ERROR information is cleared. Also, all SQL.ERROR information is automatically removed whenever an ODBC function completes successfully.

Each row will have exactly three fields. The information in these three fields is obtained through the SQLERROR API function call. These fields are:

- A textual message describing the error.
- The ODBC error class and subclass as a character string.
- The data source native error code as a numeric value.

If one or more of these fields is not available for the type of error that was encountered, the field will be left blank. For more information on the meaning of these three fields, refer to Chapter 24, "ODBC Function Reference", in the Microsoft Open Database Connectivity Programmer's Reference for the SQLERROR API function. See also Appendix A, "ODBC Error Codes" in the same manual.

### Remarks

- SQL.ERROR cannot provide information on Excel errors.
- If no error information is available when SQL.ERROR is called, then it will return the error value #N/A but does not post any error information to SQL.ERROR.
- SQL.ERROR stores and returns error information by processing SQL.ERROR (in the ODBC API reference) in a loop until SQL\_NO\_DATA\_FOUND is encountered. In the SQL.ERROR function, the error information is automatically defined and stored in memory whenever an XLODBC.XLA function fails. If the call is successful then SQL.ERROR will return the error information available. If SQL.ERROR fails, it will return the error value #N/A.

### Example

When entered as an array formula, the following example will return error information about each argument in, for example, SQL.GET.QUERY.

SQL.ERROR()

### Related Functions

|                                      |                                                   |
|--------------------------------------|---------------------------------------------------|
| <a href="#">SQL.OPEN</a>             | Establishes a connection with a data source       |
| <a href="#">SQL.EXEC.QUERY</a>       | Sends a query to a data source                    |
| <a href="#">SQL.BIND</a>             | Specifies storage for a result column             |
| <a href="#">SQL.RETRIEVE.TO.FILE</a> | Retrieves query results and places them in a file |
| <a href="#">SQL.RETRIEVE</a>         | Retrieves query results                           |
| <a href="#">SQL.CLOSE</a>            | Closes a data source connection                   |

List of [Command-Equivalent Functions](#)

## SQL.EXEC.QUERY

Macro Sheets Only

Sends a query to a data source using an existing connection. If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.EXEC.QUERY(connection\_num, query\_text)**

Connection\_num is the unique connection ID of the data source you want to query.

- Connection\_num is returned by a previously executed SQL.OPEN function.
  - If connection\_num is not valid, SQL.EXEC.QUERY returns the #VALUE! error value.
- Query\_text is the SQL language query that is to be executed on the data source. The query must follow the SQL syntax guidelines in the appendix of the Microsoft Excel ODBC Developers Guide.
- If SQL.EXEC.QUERY is unable to execute query\_text on the specified data source, SQL.EXEC.QUERY returns the #N/A error value.
  - Excel limits strings to a length of 255 characters. If query\_text needs to be longer than 255 characters then query\_text should be a vertical array or vertical range of cells. The values in the array will be joined together to form the complete SQL query.

### Remarks

- Before calling SQL.EXEC.QUERY a connection must be established with a data source using SQL.OPEN. A successful call to SQL.OPEN returns a unique connection ID number. SQL.EXEC.QUERY uses that connection ID number to send SQL language queries to the data source.
- Any results generated from the query will not be returned immediately-- SQL.EXEC.QUERY only executes the query. Retrieving results is handled by the functions SQL.RETRIEVE and SQL.RETRIEVE.TO.FILE.
- If SQL.EXEC.QUERY is called using a previously used connection ID number, all pending results on that connection will automatically be discarded. The connection ID will then refer to the new query and its results.
- If SQL.EXEC.QUERY is unable to successfully execute the query on the specified data source then an error value will be returned. In such a case SQL.EXEC.QUERY will place error information in memory for the SQL.ERROR function, if such information is available. If SQL.EXEC.QUERY is able to successfully execute the query on the specified connection it will return one of three values depending on the type of SQL statement that was executed.
  - If it was a SELECT statement then SQL.EXEC.QUERY will return the number of result columns available.
  - If it was an UPDATE, INSERT, or DELETE statement then SQL.EXEC.QUERY will return the number of rows affected by the statement.
  - If it was a legal SQL query that is not in one of the categories above, SQL.EXEC.QUERY will return 0 (zero).

### Example

SQL.EXEC.QUERY(conn1, "SELECT Custmr\_ID, Due Date FROM Orders WHERE Order\_Amt > 100") executes a SQL query from a SQL table named "Orders"

### Related Functions

|                                      |                                                   |
|--------------------------------------|---------------------------------------------------|
| <a href="#">SQL.OPEN</a>             | Establishes a connection with a data source       |
| <a href="#">SQL.BIND</a>             | Specifies storage for a result column             |
| <a href="#">SQL.RETRIEVE.TO.FILE</a> | Retrieves query results and places them in a file |
| <a href="#">SQL.RETRIEVE</a>         | Retrieves query results                           |
| <a href="#">SQL.GET.SCHEMA</a>       | Gets information about a connected data source.   |
| <a href="#">SQL.CLOSE</a>            | Closes a data source connection                   |
| <a href="#">SQL.ERROR</a>            | Returns detailed error information                |

List of [Command-Equivalent Functions](#)



## SQL.GET.SCHEMA

Macro Sheets Only

Returns information about the structure of the data source on a particular connection. The return value from a successful call to SQL.GET.SCHEMA depends on the type of information that was requested. A list of the accepted requests and their respective return values is listed in the syntax section below.

If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

SQL.GET.SCHEMA(**connection\_num**, **type\_num**, **qualifier\_text**)

Connection\_num is the unique connection ID of the data source you want information about.

- Connection\_num is returned by a previously executed SQL.OPEN function.
- If connection\_num is not valid, SQL.GET.SCHEMA returns the #VALUE! error value.

Type\_num specifies the type of information you want returned. The following is a list of valid type\_num values.

| Type_num | Returns                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1        | A list of available data sources, as a vertical array.                                                                                                                                                                                                                                                                                                                                                                                              |
| 2        | A list of databases on the current connection, as a vertical array .                                                                                                                                                                                                                                                                                                                                                                                |
| 3        | A list of owners in a database on the current connection, as a vertical array.                                                                                                                                                                                                                                                                                                                                                                      |
| 4        | A list of tables for a given owner and database on the current connection, as a vertical array.                                                                                                                                                                                                                                                                                                                                                     |
| 5        | A list of columns in a particular table and their data types, as a two-dimensional array. The returned array will have two fields and will have a row for each column in the table. The first field will be the name of the column. The second field is the data type of the column. The data type will be a number that corresponds to the ODBC C header file data types. These #define numbers are found in Microsoft Excel ODBC Developer Guide. |
| 6        | User ID of the current user                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 7        | Name of the current database.                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 8        | The name of the data source as given in the ODBC.INI file.                                                                                                                                                                                                                                                                                                                                                                                          |
| 9        | The name of the data source DBMS (i.e. Oracle, SQL Server, etc.).                                                                                                                                                                                                                                                                                                                                                                                   |
| 10       | The server name for the data source.                                                                                                                                                                                                                                                                                                                                                                                                                |
| 11       | The terminology used by the data source to refer to owners ( i.e. "owner", "Authorization ID", "Schema", etc.).                                                                                                                                                                                                                                                                                                                                     |
| 12       | The terminology used by the data source to refer to tables ( i.e. "table", "file", etc.).                                                                                                                                                                                                                                                                                                                                                           |
| 13       | The terminology used by the data source to refer to qualifiers (i.e. "database" or "directory").                                                                                                                                                                                                                                                                                                                                                    |
| 14       | The terminology used by the data source to refer to procedures (i.e. "database procedure", "stored procedure", or "procedure").                                                                                                                                                                                                                                                                                                                     |

Qualifier\_text is only included for type\_num values of 3, 4 and 5. It is a text string used to qualify the search for the requested information and should be enclosed by quotation marks.

| Type_num | Qualifier_text                                                                                                                                                                                                                                                                                                   |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3        | The value of qualifier_text should be the name of a database in the current data source. SQL.GET.SCHEMA will then only return the names of table owners in that database.                                                                                                                                        |
| 4        | The value of qualifier_text should be both a database name and an owner name. The syntax of qualifier_text is "DatabaseName.OwnerName". A period is used to separate the two names. SQL.GET.SCHEMA will then return an array of table names that are located in the given database and owned by the given owner. |
| 5        | The value of qualifier_text should be the name of a table. Information about the columns in that table will be returned.                                                                                                                                                                                         |

### Remarks

- If SQL.GET.SCHEMA is unable to find the requested information then it will return the error value #N/A. In such a case SQL.GET.SCHEMA will place error information in memory for the SQL.ERROR function, if such information is available.
- SQL.GET.SCHEMA works with the ODBC functions SQLGetInfo and SQLTables to find the requested information. Refer to the Microsoft Excel ODBC Programmer Guide for more information on these two functions.

### Example

SQL.GET.SCHEMA(conn1, 7) returns the name of the current database.

SQL.GET.SCHEMA(conn1, 9) returns the name of the DBMS used by the data source.

### Related Functions

|                             |                                                   |
|-----------------------------|---------------------------------------------------|
| <u>SQL.OPEN</u>             | Establishes a connection with a data source       |
| <u>SQL.EXEC.QUERY</u>       | Sends a query to a data source                    |
| <u>SQL.BIND</u>             | Specifies storage for a result column             |
| <u>SQL.RETRIEVE.TO.FILE</u> | Retrieves query results and places them in a file |
| <u>SQL.RETRIEVE</u>         | Retrieves query results                           |
| <u>SQL.CLOSE</u>            | Closes a data source connection                   |
| <u>SQL.ERROR</u>            | Returns detailed error information                |

List of Command-Equivalent Functions

## SQL.OPEN

Macro Sheets Only

Establishes a connection with a data source. If the connection is successfully established SQL.OPEN will return a connection ID number. Use the connection ID number with other ODBC macro functions to identify a connection.

If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.OPEN(connection\_string, output\_ref, driver\_prompt)**

Connection\_string is a text string that contains the information necessary to establish a connection to a data source. Any data-source-name that is used in connection\_string must be an existing data source name defined with ODBC Setup or the ODBC Administration Utility.

- Connection\_string must follow the format described in Chapter 24, "ODBC Function Reference", of the Microsoft Open Database Connectivity Programmer's Reference for SQLDriverConnect. In this string the user supplies the data source name, one or more user ID's, one or more passwords, and any other information necessary to successfully connect to a DBMS. An example of a SQL.OPEN connection\_string entered would be: "DSN=MyServer; UID=dbayer; PWD=123; Database=pubs"
- Enter the connection\_string as an array when the length exceeds 255 characters. Or enter connection\_string as an array of cells containing the same information. The connection string should be horizontal array.

Output\_ref is a single cell reference where you want the completed connection string to be placed. Use output\_ref when you want SQL.OPEN to return the completed connection string. If output\_ref is omitted, a completed connection string will not be returned.

Driver\_prompt is a number from 1 to 4 specifying if and how you want to be prompted by the driver. This sets the fDriverCompletion flag in ODBC's SQLDriverConnect.

| Number | Description                                                                                                                                                                    |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1      | Always brings up a dialog box.                                                                                                                                                 |
| 2      | Bring up dialog only if there is not enough information to connect. The driver uses information from the connection string and from the data source specification as defaults. |
| 3      | Same as 2, but the driver grays and disables any prompts for information not needed.                                                                                           |
| 4      | If the connection string is unsuccessful, do not bring up a dialog box.                                                                                                        |

### Remarks

- If SQL.OPEN is unable to connect with the information provided then it will return the error value #N/A. In such a case, SQL.OPEN will place error information in memory for the SQL.ERROR function, if more information is available.
- If the call is successful then SQL.OPEN will return a unique connection ID number that can be used in future function calls to identify the connection.
- If connection\_array does not allow SQL.OPEN to connect to a data source, then the error value #N/A will be returned.

### Example

conn1=SQL.OPEN('DSN=NWind;DBQ=C:\MSQUERY;FIL=dBASE4',C15, 2) sets the name conn1 to the return value of SQL.OPEN, which connects to the NWind data source, specifies where to place the connection string, and displays the driver dialog box only if additional information is needed.

### Related Functions

|                                      |                                                   |
|--------------------------------------|---------------------------------------------------|
| <a href="#">SQL.EXEC.QUERY</a>       | Sends a query to a data source                    |
| <a href="#">SQL.BIND</a>             | Specifies storage for a result column             |
| <a href="#">SQL.RETRIEVE.TO.FILE</a> | Retrieves query results and places them in a file |
| <a href="#">SQL.RETRIEVE</a>         | Retrieves query results                           |
| <a href="#">SQL.GET.SCHEMA</a>       | Gets information about a connected data source.   |

SQL.CLOSE

Closes a data source connection

SQL.ERROR

Returns detailed error information

List of Command-Equivalent Functions

## SQL.RETRIEVE

Macro Sheets Only

Retrieves all or part of the results from a previously executed query. The connection used must have already been established using the macro function SQL.OPEN. Also, a query must already have been executed using SQL.EXEC.QUERY and results must be pending.

If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.RETRIEVE**(**connection\_num**, destination\_ref, max\_columns, max\_rows, column\_names\_logical, row\_numbers\_logical, named\_range\_logical, fetch\_first\_logical)

Connection\_num is the unique connection ID for a data source. The data source specified must have pending query results. Pending query results are generated by a call to SQL.EXEC.QUERY on the same connection.

- If there are no pending results on the connection SQL.RETRIEVE returns the #N/A error value.
- If connection\_num is not valid, SQL.EXEC.QUERY returns the #VALUE! error value.

Destination\_ref specifies where the results should be placed. It is either a reference to a single cell or it is omitted.

- If destination\_ref refers to a single cell then SQL.RETRIEVE will return all of the pending results in the cells to the right, below, and including destination\_ref. This is the same convention used in Microsoft Excel when multiple cells are pasted into a single-cell selection. Any previous values contained in the destination cells will be overwritten without confirmation.
- If destination\_ref is omitted then the bindings established by previous calls to SQL.BIND will be used to return results. If no such bindings exist for the current connection then SQL.RETRIEVE will return the #REF! error value. If a particular result column has not been bound then its results will be discarded. Max\_rows specifies the number of rows that will be returned under each bound column. The first row of results will be placed in the bound cell and any additional rows will be placed in the rows immediately under the bound cell.

Max\_columns is the maximum number of columns to be retrieved. It is only used when destination\_ref is not omitted.

- If max\_columns specifies more columns than are available in the results, SQL.RETRIEVE will place data in the columns for which data is available and clear the additional columns.
- If max\_columns specifies fewer columns than are available in the results, the rightmost result columns will be discarded to fit the chosen size. Column position will be determined by the order in which the data source returned them.
- If max\_columns is omitted then all of the result columns will be returned.

Max\_rows is the maximum number of rows to be returned.

- If max\_rows specifies more rows than are available in the results, SQL.RETRIEVE will place data in the rows for which data is available and clear the additional rows.
- If max\_rows specifies fewer rows than are available in the results, SQL.RETRIEVE will place data in the selected rows but will not discard the additional rows. These extra rows can be retrieved via additional calls to SQL.RETRIEVE. This process is described in the fetch\_first\_logical argument description.
- If max\_rows is omitted then all rows in the result set will be returned.

Column\_names\_logical is a logical value which, if TRUE, causes the column names to be returned as the first row of results. If FALSE or omitted, the column names will not be returned.

Row\_numbers\_logical is used only when destination\_ref is included. If row\_numbers\_logical is TRUE then the first column in the result set will contain row numbers. If FALSE then row numbers will not be returned. This column of row numbers will not have a column name and the column heading will be left blank. Row numbers can also be retrieved by binding column number 0 with SQL.BIND.

Named\_range\_logical is a logical value which, if TRUE, sets each column of results to be declared as a named range on the worksheet. The name of the each range will be the result column name. The named range will only include the rows that were fetched with this SQL.RETRIEVE function call. If

FALSE, the results will not be declared as a named range.

Fetch\_first\_logical is a logical value that allows you to request results from the beginning of the result set.

- If the first call to SQL.RETRIEVE did not return all of the rows in the result set then SQL.RETRIEVE may be called again to return the next set of rows. This process can be repeated until no more result rows are available, at which time SQL.RETRIEVE will return the value 0 (zero). This will not halt the running of the macro. During each of these calls, including the first call, fetch\_first\_logical should be set to FALSE.
- If you want to move the cursor back to the beginning of the result set then fetch\_first\_logical should be set to TRUE. This causes the same SQL query text to be executed again on the data source. The cursor will then be positioned at the top of the result set and SQL.RETRIEVE will fill destination\_ref beginning with the first row of results. Further calls to SQL.RETRIEVE, for the purpose of retrieving additional rows, can then be made with fetch\_first\_logical set to FALSE .

### Remarks

- Before calling SQL.RETRIEVE a connection must be established with a data source using SQL.OPEN.
- If SQL.RETRIEVE is unable to retrieve the results on the specified data source then an error value will be returned. In such a case SQL.RETRIEVE will place error information in memory for the SQL.ERROR function, if such information is available.
- If SQL.RETRIEVE is able to successfully return rows of results on the specified connection it will return the number of rows that were actually returned. If there were no results pending on the connection then SQL.RETRIEVE will return the #N/A error value. If no data was found then SQL.RETRIEVE returns 0 (zero).
- A successful call to SQL.OPEN returns a unique connection ID number, which is used in a call to SQL.EXEC.QUERY to send a SQL language query. Following this call to SQL.EXEC.QUERY, SQL.RETRIEVE uses the same connection ID number to retrieve query results.

### Example

SQL.RETRIEVE(conn1,sheet1!C1,1) stores data obtained from the data source conn1 on Sheet1 from left to right in cell C1, using only column 1.

### Related Functions

|                                             |                                                   |
|---------------------------------------------|---------------------------------------------------|
| <u>SQL.OPEN</u>                             | Establishes a connection with a data source       |
| <u>SQL.EXEC.QUERY</u>                       | Sends a query to a data source                    |
| <u>SQL.BIND</u>                             | Specifies storage for a result column             |
| <u>SQL.RETRIEVE.TO.FILE</u>                 | Retrieves query results and places them in a file |
| <u>SQL.GET.SCHEMA</u>                       | Gets information about a connected data source.   |
| <u>SQL.CLOSE</u>                            | Close a data source connection                    |
| <u>SQL.ERROR</u>                            | Returns detailed error information                |
| List of <u>Command-Equivalent Functions</u> |                                                   |

## SQL.RETRIEVE.TO.FILE

Macro Sheets Only

Retrieves all of the results from a previously executed query and places them in a file. The connection used must have already been established using the macro function SQL.OPEN. Also, a query must already have been executed using SQL.EXEC.QUERY and results must be pending.

If this function is not available, you must install the Microsoft ODBC add-in (XLODBC.XLA). For more information, see "Installing Add-ins" in Chapter 37 of the Microsoft Excel User's Guide, or [Installing or removing an add-in](#) in online Help.

### Syntax

**SQL.RETRIEVE.TO.FILE(connection\_num, destination, column\_names\_logical, column\_delimiter)**

Connection\_num is the unique connection ID for a data source. The data source specified must have query results pending. Pending results were generated by a previous call to SQL.EXEC.QUERY on the same connection.

- If there are no pending results on the connection SQL.RETRIEVE.TO.FILE returns the #N/A error value. The file is not affected.

- If connection\_num is not valid, SQL.RETRIEVE.TO.FILE returns the #VALUE! error value.

Destination specifies the name and path of the file where the results should be placed.

SQL.RETRIEVE.TO.FILE will open the specified file and fill it with the entire result set.

- The format of the data in the file will be compatible with the Microsoft Excel ".CSV" format. The overall format will be that columns will be separated by the value in column\_delimiter (see below) and the individual rows will be separated by a linefeed/carriage-return.

- If the file specified by destination cannot be opened then the error value #N/A will be returned by SQL.RETRIEVE.TO.FILE.

- If the file already exists its previous contents will be overwritten by SQL.RETRIEVE.TO.FILE.

Column\_names\_logical is a logical value that, if TRUE, allows the column names to be returned as the first row of data. If FALSE or omitted, the column names will not be returned.

Column\_delimiter is the value that will be used to separate the elements in each row. If column\_delimiter is omitted then a tab will be used. If another value is desired then it should be enclosed in quotation marks. Possible values for column\_delimiter might be: ",", " or ";", or " ". The string "tab" can also be used to specify a tab separator (even though this is redundant, since a tab is the default).

### Remarks

- If SQL.RETRIEVE.TO.FILE is unable to retrieve the results on the specified connection then an error value will be returned. In such a case SQL.RETRIEVE.TO.FILE will place error information in memory for the SQL.ERROR function, if such information is available.

- If SQL.RETRIEVE.TO.FILE is able to successfully return rows of results on the specified connection and place them in a file it will return the number of rows that were actually written to the file. If there were no results pending on the connection then SQL.RETRIEVE.TO.FILE will return the #N/A error value and the file will not be created or modified.

- Before calling SQL.RETRIEVE.TO.FILE a connection must be established with a data source using SQL.OPEN.

- A successful call to SQL.OPEN returns a unique connection ID number, which can be used in a call to SQL.EXEC.QUERY to send a SQL language query. Following this call to SQL.EXEC.QUERY, SQL.RETRIEVE.TO.FILE uses the same connection ID number to retrieve query results and place them in a file.

### Example

SQL.RETRIEVE.TO.FILE(conn1,"C:\MSQUERY\RESULTS1.QRY",TRUE,",") retrieves the results of a previously executed query and places them in the file RESULTS1.QRY, with column names that are comma delimited.

### Related Functions

[SQL.OPEN](#)

Establishes a connection with a data source

[SQL.EXEC.QUERY](#)

Sends a query to a data source

SQL.BIND

Specifies storage for a result column

SQL.RETRIEVE

Retrieves query results

SQL.GET.SCHEMA

Gets information about a connected data source.

SQL.CLOSE

Closes a data source connection

SQL.ERROR

Returns detailed error information

List of Command-Equivalent Functions



## SUBTOTAL.CREATE

Equivalent to choosing the Subtotals command from the Data menu. Generates a subtotal in a list or database. For more information on subtotalling within a list, see "Summarizing Data in a List" in Chapter 22 of the Microsoft Excel User's Guide or [Subtotal Commands](#) in online Help.

### Syntax

**SUBTOTAL.CREATE**(at\_change\_in, function\_num, total, replace, pagebreaks, summary\_below)

**SUBTOTAL.CREATE?**(at\_change\_in, function\_num, total, replace, pagebreaks, summary\_below)

At\_change\_in is a column offset corresponding to the At Each Change In text box on the Subtotal dialog box.

Function\_num is a number corresponding to the Use Function list box specifying which function you want to use in subtotalling your data.

| Function | Function_Num |
|----------|--------------|
|----------|--------------|

|         |    |
|---------|----|
| SUM     | 1  |
| COUNTA  | 2  |
| AVERAGE | 3  |
| MAX     | 4  |
| MIN     | 5  |
| PRODUCT | 6  |
| COUNT   | 7  |
| STDEV   | 8  |
| STDEVP  | 9  |
| VAR     | 10 |
| VARP    | 11 |

Total is an Array of column offsets corresponding to the Add Subtotal To list box. Indicates which columns you want aggregated according to function\_num; for example, {4,5}

Replace is a logical value which, if TRUE, causes any previous subtotals to be replaced by new subtotals. If FALSE or omitted, subtotals will not be replaced.

Pagebreaks is a logical value corresponding to the Page Break Between Groups check box which, if TRUE, creates a page break below each subtotal. If FALSE or omitted, does not create a page break below each subtotal.

Summary\_below is a logical value corresponding to the Summary Below Data check box which, if TRUE, places subtotal rows below the data they refer to, and a grand total at the bottom of the list. If FALSE, places subtotal rows above the data they refer to, and a grand total just below the header.

### Related Functions

[SUBTOTAL.REMOVE](#) Removes all previously existing subtotals and grand totals in a list

List of [Command-Equivalent Functions](#)

## **SUBTOTAL.REMOVE**

Equivalent to choosing the Subtotal command from the Data menu, and then selecting the Remove All button in the Subtotal dialog box. Removes all previously existing subtotals and grand totals in a list. Any page breaks and outlines will also be removed.

### **Syntax**

**SUBTOTAL.REMOVE()**

### **Related Functions**

SUBTOTAL.CREATE      Generates a subtotal in a list or database

List of Command-Equivalent Functions

## SUMMARY.INFO

Equivalent to choosing the Summary Info command from the File menu. Generates the summary information for the active workbook.

### Syntax

**SUMMARY.INFO**(title, subject, author, keywords, comments)

**SUMMARY.INFO?**(title, subject, author, keywords, comments)

The arguments correspond to the text boxes in the Summary Info dialog boxes. If any arguments are omitted, that text box is left empty.

Title specifies a title for the file, not necessarily a file name. Long names can be entered, up to 255 characters.

Subject is information pertaining to the subject matter of the workbook.

Author is initially the name specified in the General tab of the Options dialog box, which appears from the Options command from the Tools menu. If this is omitted, the registered user of the copy of Microsoft Excel will be used.

Keywords are keywords that can be later used in searching for the contents in the file.

Comments is a comment that can be entered to help a user learn more about the contents or subject matter of the workbook.

### Related Functions

FIND.FILE Lets you search for files based on criteria such as author or creation date

GET.WORKBOOK Returns information about a workbook document

List of Command-Equivalent Functions

## TEXT.TO.COLUMNS

Equivalent to choosing the Text To Columns command from the Data menu when a column of data is to be separated into multiple columns. Parses text into columns of data.

### Syntax

**TEXT.TO.COLUMNS**(destination\_ref, data\_type, text\_delim, consecutive\_delim, tab, semicolon, comma, space, other, other\_char, field\_info)

The following arguments correspond to check boxes, option buttons and text buttons in the Text To Columns Wizard, which is started with the Text To Columns command on the Data menu.

Destination\_Ref is a single cell reference to specifies where to place the converted text.

Data\_Type is a number specifying whether that data is delimited (1) or fixed width (2)

Text\_Delim denotes how text strings are represented, and can be one of the following values:

| Number | Text_Delim |
|--------|------------|
| 1      | "          |
| 2      | '          |
| 3      | none       |

Consecutive\_delim is a logical value corresponding to the Treat Consecutive Delimiters as One check box which, if TRUE, allows consecutive delimiters (such as ",,," to be treated as a single delimiter. If FALSE, all consecutive delimiters are considered separate column breaks.

Tab is a logical value which, if TRUE, specifies that the column has tab delimiters. If FALSE, the column does not have tab delimiters. Tab is ignored if data\_type is 2.

Semicolon is a logical value which, if TRUE, specifies that the column has semicolon delimiters. If FALSE, the column does not have semicolon delimiters. Semicolon is ignored if data\_type is 2.

Comma is a logical value which, if TRUE, specifies that the column has comma delimiters. If FALSE, the column does not have comma delimiters. Comma is ignored if data\_type is 2.

Space is a logical value which, if TRUE, specifies that the column has space delimiters. If FALSE, the column does not have space delimiters. Space is ignored if data\_type is 2.

Other is a logical value which, if TRUE, specifies that the column has custom delimiters. If FALSE, the column does not have custom delimiters. Other is ignored if data\_type is 2.

Other\_Char is a single character that specifies a delimiter not included in the list of delimiters.

Other\_Char is ignored if data\_type is 2 and if the argument other is FALSE.

Field\_Info is an array which consists of the following elements: "column number, data\_format", if data\_type is 1; or "start\_pos, data\_format" if data\_type is 2. The second number defines the column's data format, and can be one of the following.

| 2nd Number | Data Format                 |
|------------|-----------------------------|
| 1          | General                     |
| 2          | Text                        |
| 3          | Date, in the form MDY       |
| 4          | Date, in the form DMY       |
| 5          | Date, in the form YMD       |
| 6          | Date, in the form MYD       |
| 7          | Date, in the form DYM       |
| 8          | Date, in the form YDM       |
| 9          | Do not import column (skip) |

### Related Functions

List of [Command-Equivalent Functions](#)

## **TRACER.CLEAR**

Macro Sheets Only

Equivalent to clicking the Remove All Arrows button on the Auditing toolbar on a worksheet. Clears all tracer arrows on the worksheet.

### **Syntax**

**TRACER.CLEAR()**

### **Remark**

Returns the #VALUE! error value if not available; for example, the selection is something other than worksheet.

### **Related Functions**

TRACER.DISPLAY Allows tracer arrow to be displayed showing relationship among cells

List of Command-Equivalent Functions

## TRACER.DISPLAY

Macro Sheets Only

Equivalent to clicking the Trace Precedents or Trace Dependents buttons on the Auditing toolbar on a worksheet. Allows tracer arrow to be graphically displayed showing relationship among cells.

### Syntax

**TRACER.DISPLAY**(direction, create)

Direction is logical value which, if TRUE, displays tracer arrows for precedents. If FALSE tracer arrows for dependents are displayed.

Create is a logical value which, if TRUE displays the next level of tracer arrows in the direction specified by direction. If FALSE, removes the current level of tracer arrows in the direction specified by direction. A level is the number of "arrows" away from the source cell.

### Remark

Returns the #VALUE! error value if not available; for example, the selection is something other than a worksheet, or the cell(s) cannot be traced.

### Related Functions

TRACER.CLEAR Clears all tracer arrows on the worksheet

List of Command-Equivalent Functions

## TRACER.ERROR

Macro Sheets Only

Equivalent to clicking the Tracer Error button on the Auditing toolbar on a worksheet. Allow tracer arrows to be graphically displayed showing which cells are generating an error value.

### Syntax

#### TRACER.ERROR()

Returns TRUE if Microsoft Excel successfully found the cell at which the error occurred. Returns FALSE if an error is not found.

### Remark

- Returns the #VALUE! error value if not available; for example, the selection is something other than worksheet, or cell(s) that cannot be traced.
- If you need to know if there is an error in a cell, use ISERROR().

### Related Functions

IS Functions Tests the type of return values and error values

TRACER.DISPLAY Allows tracer arrow to be displayed

TRACER.CLEAR Clears all tracer arrows on the worksheet

List of Command-Equivalent Functions

## TRACER.NAVIGATE

Macro Sheets Only

Equivalent to double-clicking on a displayed tracer arrow. Moves the selection from one end of a tracer arrow to the other. If it is an error tracer arrow, then the selection goes to the end of the branch.

### Syntax

**TRACER.NAVIGATE**(direction, arrow\_num, ref\_num)

Direction is a logical value which, if TRUE, moves the selection to the arrow endpoint in the precedents direction. If FALSE, moves the selection to the arrow endpoint in the dependents direction.

Arrow\_num is a number specifying which reference a tracer arrow will follow. For example, a 1 indicates that the arrow will follow the first reference in the formula. 1 is the default.

Ref\_num If the arrow is an external reference arrow with multiple links, this argument tells which of the links to follow. Refer to the Links dialog, which is displayed with the Links command from the Edit menu. If ref\_num is 1, the link in the first reference in the Links dialog box will be followed. The default is 1.

### Remarks

- Returns TRUE if successful. Returns FALSE if arrow\_num exceeds the number of tracer arrows or if there are no tracer arrows.
- Returns FALSE if ref\_num exceeds the number of links.
- Returns the #VALUE! error value if not available; for example, if the selection is something other than a worksheet, or the active cell does not contain an arrow.

### Related Functions

TRACER.DISPLAY Allows tracer arrow to be displayed showing which cells formulas in other cells depend on

List of Command-Equivalent Functions



## UNHIDE

Macro Sheets Only

Equivalent to choosing the Unhide command from the Window menu. Use UNHIDE to display hidden windows.

### Syntax

#### **UNHIDE(window\_text)**

Window\_text is the name of the window to unhide. If window\_text is not the name of an open workbook, an error value is returned and the macro is interrupted. You cannot unhide a window of an add-in workbook.

---

**Tip** You can use UNHIDE to activate an embedded chart in order to edit and format it. Use the HIDE function to de-activate the chart window.

---

### Related Functions

GET.WINDOW Returns information about a window

HIDE Hides the active window

List of Command-Equivalent Functions

## UPDATE.LINK

Macro Sheets Only

Equivalent to choosing the Links command from the Edit menu and choosing the Update Now button with a link selected in the Links dialog box. Updates a link to another document. Use UPDATE.LINK to get the newest information from a supporting document.

### Syntax

**UPDATE.LINK**(link\_text, type\_of\_link)

Link\_text is a text string describing the full path of the link as displayed in the Links dialog box. If link\_text is omitted, only links from the active workbook to other Microsoft Excel workbooks are updated.

Type\_of\_link is a number from 1 to 4 that specifies the type of link to update.

| Type_of_link | Link document type   |
|--------------|----------------------|
| 1 or omitted | Microsoft Excel link |
| 2            | DDE link             |
| 3            | Not available        |
| 4            | Not available        |

### Related Functions

CHANGE.LINK Changes supporting links  
GET.LINK.INFO Returns information about a link  
OPEN.LINKS Opens specified supporting documents  
List of Command-Equivalent Functions

## **WORKBOOK.ACTIVATE**

Macro Sheets Only

Equivalent to activating a worksheet by clicking on its tab.

### **Syntax**

**WORKBOOK.ACTIVATE(sheet\_name)**

Sheet\_name is the name of the document you want to activate within the active workbook.

### **Related Function**

WORKBOOK.SELECT Selects one or more sheets for group editing

WORKBOOK.OPTIONS Changes the settings of a workbook document

List of Command-Equivalent Functions

## WORKBOOK.ADD

Macro Sheets Only

Equivalent to choosing the Add button on the workbook contents window in Microsoft Excel version 4. Moves a sheet between workbooks. For Microsoft Excel version 5.0 use WORKBOOK.MOVE.

### Syntax

**WORKBOOK.ADD**(name\_array, dest\_book, position\_num)

**WORKBOOK.ADD?**(name\_array, dest\_book, position\_num)

Name\_array is the name of a sheet, or an array of names of sheets, that you want to move.

Dest\_book is the name of the workbook to which you want to add name\_array. If dest\_book is omitted, it is assumed to be the active workbook.

Position\_num is a number that specifies the position of the document within the workbook.

### Related Function

WORKBOOK.MOVE

Moves one or more sheets between workbooks or changes a sheet's position within a workbook

WORKBOOK.COPY

Copies one or more documents from their current workbook to another workbook

List of Command-Equivalent Functions

## WORKBOOK.COPY

Macro Sheets Only

Equivalent to choosing the Move or Copy Sheet command from the Edit menu. Copies one or more sheets from their current positions to the specified position in the same workbook or to another workbook.

### Syntax

**WORKBOOK.COPY**(name\_array, dest\_book, position\_num)

**WORKBOOK.COPY?**(name\_array, dest\_book, position\_num)

Name\_array is the name of a sheet, or an array of names of sheets, that you want to copy in the currently active workbook.

Dest\_book is the name of the workbook to which you want to copy name\_array. If dest\_book is the current workbook, name\_array is copied within the workbook. If dest\_book is omitted, the copy of name\_array becomes a separate workbook.

Position\_num is a number that specifies the target position for the sheet within the new workbook. The first position is 1.

- If position\_num is specified, Microsoft Excel inserts the copy of the sheet at the specified position in the workbook.
- If position\_num is omitted, Microsoft Excel places the sheet at the last position in the workbook.
- If dest\_book is omitted, position\_num is ignored.

### Remarks

- If the structure of the workbook is protected, you cannot copy sheets within the workbook or to another workbook.
- You cannot copy a hidden sheet.

### Related Function

WORKBOOK.MOVE

Moves one or more documents from one workbook to another workbook or to another position in the same workbook

List of Command-Equivalent Functions

## WORKBOOK.DELETE

Macro Sheets Only

Equivalent to choosing the Delete Sheet command from the Edit menu. Deletes a sheet or group of sheets from the current workbook.

### Syntax

**WORKBOOK.DELETE**(sheet\_text)

Sheet\_text is the name of the sheet to delete. If omitted, the currently active sheet or sheets is deleted.

### Remarks

- This function prompts for confirmation. To suppress the prompt, use the ERROR function. For example, =ERROR(FALSE).
- If the structure of the workbook is protected, you cannot delete any of its sheets.
- If you want to delete Sheet1:Sheet10, you must select them first with WORKBOOK.SELECT(). You can also place the sheets in an array first, as in {"Sheet1", "Sheet2", "Sheet3",...}.
- You cannot delete the last visible sheet in a workbook.

### Related Functions

List of Command-Equivalent Functions

## WORKBOOK.HIDE

Macro Sheets Only

Equivalent to choosing Sheet command from the Format menu, and then choosing Hide from the Sheet submenu. Hides sheets in the active workbook.

### Syntax

**WORKBOOK.HIDE**(sheet\_text, very\_hidden)

Sheet\_text is the name of the sheet to hide. If omitted, the currently selected sheet(s) are hidden.

Very\_hidden specifies how the sheet is hidden. If TRUE, then the sheet name does not appear in the Unhide dialog box. After using this argument, use WORKBOOK.UNHIDE to unhide the sheet. If FALSE or omitted, hides the sheet but does not prevent the sheet's name from appearing in the Unhide dialog box.

### Remarks

- If the structure of the workbook is protected, you cannot hide any sheets in the workbook.
- You cannot hide the last visible sheet in a workbook.
- To hide Sheet1:Sheet10, select them first with the WORKBOOK.SELECT function. You can also place the sheets in an array first, as in {"Sheet1", "Sheet2", "Sheet3",...}.

### Related Function

List of [Command-Equivalent Functions](#)

## WORKBOOK.INSERT

Macro Sheets Only

Equivalent to choosing the Worksheet, Chart, or Macro commands from the Insert menu. Inserts one or more new sheets into the current workbook.

### Syntax

**WORKBOOK.INSERT**(type\_num)

**WORKBOOK.INSERT?**(type\_num)

**Type\_num** specifies the type of sheet to insert.

| Type_num    | Type of sheet                       |
|-------------|-------------------------------------|
| 1           | Worksheet                           |
| 2           | Chart                               |
| 3           | Excel 4 Macro Sheet                 |
| 4           | Excel 4 International Macro Sheet   |
| 5           | (Reserved)                          |
| 6           | Microsoft Excel Visual Basic Module |
| 7           | Dialog                              |
| quoted text | Template                            |

If omitted, the type of the active sheet is used.

If the current selection contains one sheet, then only one sheet is inserted. If the selection contains more than one sheet and the active sheet is a worksheet, then an equal number of sheets is inserted to the left of the selected group of sheets.

### Remarks

- The new sheets are always inserted to the left of the current selection.
- If the workbook structure is protected, you cannot insert new sheets.

### Related Function

List of [Command-Equivalent Functions](#)



## WORKBOOK.MOVE

Macro Sheets Only

Equivalent to choosing the Move or Copy Sheet command from the Edit menu. Moves one or more sheets between workbooks or changes a sheet's position within a workbook.

### Syntax

**WORKBOOK.MOVE**(name\_array, dest\_book, position\_num)

**WORKBOOK.MOVE?**(name\_array, dest\_book, position\_num)

Name\_array is the name of a sheet, or an array of names of sheets, that you want to move.

Dest\_book is the name of the workbook to which you want to move name\_array. If dest\_book is omitted, WORKBOOK.MOVE moves the sheet out of the workbook and puts it in a new separate workbook. If dest\_book is the same as the current book, then the sheet is moved within the workbook.

Position\_num is a number that specifies the target position for the sheet within dest\_book. The first position is 1.

- If position\_num is specified, Microsoft Excel inserts the sheet at the specified position in the workbook.
- If position\_num is omitted, Microsoft Excel moves the sheet to the last position in the workbook.
- If you move the last sheet out of a workbook, the workbook closes.

### Related Functions

WORKBOOK.COPY

Copies one or more documents from their current workbook into another workbook

List of Command-Equivalent Functions

## WORKBOOK.NAME

Macro Sheets Only

Equivalent to choosing the Rename command on the Sheet submenu from the Format menu. Renames a sheet in a workbook.

### Syntax

**WORKBOOK.NAME**(oldname\_text, newname\_text)

**WORKBOOK.NAME?**(oldname\_text, newname\_text)

Oldname\_text is the name of the sheet that you want to rename.

Newname\_text is the new name of the sheet.

### Remarks

- If you try to rename a sheet using a sheet name that already exists in the workbook, Microsoft Excel displays an error message and interrupts the macro.
- If the structure of the workbook is protected, you cannot rename any of the sheets in the workbook.

### Related Function

List of [Command-Equivalent Functions](#)

## **WORKBOOK.NEW**

Macro Sheets Only

Adds a sheet to a workbook. This function is for compatibility with Microsoft Excel version 4.0. To add a new sheet to a workbook in Microsoft Excel version 5, use the [WORKBOOK.INSERT](#) function.

### **Syntax**

**WORKBOOK.NEW()**

**WORKBOOK.NEW?()**

In both forms of this function, you will be prompted for the name of the workbooks.

### **Related Function**

[WORKBOOK.INSERT](#)      Adds sheets to workbooks

List of [Command-Equivalent Functions](#)

## **WORKBOOK.NEXT**

Macro Sheets Only

Activates the next sheet in the active workbook.

### **Syntax**

**WORKBOOK.NEXT()**

### **Remarks**

- If the last sheet in the workbook is active, this function has no effect.
- This function skips over hidden sheets in the workbook.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## WORKBOOK.OPTIONS

Macro Sheets Only

Equivalent to selecting the Options button in a workbook contents window in Microsoft Excel version 4. This function is for compatibility with Microsoft Excel version 4. To change the name of a sheet in Microsoft Excel version 5, use the WORKBOOK.NAME function.

### Syntax

**WORKBOOK.OPTIONS**(sheet\_name, bound\_logical, new\_name)

Sheet\_name is the name of a sheet.

Bound\_logical is for compatibility with Microsoft Excel version 4. In Microsoft Excel version 5, returns an error if FALSE.

New\_name is the sheet name to assign to sheet\_name. If new\_name is omitted, then the sheet name is not changed.

### Related Functions

GET.WORKBOOK

Returns information about a workbook document

WORKBOOK.NAME

Changes the name of a sheet in a workbook

WORKBOOK.SELECT

Selects the specified documents in a workbook

List of Command-Equivalent Functions

## **WORKBOOK.PREV**

Activates the previous sheet in the workbook.

### **Syntax**

**WORKBOOK.PREV()**

### **Remarks**

- If the first sheet in the workbook is active, this function has no effect.
- This function skips over hidden sheets in the workbook.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## WORKBOOK.PROTECT

Equivalent to choosing the Protect Workbook command from the Protection submenu on the Data menu. Controls protection of workbooks.

### Syntax

**WORKBOOK.PROTECT**(structure, windows, password)

**WORKBOOK.PROTECT?**(structure, windows, password)

Structure specifies whether the structure of the workbook is protected. If TRUE, the structure is protected. If FALSE or omitted, the structure is not protected.

Windows specifies whether the windows of the workbook are protected. If TRUE, the windows are protected. If FALSE or omitted, the windows are not protected.

Password specifies whether to protect the workbook with a password. If omitted no password is used. When recording a macro, this argument is not recorded. In the dialog box form of this function, you can specify a password.

### Remarks

To protect a sheet in a workbook, use the PROTECT.DOCUMENT function.

### Related Functions

PROTECT.DOCUMENT Protects a sheet in a workbook

List of Command-Equivalent Functions

## **WORKBOOK.SCROLL**

Scrolls through the sheets in a workbook.

### **Syntax**

**WORKBOOK.SCROLL**(num\_sheets, firstlast\_logical)

Num\_sheets. is a number specifying how many sheets to scroll and the direction of scrolling. Positive numbers scroll forward by that many sheets. Negative numbers scroll backward by that many sheets. Zero (0) does not scroll.

Firstlast\_logical specifies whether to scroll to the first or last sheet in the workbook. If TRUE, scrolls to the last sheet in the workbook. If FALSE, scrolls to the first sheet in the workbook. If this argument is specified, then num\_sheets is ignored.

### **Related Functions**

List of [Command-Equivalent Functions](#)



## WORKBOOK.SELECT

Macro Sheets Only

Equivalent to selecting a sheet or group of sheets in the active workbook. If you select a group of sheets, subsequent commands effect all the sheets in the group.

### Syntax

**WORKBOOK.SELECT**(name\_array, active\_name, replace)

Name\_array is a horizontal array of text names of sheets you want to select. If name\_array is omitted, no sheets are selected.

Active\_name is the name of a single sheet in the workbook that you want to be the active sheet. If active\_name is omitted, the first sheet in name\_array is made the active sheet.

Replace specifies whether the currently selected sheets or macrosheets are to be replaced by name\_array. If TRUE or omitted, then the current sheet selection is replaced by name\_array. If FALSE, then name\_array will be appended to the current sheet.

### Related Functions

GET.WORKBOOK Returns information about a workbook

SELECT Selects a cell, worksheet object, or chart item

List of Command-Equivalent Functions

## **WORKBOOK.TAB.SPLIT**

Sets the ratio of the tabs to the horizontal scrollbar.

### **Syntax**

**WORKBOOK.TAB.SPLIT**(ratio\_num)

Ratio\_num is the ratio of tabs to horizontal scroll, as a decimal value between 0 and 1. If omitted defaults to .6.

### **Remarks**

- If the structure of the workbook is protected, you cannot use this function.
- Use GET.WINDOW(28) to find out what the current ratio is.

### **Related Functions**

GET.WINDOW Returns information about a

List of Command-Equivalent Functions

## WORKBOOK.UNHIDE

Equivalent to choosing the Unhide command from the sheet submenu on the Format menu. Unhides one or more sheets in the current workbook.

### Syntax

**WORKBOOK.UNHIDE**(sheet\_text)

**WORKBOOK.UNHIDE?**(sheet\_text)

Sheet\_text. specifies the sheet that you want to unhide. If sheet\_text is omitted, then this function unhides the sheets in the order that they would appear in the workbook.

### Remarks

- If the workbook is protected, you cannot unhide any sheets in the book.

### Related Functions

WORKBOOK.HIDE Hides sheets in the active workbook

List of Command-Equivalent Functions

## WORKGROUP

Macro Sheets Only

Equivalent to choosing the Group Edit command from the Options menu in Microsoft Excel version 4. Creates a group. This function is provided for compatibility only. In Microsoft Excel version 5, you can create a group by using the WORKBOOK.SELECT function.

### Syntax

**WORKGROUP**(name\_array)

**WORKGROUP?**(name\_array)

Name\_array is the list of workbooks or sheets in workbooks that you want grouped.

- If name\_array is omitted, the most recently created group is recreated.
- If no group has been created during the current Microsoft Excel session, all open, unhidden worksheets are created as a group.
- If you specify just the name of a workbook, WORKGROUP adds the first sheet of the workbook to the group.

### Remarks

WORKGROUP returns the #VALUE! error value and interrupts the macro if it can't find any of the sheets in name\_array or if any of the sheets is a chart or module.

### Related Functions

FILL.GROUP

Fills the contents of the active worksheet's selection to the same area on all other worksheets in the group

WORKBOOK.SELECT

Selects one or more sheets in a workbook

List of Command-Equivalent Functions

## WORKSPACE

Macro Sheets Only

Changes the workspace settings for a workbook. This function is provide for compatibility with Microsoft Excel version 4.0 only. In Microsoft Excel version 5, you can change workbook settings with OPTIONS.GENERAL function.

### Syntax

**WORKSPACE**(fixed, decimals, r1c1, scroll, status, formula, menu\_key, remote, entermove, underlines, tools, notes, nav\_keys, menu\_key\_action, drag\_drop, show\_info)

**WORKSPACE?**(fixed, decimals, r1c1, scroll, status, formula, menu\_key, remote, entermove, underlines, tools, notes, nav\_keys, menu\_key\_action, drag\_drop, show\_info)

Arguments correspond to check boxes and text boxes in the Workspace dialog box. Arguments corresponding to check boxes are logical values. If an argument is TRUE, the check box is selected; if FALSE, the check box is cleared; if omitted, the current setting is not changed.

Fixed corresponds to the Fixed Decimal check box.

Decimals specifies the number of decimal places. Decimals is ignored if fixed is FALSE or omitted.

R1c1 corresponds to the R1C1 check box.

Scroll corresponds to the Scroll Bars check box.

Status corresponds to the Status Bar check box.

Formula corresponds to the Formula Bar check box.

Menu\_key is a text value indicating an alternate menu key, and corresponds to the Alternate Menu Or Help Key box.

Remote corresponds to the Ignore Remote Requests check box.

---

**Important** Microsoft Excel for the Macintosh requires system software version 7.0 or later for this argument.

---

Entermove corresponds to the Move Selection After Enter/Return check box.

Underlines is a number corresponding to the Command Underline options as shown in the following table.

---

**Note** This argument is only available in Microsoft Excel for the Macintosh.

---

| If underlines is | Command underlines are |
|------------------|------------------------|
| 1                | On                     |
| 2                | Off                    |
| 3                | Automatic              |

Tools is a logical value. If TRUE, the Standard toolbar is displayed; if FALSE, all visible toolbars are hidden. If omitted, the current toolbar display is not changed.

Notes corresponds to the Note Indicator check box.

Nav\_keys corresponds to the Alternate Navigation Keys check box. In Microsoft Excel for the Macintosh, nav\_keys is ignored.

Menu\_key\_action is the number 1 or 2 specifying options for the alternate menu or Help key. In Microsoft Excel for the Macintosh, menu\_key\_action is ignored.

| Menu_key_action | Alternate menu or Help key activates             |
|-----------------|--------------------------------------------------|
| 1 or omitted    | Microsoft Excel menus                            |
| 2               | Lotus 1-2-3 Help                                 |
| Drag_drop       | corresponds to the Cell Drag And Drop check box. |
| Show_info       | corresponds to the Info Window check box.        |

### Related Function

GET.WORKSPACE

Returns information about the workspace

## List of Command-Equivalent Functions

## ADD.LIST.ITEM

Macro Sheets Only

Adds an item in a list box or drop-down control on a worksheet or dialog sheet control.

### Syntax

**ADD.LIST.ITEM**(text, index\_num)

Text specifies the text of the item to be added. Instead of text, an empty string may be inserted.

Index\_num is the list index to be used for the new item. Blank entries are created from the end of the current list to the new item index. If index\_num is omitted the new item is appended to the list.

### Remarks

If the list box or drop-down box was already filled using the LISTBOX.PROPERTIES function, then adding an item with ADD.LIST.ITEM causes the fillrange contents to be discarded in favor of the new list.

### Related Functions

REMOVE.LIST.ITEM

Removes an item in a list box or drop-down box

SELECT.LIST.ITEM

Selects an item in a list box or in a group box

List of Customizing Functions

## CHECKBOX.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when a check box or option button control is selected on a worksheet or dialog sheet. Sets various properties of check box and option box controls.

### Syntax

**CHECKBOX.PROPERTIES**(value, link, accel\_text, accel\_text2, 3d\_shading)

**CHECKBOX.PROPERTIES?**(value, link, accel\_text, accel\_text2, 3d\_shading )

Value is the value of the check box or option button setting that determines whether it is selected or not.

| Value      | Box or Button Setting |
|------------|-----------------------|
| 0 or FALSE | Off                   |
| 1 or TRUE  | On                    |
| 2          | Mixed                 |

Link is the cell on the sheet to which the check box or option button value is linked. Whenever one of these two controls is changed, the value of the control is entered into the cell. Similarly, whenever the value in the cell is changed, the setting for the corresponding check box or option button is also changed. To clear the link, set this value to an empty string. For example, entering "TRUE" into a cell linked to a check box will select that check box.

3d\_shading is a logical value that specifies whether the check box appears as 3-D. If TRUE, the check box will appear as 3-D. If FALSE or omitted, the check box will not be 3-D. This argument is available for only worksheets.

Accel\_text is a text string containing the character to use as the control's accelerator key on a dialog sheet. The character is matched against the text of the control, and the first matching character is underlined. When the user presses ALT+accel\_text in Microsoft Excel for Windows or COMMAND+accel\_text in Microsoft Excel for the Macintosh, the control is clicked.

Accel\_text2 is a text string containing the second accelerator key on a dialog sheet. This argument is for only Far East versions of Microsoft Excel.

### Remarks

Only controls on dialog sheets can have accelerator keys. Worksheet controls cannot have accelerator keys.

### Related Functions

[PUSHBUTTON.PROPERTIES](#)

Sets the properties of the push button control

[EDITBOX.PROPERTIES](#)

Sets the properties of an edit box on a worksheet or dialog sheet

[LABEL.PROPERTIES](#)

Sets the accelerator property of the label and group box control

[LISTBOX.PROPERTIES](#)

Sets the properties of a list box and drop-down box controls on a worksheet or dialog sheet

List of [Customizing Functions](#)



## EDITBOX.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when an edit box is selected on a worksheet or dialog sheet. Sets the properties of an edit box on a dialog sheet.

### Syntax

**EDITBOX.PROPERTIES**(validation\_num, multiline\_logical, vscroll\_logical)

**EDITBOX.PROPERTIES?**(validation\_num, multiline\_logical, vscroll\_logical)

Validation\_num is the validation applied to the edit box when the dialog is dismissed. If the edit box contains a value other than the type specified (or validation), an error is returned.

| Validation_num | Type                           |
|----------------|--------------------------------|
| 1              | Text                           |
| 2              | Integer                        |
| 3              | Number (allows floating point) |
| 4              | Reference                      |
| 5              | Formula                        |

Multiline\_logical is a logical value specifying whether word wrapping is allowed in the edit box control. If TRUE, word wrapping is allowed. If FALSE, word wrapping is not allowed

Vscroll\_logical is a logical value specifying whether edit box displays a vertical scrollbar. If TRUE, a scrollbar is displayed. If FALSE, a scrollbar is not displayed.

### Related Functions

[CHECKBOX.PROPERTIES](#)

Sets various properties of check box and option box controls

[PUSHBUTTON.PROPERTIES](#)

Sets the properties of the push button control

List of [Customizing Functions](#)

## ENABLE.OBJECT

Macro Sheets Only

Enables or disables a drawing object or the selected drawing object. A disabled object will not run any macro events assigned to it, and the controls will be grayed out.

### Syntax

**ENABLE.OBJECT**(object\_id\_text, enable\_logical)

Object\_id\_text is the name of the object(s) as text. If omitted, the selected object(s) are assumed.

Enable\_logical is a logical value that specifies whether the object is to be enabled. If TRUE, the object is enabled. If FALSE, the object is disabled.

### Examples

ENABLE.OBJECT("Button 2", FALSE) disables the button with object name Button 2 on the dialog box.

### Related Functions

SET.CONTROL.VALUE

Changes the value of the active control

List of Customizing Functions

## HIDE.DIALOG

Macro Sheets Only

Closes the dialog box that has the current focus.

### Syntax

**HIDE.DIALOG**(cancel\_logical)

Cancel\_logical is a logical value that specifies whether to close the dialog box and validate any edit boxes. If FALSE, the dialog box is closed and edit boxes are validated, or checked to determine if they contain a valid data type. If TRUE, the dialog box is closed and the edit boxes are not validated.

### Remarks

If the edit box does not contain a valid data type when the dialog box is closed, the dialog will remain open. For example, if the edit box is supposed to contain integer values, and a text value is entered, the dialog box will not close. This applies to only those dialog boxes that must be closed before any further user action can happen.

### Examples

`HIDE.DIALOG (FALSE)` closes the dialog box and checks to see if the edit box contains a valid data type (validated)

### Related Functions

[EDITBOX.PROPERTIES](#)

Sets the properties of an edit box on a worksheet or dialog sheet

[SHOW.DIALOG](#)

Runs a dialog on a dialog sheet

List of [Customizing Functions](#)

## **LABEL.PROPERTIES**

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when a label control is selected on a worksheet or dialog sheet. Sets the accelerator property of the label and group box controls.

### **Syntax**

**LABEL.PROPERTIES**(accel\_text, accel\_text2, 3d\_shading)

**LABEL.PROPERTIES?**(accel\_text, accel\_text2, 3d\_shading)

Accel\_text is a text string containing the character to use as the label's accelerator key on a dialog sheet. The character is matched against the text of the control, and the first matching character is underlined. When the user presses ALT+accel\_text in Microsoft Excel for Windows or COMMAND+accel\_text in Microsoft Excel for the Macintosh, the control is clicked. This argument is ignored for controls on worksheets.

Accel\_text2 is a text string containing the second accelerator key on a dialog sheet. This argument is for only Far East versions of Microsoft Excel.

3d\_shading is a logical value that specifies whether the list box appears as 3-D. If TRUE, the list box will appear as 3-D. If FALSE or omitted, the list box will not be 3-D. This argument is available for only worksheets.

### **Related Functions**

[CHECKBOX.PROPERTIES](#)

Sets various properties of check box and option box controls

[SCROLLBAR.PROPERTIES](#)

Sets the properties of the scroll bar and spinner controls

[PUSHBUTTON.PROPERTIES](#)

Sets the properties of the push button control

List of [Customizing Functions](#)

## LINK.COMBO

Macro Sheets Only

Links an edit box and a list box control into a linked combination box group. The resulting linked controls track each other's selection and contents. Linked edit and list box combinations are similar to an editable drop-down list box, except that the list box is permanently visible and dropped down.

### Syntax

**LINK.COMBO(link\_logical)**

Link\_logical is a logical value that specifies whether the controls are linked or unlinked. If TRUE, the controls will become linked. If FALSE, the controls will be unlinked.

### Remarks

To use this function, first select the list box and edit box to be linked or unlinked. You can do this with `SELECT("list box 1,Edit box 2")`.

### Examples

`LINK.COMBO(FALSE)` will unlink a list box and an edit box.

### Related Functions

[ADD.LIST.ITEM](#)

Adds an item in a list box or drop-down control on a worksheet or dialog sheet control

[SELECT.LIST.ITEM](#)

Selects an item in a list box or in a group box

List of [Customizing Functions](#)

## LISTBOX.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when a list box or drop-down control is selected on a worksheet or dialog sheet. Sets the properties of a list box and drop-down controls on a worksheet or dialog sheet.

### Syntax

**LISTBOX.PROPERTIES**(range, link, drop\_size, multi\_select, 3d\_shading)

**LISTBOX.PROPERTIES?**(range, link, drop\_size, multi\_select, 3d\_shading)

Range is the cell range that the initial contents of the list box are taken from. If blank (empty text), the list box is initially unfilled.

Link is the cell on the sheet to which the list box is linked, and indicates the numeric position of the currently selected item in the list box. Whenever an item in the list box is selected, its numeric position is entered into the linked cell on the sheet.

Drop\_size is the number of lines shown when a drop-down control is dropped. This value is ignored when applied to a non-drop-down list box.

Multi\_select is a number that specifies the selection mode of the list box. Zero is single selection. 1 is simple multi-select. 2 is extended multi-select.

3d\_shading is a logical value that specifies whether the list box appears as 3-D. If TRUE, the list box will appear as 3-D. If FALSE or omitted, the list box will not be 3-D. This argument is available for only worksheets.

### Related Functions

[ADD.LIST.ITEM](#)

Adds an item in a list box or drop-down control on a worksheet or dialog sheet control

[SELECT.LIST.ITEM](#)

Selects an item in a list box or in a group box

[CHECKBOX.PROPERTIES](#)

Sets various properties of check box and option box controls

[SCROLLBAR.PROPERTIES](#)

Sets the properties of the scroll bar and spinner controls

[PUSHBUTTON.PROPERTIES](#)

Sets the properties of the push button control

List of [Customizing Functions](#)

## ON.SHEET

Macro Sheets Only

Triggers a macro whenever the specified sheet is activated from another sheet.

### Syntax

**ON.SHEET**(sheet\_text, macro\_text, activate\_logical)

Sheet\_text is the name of the sheet that triggers a macro when it is activated, in the form "[Book1]Sheet1". If omitted, then when any sheet in any book is activated, macro\_text will run.

Macro\_text is the name of the macro to run when the specified sheet is activated. If omitted, then the triggering of a macro on the specified sheet is cancelled.

Activate\_logical is a logical value that specifies if the macro is run when the sheet is activated or deactivated. If TRUE or omitted, the macro is run when the sheet is activated. If FALSE, the macro is run when the sheet is deactivated.

### Examples

ON.SHEET("[STORE.XLS]Sheet1", "WeeklyCalc") will run the macro "WeeklyCalc" when "[STORE.XLS]Sheet1" is activated.

ON.SHEET(, "WeeklyCalc") runs "WeeklyCalc" when any sheet in the book is activated.

ON.SHEET("[STORE.XLS]Sheet1") cancels the triggering of "WeeklyCalc" when Sheet1 in the book STORE.XLS is activated.

ON.SHEET("[STORE.XLS]", "WeeklyCalc") runs "WeeklyCalc" when any sheet in the book STORE.XLS is activated

### Related Functions

ON.WINDOW

Runs a specified macro when you switch to a particular window.

List of Customizing Functions

## OPEN.DIALOG

Macro Sheets Only

Displays the standard Microsoft Excel File Open dialog box with the specified file filters. When the user presses the button specified by `button_text`, the filename the user picks or types in will be returned.

### Syntax

**OPEN.DIALOG**(`file_filter`, `button_text`, `title`, `filter_index`)

`File_filter` is the file filtering criteria to use, as text. For Microsoft Excel for Windows, `file_filter` consists of two parts, a descriptive phrase denoting the file type followed by a comma and then the MS-DOS wildcard file filter specification, as in "Text Files (\*.TXT), \*.TXT, Add-in Files (\*.XLA), \*.XLA". Groups of filter specifications are also separated by commas. Each separate pair is listed in the file type drop-down list box. `File_filter` can include an asterisk (\*) to represent any sequence of characters and a question mark (?) to represent any single character. For Microsoft Excel for the Macintosh, `file_filter` consists of file type codes separated by commas, as in "TEXT,XLA,XLS4". Spaces are significant and should not be inserted before or after the comma separators unless they are part of the file type code.

`Button_text` is the text used for the Open button in the dialog. If omitted, "Open" will be used.

`Button_text` is ignored on Microsoft Excel for Windows.

`Title` specifies the dialog window title. If omitted, "File Open" will be used as the default.

`Filter_index` specifies the index number of the default file filtering criteria from 1 to the number of filters specified in `file_filter`. If omitted or greater than the number of filters present, 1 will be used as the starting index number. The argument is ignored on Microsoft Excel for the Macintosh.

### Remarks

- To use multiple MS-DOS wildcard expressions within `file_filter` for a single file filter type, separate the wildcard expressions with semicolons, as in "VB Files (\*.bas; \*.txt), \*.bas;\*.txt"
- If `file_filter` is omitted, "ALL Files (\*.\*) , \*.\*" will be used as the default in Microsoft Excel for Windows. The default for Microsoft Excel for the Macintosh is all file types.
- If the user cancels the dialog box, FALSE is returned.

### Examples

`OPEN.DIALOG("Text Files (*.TXT), *.TXT, Add-in Files (*.XLA), *.XLA, ALL FILES (*.*) , *.*", "FILE OPENER")` opens a file open dialog box titled "FILE OPENER" with three file filter criteria in the drop-down list box.

### Related Functions

[SAVE.DIALOG](#)

Displays the standard Microsoft Excel File Save As dialog box and gets a file name from the user

List of [Customizing Functions](#)



## PUSHBUTTON.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when a push button control is selected on a worksheet or dialog sheet. Sets the properties of the push button control.

### Syntax

**PUSHBUTTON.PROPERTIES**(default\_logical, cancel\_logical, dismiss\_logical, help\_logical, accel\_text, accel\_text2)

**PUSHBUTTON.PROPERTIES?**(default\_logical, cancel\_logical, dismiss\_logical, help\_logical, accel\_text, accel\_text2)

Default\_logical is a logical value that determines whether the button is the default button for the dialog. If TRUE, the button is the default button. If FALSE, the button is not the default button for the control.

Cancel\_logical is a logical value that determines whether the button is activated when the dialog is cancelled with the Close button or the ESC key. If TRUE, the button is activated when the dialog box is cancelled, and edit boxes are not checked to see if they contain valid data types. If FALSE, the button is not activated when the dialog box is cancelled.

Dismiss\_logical is a logical value that determines whether the button dismisses the dialog when pressed, as when the user presses the box's OK button. If TRUE, the button dismisses the dialog box. If FALSE, the button does not dismiss the dialog box.

Help\_logical is a logical value that determines whether the button is activated when the user presses the F1 key. If TRUE, the button is activated when the user presses the F1 key. If FALSE, the button is not activated when the user presses the F1 key.

Accel\_text is a text string containing the character to use as the dialog button's accelerator key. The character is matched against the text of the control, and the first matching character is underlined. When the user presses ALT+accel\_text in Microsoft Excel for Windows or COMMAND+accel\_text in Microsoft Excel for the Macintosh, the control is clicked. This argument is ignored for push button controls on worksheets.

Accel\_text2 is a text string containing the second accelerator key on a dialog sheet. This argument is for only Far East versions of Microsoft Excel.

### Related Functions

CHECKBOX.PROPERTIES

Sets various properties of check box and option box controls

SCROLLBAR.PROPERTIES

Sets the properties of the scroll bar and spinner controls

EDITBOX.PROPERTIES

Sets the properties of an edit box on a worksheet or dialog sheet

List of Customizing Functions

## REMOVE.LIST.ITEM

Macro Sheets Only

Removes an item in a list box or drop-down box.

### Syntax

**REMOVE.LIST.ITEM**(index\_num, count\_num)

Index\_num specifies the index of the item to remove, from 1 to the number of items in the list.

Specify zero to remove all items in the list.

Count\_num Specifies the number of items to delete starting from index\_num. If omitted, only one item is removed.

### Remarks

If count\_num + index\_num is greater than the number of items in the list, all items starting with index\_num to the end of the list are removed.

### Examples

REMOVE.LIST.ITEM(3,2) removes two items starting with the third item

REMOVE.LIST.ITEM(3) removes only the third item

### Related Functions

[LISTBOX.PROPERTIES](#)

Sets the properties of a list box and drop-down controls on worksheet and dialog sheets

List of [Customizing Functions](#)

## SAVE.DIALOG

Macro Sheets Only

Displays the standard Microsoft Excel File Save As dialog box and gets a file name from the user. This function returns the path and file name of the file that has been saved. Use SAVE.AS to automatically save a file with a particular format and other properties.

### Syntax

**SAVE.DIALOG**(init\_filename, title, button\_text, file\_filter, filter\_index)

**Init\_filename** Specifies the suggested filename for saving. If omitted, the active workbook's name is used, as returned by the GET.DOCUMENT(1) function.

**Title** Specifies the default window title on Microsoft Excel for Windows. For Microsoft Excel for the Macintosh, title specifies the prompt string. If omitted, "File Save As" will be used for Microsoft Excel for Windows, and "Save As:" For Microsoft Excel for the Macintosh.

**Button\_text** is the text used for the save button in the dialog. If omitted, "Save" will be used as the default. This argument is ignored on the Microsoft Excel for Windows.

**File\_filter** is the file filtering criteria to use, as text. For Microsoft Excel for Windows, file\_filter consists of two parts, a descriptive phrase denoting the file type followed by a comma and then the MS-DOS wildcard file filter specification, as in "Text Files (\*.TXT), \*.TXT, Add-in Files (\*.XLA), \*.XLA". Groups of filter specifications are also separated by commas. Each separate pair is listed in the file type drop-down list box. File\_filter can include an asterisk (\*) to represent any sequence of characters and a question mark (?) to represent any single character. For Microsoft Excel for the Macintosh, file\_filter consists of file type codes separated by commas, as in "TEXT,XLA,XLS4". Spaces are significant and should not be inserted before or after the comma separators unless they are part of the file type code.

**Filter\_index** specifies the index number of the default file filtering criteria from 1 to the number of filters specified in file\_filter. If omitted or greater than the number of filters present, 1 will be used as the starting index number. The argument is ignored on Microsoft Excel for the Macintosh.

### Remarks

- To use multiple MS-DOS wildcard expressions within file\_filter for a single file filter type, separate the wildcard expressions with semicolons, as in "VB Filles (\*.bas; \*.txt), \*.bas;\*.txt".
- If file\_filter is omitted, "ALL Files (\*.\*) , \*.\*)" will be used as the default in Microsoft Excel for Windows. The default for Microsoft Excel for the Macintosh is all file types.
- If the user cancels the dialog box, FALSE is returned.

### Examples

```
SAVE.DIALOG("TRAVEL.XLS","How do you want to save this file?",,
"Text Files (*.TXT), *.TXT, Add-in Files (*.XLA), *.XLA, ALL FILES (*.*) ,
.") opens a File Save As dialog box titled "How do you want to save this file?", with "TRAVEL.XLS"
as the suggested file name, and with three file filter criteria in the drop-down list box.
```

### Related Functions

[OPEN.DIALOG](#)

Displays the standard Microsoft Excel File Open dialog box with the specified file filters

List of [Customizing Functions](#)

## SCROLLBAR.PROPERTIES

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu when an edit box is selected on a worksheet or dialog sheet. Sets the properties of the scroll bar and spinner button.

### Syntax

**SCROLLBAR.PROPERTIES**(value, min, max, inc, page, link, 3d\_shading)

**SCROLLBAR.PROPERTIES?**(value, min, max, inc, page, link, 3d\_shading)

Value is the value of the control, and can range from min to max, inclusive. It designates where the scroll bar button is positioned along the scroll bar.

Min is a number specifying the minimum value that the scroll bar can have. This number ranges from 0 to 30,000, but cannot be greater than the maximum value given in max.

Max is a number specifying the maximum value that the scroll bar can have. This number ranges from 0 to 30,000.

Inc is a number specifying the increment that the value is adjusted by when the scrollbar arrow is clicked.

Page is a number specifying the increment that the value is adjusted by when the page scroll region of a scroll bar is clicked.

Link is the cell on the macro sheet to which the scroll bar value is linked. Whenever the scroll bar control is changed, the value of the control is entered into the cell. Similarly, whenever the value in the cell is changed, the setting for the scroll bar is also changed. To clear the link, set this value to an empty string.

3d\_shading is a logical value that specifies whether the scroll bar or spinner button appears as 3-D. If TRUE, the scroll bar or spinner button will appear as 3-D. If FALSE or omitted, the scroll bar or spinner button will not be 3-D. The argument is available for only worksheets.

### Related Functions

[PUSHBUTTON.PROPERTIES](#)

Sets the properties of the push button control

[EDITBOX.PROPERTIES](#)

Sets the properties of an edit box on a worksheet or dialog sheet

List of [Customizing Functions](#)

## SCENARIO.MERGE

Macro Sheets Only

Equivalent to choosing the Scenarios command from the Tools menu and then selecting Merge. This function merges scenarios from other sheets onto the active sheet. A scenario is a set of values to be used as input for a model on your worksheet.

### Syntax

**SCENARIO.MERGE**(source\_file)

**SCENARIO.MERGE?**(source\_file)

Source\_file is the name of the book and sheet from which you want to merge scenarios onto the active sheet.

### Related Functions

SCENARIO.GET Returns the specified information about the scenarios defined on your worksheet

List of Customizing Functions

## **SELECT.LIST.ITEM**

Macro Sheets Only

Selects an item in a list box or in a group box.

### **Syntax**

**SELECT.LIST.ITEM**(**index\_num**, selected\_logical)

Index\_num is the index number of the item to select. Using zero will deselect all items. Adding 1 to the number of items in the list will select all the items specified.

Selected\_logical is a logical value that specifies whether the item in a multi-select list box should be selected or not. If TRUE, the item is selected. If FALSE, it is not selected.

### **Related Functions**

[ADD.LIST.ITEM](#)

Adds an item in a list box or drop-down control on a worksheet or dialog sheet control

[REMOVE.LIST.ITEM](#)

Removes an item in a list box or drop-down box

List of [Customizing Functions](#)

## SET.CONTROL.VALUE

Macro Sheets Only

Equivalent to choosing the Object command from the Format menu and changing the value for the active control, such as a list box, drop-down box, check box, option button, scroll bar, and spinner button.

### Syntax

**SET.CONTROL.VALUE**(value)

Value is the value you want to change. The control interprets this value as follows:

| Control        | Value is                                                                 |
|----------------|--------------------------------------------------------------------------|
| List box       | The index of the selected item. If zero, then no item is selected.       |
| Drop-down box  | The index of the selected item. If zero, then no item is selected.       |
| Check box      | 0 = Off<br>1 = On<br>2 = Mixed                                           |
| Option button  | 0= Off<br>1 = On                                                         |
| Scroll bar     | The numeric value of the control, between the maximum and minimum values |
| Spinner button | The numeric value of the control, between the maximum and minimum values |

### Related Functions

[ADD.LIST.ITEM](#)

Adds an item in a list box or drop-down control on a worksheet or dialog sheet control

[REMOVE.LIST.ITEM](#)

Removes an item in a list box or drop-down box

[SELECT.LIST.ITEM](#)

Selects an item in a list box or in a group box

[CHECKBOX.PROPERTIES](#)

Sets various properties of check box and option box controls

[SCROLLBAR.PROPERTIES](#)

Sets the properties of the scroll bar and spinner controls

List of [Customizing Functions](#)

## SET.DIALOG.DEFAULT

Macro Sheets Only

Sets which button is automatically pressed (the default button) when the user presses ENTER. While running, this default button is visually recognized by its thick border. This function is used only with a dialog sheet active.

### Syntax

**SET.DIALOG.DEFAULT(object\_id\_text)**

Object\_id\_text is the name of the button control to set as the default button, as in "Button 5".

### Related Functions

SET.DIALOG.FOCUS

Sets the focus of a dialog box

List of Customizing Functions



## SET.DIALOG.FOCUS

Macro Sheets Only

Sets the focus of a dialog box. This function is used only with a dialog sheet active.

### Syntax

**SET.DIALOG.FOCUS(object\_id\_text)**

Object\_id\_text     the name of the control or object as text to give the focus to, as in "Check box 4".

### Related Functions

SET.DIALOG.DEFAULT

Sets which button is automatically pressed (the default button) when the user presses ENTER

List of Customizing Functions

## SET.LIST.ITEM

Macro Sheets Only

Sets the text of an item in a list box or drop-down box control.

### Syntax

**SET.LIST.ITEM(text, index\_num)**

Text specifies the text of the item to be added. Instead of text, an empty string may be inserted.

Index\_num is the list index of the item to be changed, from 1 to the number of items in the list.

### Remarks

If the list box or drop-down box was already filled using the LISTBOX.PROPERTIES function, then changing an item with SET.LIST.ITEM causes the fillrange contents to be discarded, leaving a list with one non-blank element and index\_num entries.

### Related Functions

REMOVE.LIST.ITEM

Removes an item in a list box or drop-down box

SELECT.LIST.ITEM

Selects an item in a list box or in a group box

List of Customizing Functions

## SHOW.DIALOG

Macro Sheets Only

Runs a dialog on a dialog sheet.

### Syntax

**SHOW.DIALOG**(dialog\_sheet)

Dialog\_sheet is the name of the dialog sheet to run. If omitted, the active sheet will be the sheet that is run. If this function is run on a sheet other than a dialog sheet, this function returns the #VALUE error value.

### Remarks

Returns TRUE if the dialog box is closed by the user choosing an OK button. Returns FALSE if the dialog box is cancelled by choosing the Cancel button or the ESC key, or in Microsoft Excel for the Macintosh by pressing COMMAND+. (period).

### Related Functions

HIDE.DIALOG

Closes the dialog box that has the current focus

List of Customizing Functions

## **TAB.ORDER**

Macro Sheets Only

Equivalent to choosing the Tab Order command from the Tools menu. This function determines the order in which dialog controls will be selected when the user presses the TAB key.

### **Syntax**

**TAB.ORDER?()**

### **Remarks**

- This function brings up the Tab Order dialog box and allows the user to select the order in which buttons will be selected when the TAB key is pressed.
- The BRING.TO.FRONT and SEND.TO.BACK macro functions can also be used to programmatically set up the tab order.

### **Related Functions**

BRING.TO.FRONT

Puts the selected object or objects on top of all other objects

SEND.TO.BACK

Puts the selected object or objects behind all other objects

List of Customizing Functions

## APP.ACTIVATE.MICROSOFT

Macro Sheets Only

Activates a Microsoft application. If the application is not already activated, this function will load the application into memory. Equivalent to choosing one of the toolbar buttons on the Microsoft Toolbar.

### Syntax

**APP.ACTIVATE.MICROSOFT(app\_id)**

App\_id is the id number associated with the Microsoft Application.

| App_id | Application                                                           |
|--------|-----------------------------------------------------------------------|
| 1      | Microsoft Word                                                        |
| 2      | Microsoft Powerpoint                                                  |
| 3      | Microsoft Mail                                                        |
| 4      | Microsoft Access (for Microsoft Windows only)                         |
| 5      | Microsoft Fox Pro (for Microsoft Windows) or Fox Base (for Macintosh) |
| 6      | Microsoft Project                                                     |
| 7      | Microsoft Schedule +                                                  |

### Remarks

Returns TRUE if the application is activated successfully. Returns FALSE if the application is not activated successfully.

### Related Functions

APP.ACTIVATE Switches to an application.

List of DDE/External Functions

## FONT.PROPERTIES

Equivalent to choosing the Cells command from the Format menu. Applies a font and other attributes to the selection. Applies to cells, charts, and text boxes and buttons on worksheets and macro sheets.

### Syntax

**FONT.PROPERTIES**(font, font\_style, size, strikethrough, superscript, subscript, outline, shadow, underline, color, normal, background, start\_char, char\_count)

**FONT.PROPERTIES?**(font, font\_style, size, strikethrough, superscript, subscript, outline, shadow, underline, color, normal, background, start\_char, char\_count)

Arguments correspond to check boxes or options in the Font tab on the Format Cells dialog box. Arguments that correspond to check boxes are logical values. If an argument is TRUE, Microsoft Excel selects the check box; if FALSE, Microsoft Excel clears the check box. If an argument is omitted, the format is not changed.

Font is the name of the font as it appears in the Font tab. For example, Courier is a font name.

Font\_style is the name of the font style as it appears in the Font tab. For example, Bold Italic is a font style.

Size is the font size, in points.

Strikethrough corresponds to the Strikethrough check box.

Superscript corresponds to the Superscript check box

Subscript corresponds to the Subscript check box

Outline corresponds to the Outline check box. Outline fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, this argument is ignored by Microsoft Excel for Windows..

Shadow corresponds to the Shadow check box. Shadow fonts are available in Microsoft Excel for the Macintosh. For macro compatibility, this argument is ignored by Microsoft Excel for Windows.

---

**Note** For macro compatibility with Microsoft Excel for the Macintosh, the presence of the outline and shadow arguments do not prevent the macro from working on Microsoft Excel for Windows, nor does their absence prevent it from working on the Macintosh.

---

Underline corresponds to the Underline Drop-down box.

| Underline | Type applied      |
|-----------|-------------------|
| 0         | None              |
| 1         | Single            |
| 2         | Double            |
| 3         | Single Accounting |
| 4         | Double Accounting |

Color is a number from 0 to 56 corresponding to the colors in the Font tab; 0 corresponds to automatic color.

Normal corresponds to the Normal Font check box. Applies the default font for your system

Background is a number from 1 to 3 specifying which type of background to apply to text in a chart.

| Backgd | Type of background applied |
|--------|----------------------------|
| 1      | Automatic                  |
| 2      | Transparent                |
| 3      | Opaque                     |

Start\_char specifies the first character to be formatted. If start\_char is omitted, it is assumed to be 1 (the first character in the cell or text box).

Char\_count specifies how many characters to format. If char\_count is omitted, Microsoft Excel formats all characters in the cell or text box starting at start\_char.

### Remarks

Some extended TrueType styles do not have corresponding arguments to FONT.PROPERTIES. To access an extended TrueType font style, append the style name to the font name in the font argument. For example, the font Taipei can be formatted in an upside-down style by specifying "Taipei Upside-

down" as the font argument. For more information about TrueType, see your Microsoft Windows documentation.

**Related Functions**

ALIGNMENT      Aligns or wraps text in cells

FORMAT.NUMBER   Applies a number format to the selection

FORMAT.TEXT      Formats a worksheet text box or a chart text item

List of Command-Equivalent Functions

## ACTIVE.CELL.FONT

Macro Sheets Only

Equivalent to formatting individual characters in a cell.

### Syntax

**ACTIVE.CELL.FONT**(font, font\_style, size, strikethrough, superscript, subscript, outline, shadow, underline, color, normal, background, start\_char, char\_count)

The arguments for this function are the same as those for FONT.PROPERTIES.

### Related Functions

FONT.PROPERTIES Applies a font and other attributes to the selection

List of Command-Equivalent Functions



## ADDIN.MANAGER

Macro Sheets Only

Equivalent to choosing the Add-Ins command from the Tools menu. Adds or removes an add-in from the working set.

### Syntax

**ADDIN.MANAGER**(operation\_num, addinname\_text, copy\_logical)

**ADDIN.MANAGER?**(operation\_num, addinname\_text, copy\_logical)

Operation\_num determines the operation that the add-in manager will perform.

| Operation_num | Operation |
|---------------|-----------|
|---------------|-----------|

|   |                                                                                                                                                                                                 |
|---|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1 | Install an add-in, using the descriptive name in the add-in manager dialog.                                                                                                                     |
| 2 | Uninstall an add-in, using the descriptive name in the add-in manager dialog.                                                                                                                   |
| 3 | Adds a new add-in to the list of add-ins that Excel knows about. Equivalent to clicking on the Browse button in the add-in manager and choosing a file. The add-in is not installed by default. |

Addinname\_text specifies the name of the add-in. If operation\_num is 1 or 2, use the descriptive name of the add-in, such as "SOLVER". If operation\_num is 3, this contains the filename of the add-in.

Copy\_logical specifies whether the add-in should be copied to the library directory. This argument is only used if operation\_num is 3. If omitted, and the file is on removable media, the user will be asked if they want to copy it to removable media.

### Related Functions

List of [Command-Equivalent Functions](#)

## **ATTACH.TOOLBARS**

Macro Sheets Only

Displays the Attach Toolbars dialog box, which you use to attach or associate toolbars with documents.

### **Syntax**

**ATTACH.TOOLBARS?( )**

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **AUTO.OUTLINE**

Macro Sheets Only

Equivalent to choosing the Auto Outline command from the Group and Outline submenu on the Data menu. Creates an outline within the selection. If a single cell is selected, creates an outline for the whole sheet.

### **Syntax**

**AUTO.OUTLINE( )**

### **Related Functions**

CLEAR.OUTLINE Removes outlining from the current sheet

OUTLINE Creates an outline and defines settings for automatically creating outlines

List of Command-Equivalent Functions

## CHART.ADD.DATA

Equivalent to dragging data from a worksheet onto a chart. Adds data to an existing chart.

### Syntax

**CHART.ADD.DATA**(ref, rowcol, titles, categories, replace, series)

Ref is the cell reference for the data that is being dragged onto the chart

Rowcol is the number 1 or 2 and specifies whether the values corresponding to a particular data series are in rows or columns. Enter 1 for rows or 2 for columns.

Titles is a logical value corresponding to the Series Names In First Column check box (or First Row, depending on the value of rowcol) in the Paste Special dialog box.

- If series is TRUE, Microsoft Excel selects the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the name of the data series in that row (or column).

- If series is FALSE, Microsoft Excel clears the check box and uses the contents of the cell in the first column of each row (or first row of each column) as the first data point of the data series.

Categories is a logical value corresponding to the Categories (X Labels) In First Row (or First Column, depending on the value of rowcol) check box in the Paste Special dialog box.

- If categories is TRUE, Microsoft Excel selects the check box and uses the contents of the first row (or column) of the selection as the categories for the chart.

- If categories is FALSE, Microsoft Excel clears the check box and uses the contents of the first row (or column) as the first data series in the chart.

Replace is a logical value corresponding to the Replace Existing Categories check box in the Paste Special dialog box.

- If replace is TRUE, Microsoft Excel selects the check box and applies categories while replacing existing categories with information from the copied cell range.

- If replace is FALSE, Microsoft Excel clears the check box and applies new categories without replacing any old ones.

Series is a number specifying how cells are added to a chart.

| Series | Added as     |
|--------|--------------|
| 1      | New series   |
| 2      | New point(s) |

### Related Functions

List of [Command-Equivalent Functions](#)

## **CLEAR.OUTLINE**

Macro Sheets Only

Equivalent to choosing the Clear Outline command from the Group and Outline submenu on the Data menu. Clears the outline within the selection. If a single cell is selected, it clears the outline from the whole sheet.

### **Syntax**

**CLEAR.OUTLINE()**

### **Related Functions**

AUTO.OUTLINE Creates an outline

OUTLINE Creates an outline and defines settings for automatically creating outlines

List of Command-Equivalent Functions

## EDIT.TOOL

Macro Sheets Only

Displays the Button Editor dialog, which you use to change the appearance of a button on a toolbar.

### Syntax

**EDIT.TOOL(bar\_id, position)**

Bar\_id is the number of the toolbar containing the button you want to edit. For a list of toolbar numbers, see ADD.TOOL. Use the GET.TOOLBAR function to return the information about a toolbar.

Position is the position on the toolbar of the button you want to edit. Buttons are numbered from the left starting at 1. Gaps between buttons are counted as positions.

### Related Functions

ADD.TOOL Adds a button to a toolbar

GET.TOOLBAR Returns information about a toolbar

List of Command-Equivalent Functions

## **ENABLE.TIPWIZARD**

Macro Sheets Only

Turns on or off the Tip Wizard.

### **Syntax**

**ENABLE.TIPWIZARD**(state)

State is a logical value specifying whether the Tip Wizard is on: use TRUE to turn it on, FALSE to turn it off.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## ERRORBAR.X ERRORBAR.Y

Macro Sheets Only

Adds error bars to the selected series in a chart. ERRORBAR.X adds bars showing the error factor for the X (category) axis and works for XY (scatter) charts only. ERRORBAR.Y adds bars showing the error factor for the Y (value) axis for all charts.

### Syntax

**ERRORBAR.X**(include, type, amount, minus)

**ERRORBAR.Y**(include, type, amount, minus)

Include specifies the type of error value to include:

| Include | Type of error value |
|---------|---------------------|
|---------|---------------------|

|              |                |
|--------------|----------------|
| 1 or omitted | Plus and minus |
|--------------|----------------|

|   |      |
|---|------|
| 2 | Plus |
|---|------|

|   |       |
|---|-------|
| 3 | Minus |
|---|-------|

|   |      |
|---|------|
| 4 | None |
|---|------|

Type specifies the type of error bars to display:

| Type | Type of error displayed |
|------|-------------------------|
|------|-------------------------|

|              |              |
|--------------|--------------|
| 1 or omitted | Fixed amount |
|--------------|--------------|

|   |         |
|---|---------|
| 2 | Percent |
|---|---------|

|   |                                                            |
|---|------------------------------------------------------------|
| 3 | Multiplying factor standard deviation (default value is 1) |
|---|------------------------------------------------------------|

|   |                |
|---|----------------|
| 4 | Standard error |
|---|----------------|

|   |        |
|---|--------|
| 5 | Custom |
|---|--------|

Amount is the range of error values to display. This argument depends on the value of type:

| If type is | then amount |
|------------|-------------|
|------------|-------------|

|              |                                  |
|--------------|----------------------------------|
| 1 or omitted | can be any number greater than 0 |
|--------------|----------------------------------|

|   |                                  |
|---|----------------------------------|
| 2 | can be any number greater than 0 |
|---|----------------------------------|

|   |                                     |
|---|-------------------------------------|
| 3 | can be any number greater than or 0 |
|---|-------------------------------------|

|   |              |
|---|--------------|
| 4 | not required |
|---|--------------|

|   |                                              |
|---|----------------------------------------------|
| 5 | is the positive amount for custom error bars |
|---|----------------------------------------------|

Minus is the negative amount for custom error bars. Applicable only if type is 5.

### Remarks

For the amount argument, standard deviation(s) can be calculated using this equation:

$$S.D. = \sqrt{\frac{\sum_{s=1}^m \sum_{i=1}^n y_{is}^2}{(n_y - 1)}}$$

$$M = \frac{\sum_{s=1}^m \sum_{i=1}^n y_{is}}{n_y}$$

The standard deviation is multiplied by the value specified by amount and the error bars are placed this distance from the arithmetic mean. Therefore, these error bars are plotted along the arithmetic mean, not attached to data series.

Microsoft Excel calculates the standard error using the following equation:



$$S.E. = \sqrt{\frac{\sum_{s=1}^m \sum_{i=1}^n y_{is}^2}{(n_y - 1)(n_y)}}$$

Both the standard deviation and standard error functions use the following variables:

| Variable       | Equals                                    |
|----------------|-------------------------------------------|
| s              | series number                             |
| i              | point number in series s                  |
| m              | number of series for point y in chart     |
| n              | number of points in each series           |
| Y <sub>i</sub> | data value of series s and the ith point  |
| N <sub>y</sub> | total number of data values in all series |
| M              | arithmetic mean                           |

### Related Functions

List of [Command-Equivalent Functions](#)

## FORMAT.CHARTTYPE

Macro Sheets Only

Changes the chart type for a selected data series, a group of chart types, or an entire chart.

### Syntax

**FORMAT.CHARTTYPE**(apply\_to, group\_num, dimension, **type\_num**)

**FORMAT.CHARTTYPE?**(apply\_to, group\_num, dimension, type\_num)

Apply\_to is a number from 1 to 3 specifying what part of a chart the new chart type effects.

| Value | Part of chart |
|-------|---------------|
|-------|---------------|

|   |                      |
|---|----------------------|
| 1 | Selected data series |
|---|----------------------|

|   |                      |
|---|----------------------|
| 2 | Group of data series |
|---|----------------------|

|   |              |
|---|--------------|
| 3 | Entire chart |
|---|--------------|

Group\_num corresponds to the number of the group you want to change as listed in the Group list box of the Chart Type dialog box, which appears when you choose Chart Type from the Format menu while a chart is active. Groups are numbered starting with 1 for the group at the top of the list. This argument is required if apply\_to equals 2; otherwise it is ignored.

Dimension specifies whether to apply a 2D- or 3D-chart type. Use 1 for a 2D chart type or 2 for a 3D chart type. If omitted, uses the same dimension as the series, group, or chart to be changed.

Type\_num specifies the chart type to apply. Meaning of type\_num varies depending on the value of dimension:

| Type_num | Chart type if dimension is 1 | Chart type if dimension is 2 |
|----------|------------------------------|------------------------------|
|----------|------------------------------|------------------------------|

|   |      |         |
|---|------|---------|
| 1 | Area | 3D area |
|---|------|---------|

|   |     |        |
|---|-----|--------|
| 2 | Bar | 3D bar |
|---|-----|--------|

|   |        |           |
|---|--------|-----------|
| 3 | Column | 3D column |
|---|--------|-----------|

|   |      |         |
|---|------|---------|
| 4 | Line | 3D line |
|---|------|---------|

|   |     |        |
|---|-----|--------|
| 5 | Pie | 3D pie |
|---|-----|--------|

|   |          |            |
|---|----------|------------|
| 6 | Doughnut | 3D surface |
|---|----------|------------|

|   |       |  |
|---|-------|--|
| 7 | Radar |  |
|---|-------|--|

|   |              |  |
|---|--------------|--|
| 8 | XY (scatter) |  |
|---|--------------|--|

### Related Functions

FORMAT.CHART Formats the selected chart

List of Command-Equivalent Functions

## FUNCTION.WIZARD

Macro Sheets Only

Displays the Function Wizard dialog box, which you can use to enter functions into cells.

### Syntax

**FUNCTION.WIZARD?**( )

### Remarks

If you know the function or formula that you want to insert into a cell, use the FORMULA function.

### Related Functions

FORMULA          Enters values into a cell or range or onto a chart

List of Command-Equivalent Functions

## INSERT.TITLE

Macro Sheets Only

Equivalent to the Titles command on the Insert menu when a chart is active. Attaches text to various parts of a chart.

### Syntax

2D charts

**INSERT.TITLE**(chart, y\_primary, x\_primary, y\_secondary, x\_secondary)

3D charts

**INSERT.TITLE**(chart, z\_primary, x\_primary, y\_primary)

Chart is a logical value specifying whether to attach a title to the chart.

Y\_primary is a logical value specifying whether to attach a title to the value (y) axis of a 2D chart or the series (y) axis of a 3D chart.

X\_primary is a logical value specifying whether to attach a title to the category (x) axis of the chart.

Z\_primary is a logical value specifying whether to attach a title to the value (z) axis of a 3D chart.

Y\_secondary is a logical value specifying whether to attach a title to the second value (y) axis of a chart containing more than one chart type.

X\_secondary is a logical value specifying whether to attach a title to the second category (x) axis of a chart containing more than one chart type.

### Remarks

To change the text in a selected title, use the FORMULA function.

### Related Functions

FORMULA Enters formulas in a chart

List of Command-Equivalent Functions

## JUSTIFY

Macro Sheets Only

Equivalent to choosing the Justify command on the Fill submenu on the Edit menu. Rearranges the text in a range so that it fills the range evenly.

### Syntax

**JUSTIFY()**

### Related Functions

ALIGNMENT      Aligns the contents of the selected cells

List of Command-Equivalent Functions

## **LINK.FORMAT**

Macro Sheets Only

Equivalent to the Linked to Source checkbox in the Number tab of the Format Data Label dialog box. Links the number format of the selected data label to the worksheet cell or range containing the data label text.

### **Syntax**

**LINK.FORMAT( )**

### **Related Functions**

List of Command-Equivalent Functions

## MAIL.LOGOFF

Macro Sheets Only

Ends the current mail session.

---

**Important** To use MAIL.LOGOFF in Microsoft Excel for Windows, you must be using a mail client that supports the Messaging Applications Programming Interface (MAPI) or Vendor-Independent Messaging (VIM). The function is available for only Microsoft Excel for Windows.

---

### Syntax

**MAIL.LOGOFF( )**

### Remarks

Returns TRUE if the session was ended, or #VALUE! if there was no session.

### Related Functions

List of [Command-Equivalent Functions](#)

## MAIL.LOGON

Macro Sheets Only

Starts a mail session.

---

**Important** To use MAIL.LOGON in Microsoft Excel for Windows, you must be using a mail client that supports the Messaging Applications Programming Interface (MAPI) or Vendor-Independent Messaging (VIM). The function is available for only Microsoft Excel for Windows.

---

### Syntax

**MAIL.LOGON**(name\_text, password\_text, download\_logical)

**MAIL.LOGON?**(name\_text, password\_text, download\_logical)

Name\_text is the username of the mail account. If omitted, prompts for username.

Password\_text is the password for the account. If omitted, prompts for password. Ignored when the dialog box form is used.

Download\_logical specifies whether to download new mail. Use TRUE to download new mail; use FALSE or leave blank to skip downloading new mail.

### Remarks

Returns FALSE if you cancel the dialog box or #VALUE! if you can't log on.

If you omit both name\_text and password\_text, Microsoft Excel tries to log on using an existing mail session.

### Related Functions

MAIL.LOGOFF Ends the current mail session

List of Command-Equivalent Functions



## **MENU.EDITOR**

Macro Sheets Only

Displays the Menu Editor dialog box, which you use to create custom commands and menus to use with your Visual Basic procedures.

### **Syntax**

**MENU.EDITOR?()**

### **Related Functions**

List of [Command-Equivalent Functions](#)

## OPEN.TEXT

Macro Sheets Only

Equivalent to using the Text Import Wizard to open a text file in Microsoft Excel.

### Syntax

**OPEN.TEXT**(file\_name, file\_origin, start\_row, file\_type, text\_qualifier, consecutive\_delim, tab, semicolon, comma, space, other, other\_char, {field\_info1; field\_info2;...})

File\_name is the full pathname of the text file you want to open.

File\_origin specifies the operating environment the text file was created in.

| File_origin | Operating system |
|-------------|------------------|
|-------------|------------------|

|         |                               |
|---------|-------------------------------|
| 1       | Macintosh                     |
| 2       | Windows (ANSI)                |
| 3       | MS DOS (PC-8)                 |
| omitted | Current operating environment |

Start\_row is a number greater than or equal to one, specifying the row in the text file where you want to start importing into Microsoft Excel. This number should be less than the number of lines in the text file.

File\_type specifies the type of delimited text file to import:

| File_type | Type of file |
|-----------|--------------|
|-----------|--------------|

|              |             |
|--------------|-------------|
| 1 or omitted | Delimited   |
| 2            | Fixed Width |

Text\_qualifier indicates the character-enclosing text fields in the text file:

| Text_qualifier value | Qualifier |
|----------------------|-----------|
|----------------------|-----------|

|             |                           |
|-------------|---------------------------|
| 1 or "      | " (double quotation mark) |
| 2 or '      | ' (single quotation mark) |
| 3 or {None} | No text qualifier         |

Consecutive\_delim is a logical value corresponding to the Treat Consecutive Delimiters as One check box which, if TRUE, allows consecutive delimiters (such as ",,,"") to be treated as a single delimiter. If FALSE, all consecutive delimiters are considered separate column breaks.

Tab, semicolon, comma, space are logical values corresponding to the check boxes in the Column Delimiters group. If an argument is TRUE, the check box is selected. These arguments apply only when the file\_type argument is 1 or omitted (delimited file type).

Other is a logical value indicating a custom delimiter to use in opening the text file.

Other\_char specifies the custom delimiter to use or FALSE if no custom delimiter is used.

Field\_info is an array which consists of the following elements: "column number, data\_format", if file\_type is 1; or "start\_pos, data\_format" if file\_type is 2.

### Related Functions

TEXT.TO.COLUMNS Parses text, as in a text file, into columns of data

List of Command-Equivalent Functions

## OVERLAY

Macro Sheets Only

Equivalent to choosing the Overlay command from the Format menu in Microsoft Excel version 2.2 or earlier. This function is included only for macro compatibility. To format chart types in Microsoft Excel version 5.0 or later, use the `FORMAT.CHART` function.

### Syntax

**OVERLAY**(**type\_num**, stack, 100, vary, overlap, drop, hilo, overlap%, cluster, angle, series\_num, auto)

### Related Functions

FORMAT.CHART Formats the chart according to the arguments you specify.

List of Command-Equivalent Functions

## REPLACE.FONT

Macro Sheets Only

Replaces one of the four built-in fonts in Microsoft Excel for Windows version 2.1 or earlier with a new font and style. This function is included only for macro compatibility. To change the font of the selected cell or range as part of a macro, use the FONT.PROPERTIES function.

### Syntax

**REPLACE.FONT**(font\_num, name\_text, size\_num, bold, italic, underline, strike, color, outline, shadow)

### Related Functions

FONT.PROPERTIES      Sets various font attributes

List of Command-Equivalent Functions

## **SAVE.COPY.AS**

Macro Sheets Only

Saves a copy of the current document using a different name but all the current document settings, such as passwords and file protection. Does not affect the current document. Use this command if you need a temporary copy of the current document; for example, to include in an electronic mail message.

### **Syntax**

**SAVE.COPY.AS**(document\_text)

Document\_text is the name you want to give the copy of the workbook.

### **Example**

Suppose that you are creating a macro that makes changes to a file called BUDGET95.XLS. Use the following function to save a copy of this file called TEMP.XLS without affecting BUDGET95.XLS:

```
SAVE.COPY.AS ("temp.xls")
```

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **SELECT.ALL**

Macro Sheets Only

Equivalent to selecting all the sheets in a workbook.

### **Syntax**

**SELECT.ALL( )**

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **SERIES.AXES**

Macro Sheets Only

Equivalent to the Axis Tab in the Format Data Series dialog box. Changes the axis on which a series is plotted.

### **Syntax**

**SERIES.AXES**(axis)

Axis is a number specifying on which axis to plot the data series: use 1 for primary axis, 2 for secondary axis.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **SERIES.X**

Macro Sheets Only

Equivalent to the X Values tab in the Format Data Series dialog box. Specifies the category labels (x values) for a data series.

### **Syntax**

**SERIES.X**(x\_ref)

X-ref is an external reference in the form of text specifying the range containing the category labels (or x values for a scatter (xy) chart) you want to use.

### **Related Functions**

SERIES.Y Specifies the name and values for a data series

List of Command-Equivalent Functions



## **SERIES.Y**

Macro Sheets Only

Equivalent to the Name and Values tab in the Format Data Series dialog box. Specifies the name and values for a data series.

### **Syntax**

**SERIES.Y**(name\_ref, y\_ref)

Name\_ref is text or an external reference in the form of text specifying the name for the data series that appears in the legend for the chart.

Y\_ref is an external reference in the form of text specifying the range containing the values for the data series.

### **Related Functions**

[SERIES.X](#) Specifies the category labels (x values) for a data series

List of [Command-Equivalent Functions](#)

## **SERIES.ORDER**

Macro Sheets Only

Changes the order of series in a chart.

### **Syntax**

**SERIES.ORDER**(chart\_num, old\_series\_num, new\_series\_num)

Chart\_num is the number of the group containing the series you want to change

Old\_series\_num is the current number of the series in the group.

New\_series\_num is the new number you want for the series in the group.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## STANDARD.FONT

Macro Sheets Only

Sets the attributes of the standard font in Microsoft Excel version 2.2 and earlier. This function is included only for macro compatibility. To define and apply a style in Microsoft Excel 5.0, use the [APPLY.STYLE](#) and [DEFINE.STYLE](#) functions.

### Syntax

**STANDARD.FONT**(name\_text, size\_num, bold, italic, underline, strike, color, outline, shadow)

The arguments for this function are the same as those for [FORMAT.FONT](#).

### Related Functions

[APPLY.STYLE](#) Applies a style to the selection

[DEFINE.STYLE](#) Defines a cell style

[FORMAT.FONT](#) Applies a font to the selection

List of [Command-Equivalent Functions](#)

## **STANDARD.WIDTH**

Macro Sheets Only

Sets the default width used for all columns in a worksheet.

### **Syntax**

**STANDARD.WIDTH**(standard\_num)

Standard\_num specifies how wide you want the columns to be in units of one character of the font corresponding to the Normal cell style.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **VBA.MAKE.ADDIN**

Macro Sheets Only

Converts a workbook containing Visual Basic procedures into an add-in.

### **Syntax**

**VBA.MAKE.ADDIN(filename\_text)**

Filename\_text is the name of the workbook that you want to convert to an add-in.

### **Remarks**

For information about creating add-ins with Visual Basic, see Chapter 13, "Creating Automatic Procedures and Add-in Applications" in the Visual Basic User's Guide.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## **VBA.INSERT.FILE**

Macro Sheets Only

Inserts a text file containing code directly into your Visual Basic module.

### **Syntax**

**VBA.INSERT.FILE**(filename\_text)

Filename\_text is the name of a text file that contains Microsoft Visual Basic code that is inserted into the currently active module.

### **Related Functions**

List of [Command-Equivalent Functions](#)

## Changing links to Microsoft Excel version 4.0 add-ins

Worksheet and macro sheet functions from the following Microsoft Excel 4.0 add-in macros are now built into Microsoft Excel version 5.0:

| <b>Add-in macro name</b> | <b>Windows filename</b> | <b>Macintosh filename</b> |
|--------------------------|-------------------------|---------------------------|
| Add-in Functions         | ADDINFNS.XLS            | ADD-IN FUNCTIONS          |
| Analysis ToolPak         | ANALYSIS.XLA            | ANALYSIS TOOLS            |
|                          | ANALYSF.XLA             | ANALYSIS FUNCTIONS        |
| Scenario Manager         | SCENARIO.XLA            | SCENARIO MANAGER          |

These add-in macros have been updated so that you can use Microsoft Excel version 4.0 documents containing these worksheet and macro sheet functions in Microsoft Excel version 5.0 without modification. However, you can increase the calculation and execution speed of such documents by changing references to the add-in functions to refer directly to the built-in versions of the same functions. Microsoft Excel version 5.0 includes a utility to change these references.

### To change references to add-in functions

- 1 Open the documents containing the references you want to change.
- 2 In Microsoft Excel for Windows, open the file UPDTLINK.XLA stored in the LIBRARY subdirectory of the directory containing Microsoft Excel.  
With Microsoft Excel for the Macintosh, open the file UPDATE LINKS stored in the MACRO LIBRARY folder of the Microsoft Excel folder.
- 3 From the Tools menu, choose MS Excel 4.0 Add-In Links.
- 4 To update the references in just the active workbook, select Active Document.  
To update the references in all open workbooks, select All Open Documents.
- 5 Choose the OK button.

## Visual Basic Equivalents for Macro Functions and Commands



To see the Visual Basic statement that is equivalent to a Microsoft Excel version 4.0 macro function, click above on the first letter of the macro function. Then scroll through the list to find the macro function you're interested in.

[More about converting Microsoft Excel version 4.0 macros to Visual Basic procedures](#)



## More about converting Microsoft Excel version 4.0 macros to Visual Basic procedures

---

**Tip** With Microsoft Excel version 5.0, you can record macros in both Visual Basic and the Microsoft Excel 4.0 Macro Language. Comparing the same actions recorded in both languages can be very instructive in learning Visual Basic.

---

In many cases, some translation must occur between arguments to Microsoft Excel 4.0 Macro Language functions and Visual Basic properties or methods. Common cases where translation must occur include:

- **References** The Microsoft Excel 4.0 Macro Language reference datatype is a direct representation of a cell address or range of cells on a worksheet or macrosheet, such as \$C\$10. A reference can be entered directly in an Microsoft Excel 4.0 Macro Language expression since the macro sheet supports references. Visual Basic has an equivalent datatype, the Range object, but the means of creating and manipulating ranges differ from references. Visual Basic ranges cannot be entered directly; they must be created by using a set of numeric row and column cell indices, or by expressing the range address as A1-style text. For example, Application.Range(Cells(1, 1), Cells(10, 10)) or Application.Range("A1:J10") returns the range A1 through J10.
- **Constants as arguments** Many macro functions take numbers as arguments. For example, INSERT(2) means shift cells down when inserting new cells. In Visual Basic, methods and properties use text constants instead of numbers for arguments. For example, in the Visual Basic statement Range("A1:C5").Insert(xlDown), "xlDown" means shift cells down when inserting new cells.
- **Making selections in macros** Most Microsoft Excel 4.0-style macros implicitly operate on the selection or active sheet. Visual Basic methods and properties, on the other hand, rarely operate on the selection. Instead, you must explicitly specify the object you want to work on. This gives you the flexibility of using objects that are not active or selected. In some cases, Visual Basic treats objects in a fundamentally different manner than the Microsoft Excel 4.0 Macro language. In these cases there may be no direct translation of an Microsoft Excel 4.0 Macro Language expression.
- **Functions in Add-ins** Microsoft Excel 4.0 macro functions contained in add-ins remain the same in Visual Basic. You access these functions by making a reference to the add-in using the References command on the Tools menu when a Visual Basic Module is active. Once you have done this, you can include the Microsoft Excel 4.0 macro functions in your procedure. For example, here's how to use the ANOVA macro function after you have used the References command:

```
Sub Analyze_Data
 Anova1Q(A1:D15, F1, "C")
End Sub
```

Use the same syntax as you would in a Microsoft Excel 4.0-style macro, with one exception. To use the dialog-box form of a macro function, add a Q to the end of a macro function's name instead of a question mark. For instance, ANOVA1Q instead of ANOVA1?.

You can also use macro functions contained in add-ins by using the Visual Basic ExecuteExcel4Macro method. This method takes as arguments the name and arguments of a macro function. For example, to use the ANOVA macro function to perform analysis of variance on data in the range A1:D15, include the following statement in your procedure:

```
Application.ExecuteExcel4Macro("ANOVA1?(A1:D15, F1, ""C"")")
```

For more information on upgrading to Visual Basic, see Appendix B, "Switching from the Microsoft Excel 4.0 Macro Language," in the Microsoft Excel Visual Basic User's Guide.

## Visual Basic Equivalents for Macro Functions and Commands

.....



A

**A1.R1C1**(logical)

Application.ReferenceStyle = *logical*

**ABSREF**(ref\_text, reference)

Range("*reference*").Offset(*ref\_text*)

No direct equivalent. Use the Offset method and split the ref\_text argument into numeric arguments.

**ACTIVATE**("BOOK1.XLS")

Windows("BOOK1.XLS").Activate  
' or  
Workbooks("BOOK1.XLS").Activate

To activate the first window of a particular workbook, use the Workbooks form .

**ACTIVATE**("BOOK1.XLS:1")

Windows("BOOK1.XLS:1").Activate  
' or  
Workbooks("BOOK1.XLS").Windows(1).Activate

To activate a specific window, use the number of the window with the Workbooks form.

**ACTIVATE**("BOOK1.XLS:1", pane\_num)

Windows("BOOK1.XLS:1").Panes(pane\_num).Activate

**ACTIVATE**(, pane\_num)

ActiveWindow.Panes(pane\_num).Activate

**ACTIVATE.NEXT**( )

ActiveWindow.ActivateNext

**ACTIVATE.NEXT**(workbook\_text)

ActiveSheet.Next.Activate

**ACTIVATE.PREV**( )

ActiveWindow.ActivatePrevious

**ACTIVATE.PREV**(workbook\_text)

ActiveSheet.Previous.Activate

## **ACTIVE.CELL**( )

ActiveCell

**ACTIVE.CELL.FONT**(font, font style, size, strikethrough, superscript, subscript, outline, shadow, underline, color, normal, background, start\_char, char\_count)

```
With ActiveCell.Characters(start_char, char_count).Font
 .Name = char_count
 .FontStyle = font_style
 .Size = size
 .Strikethrough = strikethrough
 .Superscript = superscript
 .Subscript = subscript
 .OutlineFont = outline
 .Shadow = shadow
 .Underline = underline
 .ColorIndex = color
 .Background = background
```

End With

The normal parameter has no direct equivalent. You can set the font of the Normal style by copying the settings of the Normal style Font object, by using `ActiveWorkbook.Styles("Normal").Font`.

## **ADD.ARROW**( )

```
With ActiveChart.Lines.Add(0, 0, 72, 72)
 .ArrowHeadStyle = xlClosed
```

End With

No direct equivalent. Add a line with a default set of coordinates using the Add method of the Lines object, then add the arrowhead to the new line.

## **ADD.BAR**( )

Menubars.Add

## **ADD.BAR**(bar\_num)

Menubars(*bar\_num*).Reset

**ADD.CHART.AUTOFORMAT**(name\_text, desc\_text)

```
Application.AddChartAutoFormat chart := ActiveChart, _
 name := name_text, description := desc_text
```

**ADD.COMMAND**(bar\_num, menu, command\_ref, position1)

```
Menubars(bar_num).Menus(menu).MenuItems.Add _
 caption := caption, onAction := macro, _
 shortcutKey := key, before := position1
```

Adds a command to a menu. The `command_ref` reference area contains the information for the caption, onAction, and shortcutKey arguments. The status bar text and help topic assigned to the command are determined by the status bar text and help topic assigned to the OnAction macro. The macro option are set using the Options button in the Macro command on the Tools menu.

**ADD.COMMAND**(bar\_num, menu, command\_ref, position1, position2)

```
Menubars(bar_num).Menus(menu). _
 MenuItems(position1).MenuItems.Add _
 caption := caption, onAction := macro, _
 shortcutKey := key, before := position2
```

Adds a command to a submenu.

**ADD.COMMAND**(bar\_num, menu, command\_ref, "Menu To Restore")

```
Menubars(bar_num).Menus(Menu).MenuItems.Add _
 caption := "Menu To Restore", restore := True
```

Restores a deleted menu command.

**ADD.COMMAND**(bar\_num, menu, command\_ref, position1, position2)

```
ShortcutMenus(shortcut_menu).MenuItems.Add _
 caption := caption, onAction := macro, _
 shortcutKey := key, before := position1
```

Adds a command to a shortcut menu. Shortcut menus are accessed in Visual Basic by using the

ShortcutMenus method with the appropriate constant.

|                                                          |                                                                                           |
|----------------------------------------------------------|-------------------------------------------------------------------------------------------|
| <u><b>ADD.MENU</b></u> (bar_num, menu_ref,<br>position1) | Menubars( <i>bar_num</i> ).Menus.Add caption := <i>caption</i> , _<br>before := position1 |
|----------------------------------------------------------|-------------------------------------------------------------------------------------------|

Adds a menu. The menu\_ref reference area contains information on the menu and its contents. In Visual Basic, the contents of the menu must be explicitly added using the Add and AddMenu methods of the MenuItems object.

```

ADD.MENU(bar_num, menu_ref,
position1, position2)
Menubars(bar_num).Menus(position1).MenuItems.
AddMenu caption := caption, before := position2

```

Adds a submenu to an existing menu.

```
ADD.MENU(bar_num, "Menu To Restore") Menubars(bar_num).Menus.Add caption := caption, _
 restore := True
```

Restores a deleted menu.

## ADD.OVERLAY()

No direct equivalent. A secondary chart group (overlay) is created automatically when a series is added with a different chart type. See the `Type` property for the `Series` and `ChartGroup` objects.

|                                                     |                                                                                                                                                                            |
|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>ADD.TOOL</u></b> (bar_id, position, tool_ref) | Toolbars(bar_id).ToolBarButtons.Add button := <i>button</i> , _<br>before := position, onAction := <i>macro</i> , _<br>pushed := <i>pushed</i> , enabled := <i>enabled</i> |
|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Adds a custom toolbar button. The tool\_ref reference area contains the information for the button, onAction, pushed, and enabled parameters. The status bar text and help topic assigned to the button are determined by the status bar text and help topic assigned to the OnAction macro. The macro options are set using the Options button in the Macro command on the Tools menu. The Attach Toolbars command on the Tools menu lets you associate static toolbars with a document.

|                                                   |                                                                               |
|---------------------------------------------------|-------------------------------------------------------------------------------|
| <b><u>ADD.TOOL</u></b> (bar_id, position, button) | Toolbars(bar_id).ToolBarButtons.Add button := button, _<br>before := position |
|---------------------------------------------------|-------------------------------------------------------------------------------|

Adds a built-in toolbar button.

|                                                |                                         |
|------------------------------------------------|-----------------------------------------|
| <b><u>ADD.TOOLBAR</u></b> (bar_name, tool_ref) | With Toolbars.Add(bar_name := bar_name) |
|                                                | .ToolBarButtons.Add button := tool_ref  |
|                                                | End With                                |

Adds a toolbar. The `tool_ref` area contains information on the toolbar and its buttons. In Visual Basic, buttons are added to the new toolbar with the `Add` method of the `ToolbarButtons` object. This example adds a new toolbar and then adds a built-in button (`tool_ref` is a number in this syntax, not a reference).

**ADDIN.MANAGER**(1, addinname\_text)      AddIns(addinname\_text).Installed = True

|                                                 |                                          |
|-------------------------------------------------|------------------------------------------|
| <b><u>ADDIN.MANAGER</u></b> (2, addinname_text) | AddIns(addinname_text).Installed = False |
|-------------------------------------------------|------------------------------------------|

|                                                                  |                                                                      |
|------------------------------------------------------------------|----------------------------------------------------------------------|
| <b><u>ADDIN.MANAGER</u></b> (3, addinname_text,<br>copy logical) | AddIns.Add filename := addinname_text, _<br>copyFile := copy logical |
|------------------------------------------------------------------|----------------------------------------------------------------------|

**ALERT**(message\_text, type\_num, help\_ref)      MsgBox prompt := message\_text, buttons := type\_num, \_  
helpfile := help\_ref, context := help\_ref

In Visual Basic, the help filename and the help context ID must be specified as a string and a number in the "HELPPFILE!ID" format of ALERT.

**ALIGNMENT**(horiz\_align, wrap, vert\_align, orientation)

With Selection  
.HorizontalAlignment = *horiz\_align*  
.WrapText = wrap  
.VerticalAlignment = *vert\_align*  
.Orientation = *orientation*  
End With

**APP.ACTIVATE.MICROSOFT**(app\_id)

Application.ActivateMicrosoftApp index := *app\_id*

**APP.ACTIVATE**(title\_text, wait\_logical)

AppActivate title := title\_text, wait := wait\_logical

In Visual Basic, any window whose title begins with title\_text is activated. An exact match is not required.

**APP.MAXIMIZE**( )

Application.WindowState = xlMaximized

**APP.MINIMIZE**( )

Application.WindowState = xlMinimized

**APP.MOVE**(x\_num, y\_num)

With Application  
.Left = x\_num  
.Top = y\_num  
End With

**APP.RESTORE**( )

Application.WindowState = xlNormal

**APP.SIZE**(x\_num, y\_num)

With Application  
.Width = x\_num  
.Height = y\_num  
End With

**APP.TITLE**(text)

Application.Caption = text

**APPLY.NAMES**( )

Range.ApplyNames

**APPLY.STYLE**(style\_text)

Selection.Style = ActiveWorkbook.Styles(style\_text)

**ARGUMENT**( )

No direct equivalent. Arguments to Visual Basic procedures are included as a part of the procedure declaration and header.

**ARRANGE.ALL**(arrange\_num, active\_doc, sync\_horiz, sync\_vert)

Windows.Arrange arrangeStyle := *arrange\_num*, \_  
activeWorkbook := active\_doc, \_  
syncHorizontal := sync\_horiz, \_  
syncVertical := sync\_vert

**ASSIGN.TO.OBJECT**(macro\_ref)

Selection.OnAction := *macro\_ref*

Macro\_ref must specify the macro name as text, not as a reference.

**ASSIGN.TO.TOOL**(bar\_id, position,

Toolbars(bar\_id).ToolbarButtons(position). \_



Creates an outline border around the selected cells. In Visual Basic, use either the `LineStyle` or the `Weight` argument to the `BorderAround` method, depending on the desired border type.

**BORDER**(, left, , , , left\_color, right\_color)  
 With Selection.Borders(xlLeft)  
     .Weight = *left\_color*  
     .ColorIndex = *left\_color*  
 End With

Sets the border for the left side of the selected cells (other sides are not modified). In Visual Basic, use the appropriate constant with the `Borders` method to select the border to modify. Use either the `LineStyle` or the `Weight` property, depending on the desired border type.

**BREAK**( )  
 Exit For  
 ' or  
 Exit Do statement

To break out of a For...Next or For Each...Next loop, use `Exit For` . To break out of a Do...Loop statement, use `Exit Do`.

**BRING.TO.FRONT**( )  
 Selection.BringToFront

## C

**CALCULATE.DOCUMENT**( )  
 ActiveSheet.Calculate

**CALCULATE.NOW**( )  
 ActiveWorkbook.Calculate

**CALCULATION**(type\_num, iter, max\_num, max\_change, update, precision, date\_1904, calc\_save, save\_values, alt\_exp, alt\_form)  
 With Application  
     .Calculation = *type\_num*  
     .Iteration = *iter*  
     .MaxIterations = *max\_num*  
     .MaxChange = *max\_change*  
     .CalculateBeforeSave = *calc\_save*  
 End With  
 With ActiveWorkbook  
     .UpdateRemoteReferences = *update*  
     .PrecisionAsDisplay = *precision*  
     .Date1904 = *date\_1904*  
     .SaveLinkValues = *save\_values*  
 End With  
 With ActiveSheet  
     .TransitionExpEval = *alt\_exp*  
     .TransitionFormEntry = *alt\_form*  
 End With

**CALL**(register\_id, argument1, ...)  
 Application.Run macro := register\_id, argument1, ...  
 Calls a DLL function registered for use on the worksheet using the `REGISTER` function. In Visual Basic use the `Declare` statement to register DLL functions if they are not going to be used on the worksheet.

**CALL**("SAMPLE.DLL", "MyFunction", "AIC", argument1, argument2, ...)  
 Declare Function MyFunction Lib "SAMPLE.DLL" \_  
     (ByVal a As Integer, ByVal s As String) As Boolean  
 ' Call the function using regular Visual Basic syntax  
 MyFunction argument1, argument2

Calls a function in a DLL or code resource. In Visual Basic the Declare statement registers DLL or code resource functions. For a full description of the Declare statement, see Chapter 10, "Controlling and Communicating with Other Applications," in the Microsoft Excel Visual Basic User's Guide. The example shows how to declare and call a single function in SAMPLE.DLL returning a Boolean value and taking an integer and a string parameter.

**CALLER**( )

Application.Caller

**CANCEL.COPY**(render\_logical)

Application.CutCopyMode = render\_logical

**CANCEL.KEY**(FALSE)

Application.EnableCancelKey = xlDisabled

**CANCEL.KEY**(TRUE)

Application.EnableCancelKey = xlInterrupt

**CANCEL.KEY**(TRUE, macro\_ref)

Application.EnableCancelKey = xlErrorHandler

In Visual Basic, instead of specifying a procedure to run, cancel key trapping uses the standard Visual Basic run-time error trapping system. A trapped cancel key results in a trappable run-time error of code 18.

**CELL.PROTECTION**(locked, hidden)

With Selection  
     .Locked = locked  
     .FormulaHidden = hidden  
 End With

**CHANGE.LINK**(old\_text, new\_text, type\_of\_link)

ActiveWorkbook.ChangeLink name := old\_text, \_  
 newName := new\_text, type := type\_of\_link

**CHART.ADD.DATA**(ref, rowcol, titles, categories, replace, 1)

ActiveChart.SeriesCollection.Add source := ref, \_  
 rowcol := rowcol, seriesLabels := titles, \_  
 categoryLabels := categories, replace := replace

**CHART.ADD.DATA**(ref, rowcol, , categories, , 2)

ActiveChart.SeriesCollection.Extend source := ref, \_  
 rowcol := rowcol, categoryLabels := categories

**CHART.WIZARD**(long, ref, gallery\_num, type\_num, plot\_by, categories, ser\_titles, legend, title, x\_title, y\_title, z\_title, number\_cats, number\_titles)

ActiveChart.ChartWizard source := ref, \_  
 gallery := gallery\_num, type := type\_num, \_  
 plotBy := plot\_by, categoryLabels := number\_cats, \_  
 seriesLabels := number\_titles, \_  
 hasLegend := legend, title := titles, \_  
 categoryTitle := x\_title, valueTitle := y\_title, \_  
 extraTitle := z\_title

The categories and ser\_titles arguments are not used in Microsoft Excel 5.0. Visual Basic supplies only the categoryLabels and seriesLabels arguments. The long argument has no equivalent since the Visual Basic method is not interactive.

**CHECKBOX.PROPERTIES**(value, link, accel\_text, accel2\_text, 3d\_shading)

With Selection  
     .Value = value  
     .LinkCell = link  
     .Display3DShading = 3d\_shading  
     .Accelerator = accel\_text  
     .PhoneticAccelerator = accel2\_text  
 End With



**CHECK.COMMAND**(bar\_num, menu, command, check)      Menubars(*bar\_num*).Menus(menu). \_  
MenuItems(check).Checked = check

Checks or unchecks a command on a menu.

**CHECK.COMMAND**(bar\_num, menu, command, check, position)      Menubars(*bar\_num*).Menus(menu). \_  
MenuItems(position). \_  
MenuItems(position).Checked = position

Checks or unchecks a command on a submenu.

**CHECK.COMMAND**(bar\_num, menu, command, check)      ShortcutMenus(*shortcut\_menu*). \_  
MenuItems(check).Checked = check

Checks or unchecks a command on a shortcut menu. Shortcut menus are accessed in Visual Basic by using the ShortcutMenus method with the appropriate constant.

**CLEAR**(1)      Selection.Clear

Completely clears a worksheet range or chart area. The parameter 1 can be omitted.

**CLEAR**( )      Selection.Delete

Completely clears (deletes) the selected chart point, series, error bars, or trendline. The parameter 1 can be omitted.

**CLEAR**( )      Selection.ClearFormats

Clears formats from the selected chart walls, floors, and plot area.

**CLEAR**(2)      Selection.ClearFormats

Clears formats from a worksheet range, or the selected chart item (not the entire chart).

**CLEAR**(3)      Selection.ClearContents

Clears contents from a worksheet range.

**CLEAR**(3)      ActiveChart.ChartArea.ClearContents

Clears all data series from an entire chart with nothing selected.

**CLEAR**(4)      Selection.ClearNotes

Clears notes from a worksheet range.

**CLEAR.OUTLINE**( )      Selection.ClearOutline

**CLEAR.ROUTING.SLIP**(FALSE)      ActiveWorkbook.HasRoutingSlip = False

**CLEAR.ROUTING.SLIP**(TRUE)      ActiveWorkbook.RoutingSlip.Reset

**CLOSE**(save\_logical, route\_logical)      ActiveWindow.Close saveChange := save\_logical, \_  
routeWorkbook := route\_logical

To close a workbook regardless of how many windows it has open, use the Close method on the Workbook object. The Close method also includes a filename argument to specify the name that the workbook will be saved as if save\_logical is TRUE.

|                                                                                                                                                                                      |                                                                                                                                                                        |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>CLOSE.ALL</u></b> ( )                                                                                                                                                          | Workbooks.Close                                                                                                                                                        |
| <b><u>COLOR.PALETTE</u></b> (file_text)                                                                                                                                              | ActiveWorkbook.Colors = Workbooks(file_text).Colors                                                                                                                    |
| <b><u>COLUMN.WIDTH</u></b> (, reference, , 1)<br>Hides columns. If hiding the selection, use Selection.EntireColumn instead of the Columns method.                                   | Columns( <i>reference</i> ).Hidden = True                                                                                                                              |
| <b><u>COLUMN.WIDTH</u></b> (, reference, , 2)<br>Unhides columns.                                                                                                                    | Columns( <i>reference</i> ).Hidden = False                                                                                                                             |
| <b><u>COLUMN.WIDTH</u></b> (, reference, , 3)<br>Sizes the columns in reference to the width of the largest item in each column.                                                     | Columns( <i>reference</i> ).AutoFit                                                                                                                                    |
| <b><u>COLUMN.WIDTH</u></b> (width_num, reference)<br>Sets the column width to the specified value.                                                                                   | Columns( <i>reference</i> ).ColumnWidth = width_num                                                                                                                    |
| <b><u>COLUMN.WIDTH</u></b> (, reference, TRUE)<br>Sets the column width to the standard column width value.                                                                          | Columns( <i>reference</i> ).ColumnWidth = _<br>ActiveSheet.StandardWidth                                                                                               |
| <b><u>COLUMN.WIDTH</u></b> (, , , standard_num)<br>Sets the standard column width for all columns on the worksheet.                                                                  | ActiveSheet.StandardWidth = standard_num                                                                                                                               |
| <b><u>COMBINATION</u></b> (type_num)                                                                                                                                                 | ActiveChart.AutoFormat gallery := xlCombination, _<br>format := type_num                                                                                               |
| <b><u>CONSOLIDATE</u></b> (source_refs, function_num, top_row, left_col, create_links)                                                                                               | Selection.Consolidate sources := <i>source_refs</i> ,<br>function := <i>function_num</i> , topRow := top_row, _<br>leftColumn := left_col, createLinks := create_links |
| <b><u>CONSTRAIN.NUMERIC</u></b> (numeric_only)                                                                                                                                       | Application.ConstrainNumeric = numeric_only                                                                                                                            |
| <b><u>COPY</u></b> ( )<br>Copies the selected object, including ranges, chart items, and drawing objects.                                                                            | Selection.Copy                                                                                                                                                         |
| <b><u>COPY</u></b> (from_reference, to_reference)<br>Copies a worksheet range from one location to another. If from_reference is omitted, use Selection instead of the Range method. | Range( <i>from_reference</i> ).Copy to_reference                                                                                                                       |
| <b><u>COPY.CHART</u></b> (size_num)                                                                                                                                                  | ActiveChart.CopyPicture size := <i>size_num</i>                                                                                                                        |
| <b><u>COPY.PICTURE</u></b> (appearance_num, size_num, type_num)<br>Use for all documents except charts.                                                                              | Selection.CopyPicture _<br>appearance := <i>appearance_num</i> , _<br>format := <i>type_num</i>                                                                        |
| <b><u>COPY.PICTURE</u></b> (appearance_num, size_num, type_num)                                                                                                                      | ActiveChart.CopyPicture _<br>appearance := <i>appearance_num</i> , _<br>size := <i>size_num</i> , format := <i>type_num</i>                                            |

Use for charts.

**COPY.TOOL**(bar\_id, position)

Toolbars(bar\_id).ToolbarButtons(position).CopyFace

**CREATE.NAMES**( )

Selection.CreateNames

**CREATE.OBJECT**(2, ref1, x\_offset1, y\_offset1, ref2, x\_offset2, y\_offset2)

```
With Range(ref1)
 left = .Left + x_offset1
 top = .Top + y_offset1
End With
With Range(ref2)
 width = (.Left + x_offset2) - left
 height = (.Top + y_offset2) - top
End With
ActiveSheet.Rectangles.Add left, top, width, height
```

Creates an unfilled rectangle object. Other objects are similar. Use the macro recorder to determine the exact form of the Visual Basic method required to create a particular drawing object. Note that Visual Basic uses absolute coordinates relative to the upper left of the sheet, rather than the cell and offset positions of the Microsoft Excel 4.0 Macro Language. This example shows how to convert cell and offset coordinates into absolute coordinates.

**CREATE.PUBLISHER**(file\_text, appearance, , formats)

```
If formats >= 8 Then
 hasValu = True
 formats = formats - 8
Else
 hasValu = False
End If
If formats >= 4 Then
 hasRtf = True
 formats = formats - 4
Else
 hasRtf = False
End If
If formats >= 2 Then
 hasBiff = True
 formats = formats - 2
Else
 hasBiff = False
End If
If formats >= 1 Then
 hasPict = True
Else
 hasPict = False
End If
Selection.CreatePublisher edition := file_text, _
 appearance := appearance, _
 containsVALU := hasValu, containsRTF := hasRtf, _
 containsBIFF := hasBiff, containsPICT := hasPict
```

Publishes a range. The formats argument is split into the containsBIFF, containsPICT, containsRTF, and containsVALU parameters in Visual Basic.

**CREATE.PUBLISHER**(file\_text, appearance, size)

ActiveChart.CreatePublisher edition := file\_text, \_  
appearance := *appearance*, size := *size*

Publishes a chart.

**CUSTOM.REPEAT**(macro\_text,  
repeat\_text)

Application.OnRepeat text := repeat\_text, \_  
procedure := macro\_text

Sets the repeat menu command for a custom command.

**CUSTOM.REPEAT**(, , record\_text)

Application.RecordMacro xlmCode := record\_text

Records arbitrary text from a custom command. Visual Basic has both a basicCode parameter for recording Visual Basic code.

**CUSTOM.UNDO**(macro\_text, undo\_text)

Application.OnUndo text := undo\_text, \_  
procedure := macro\_text

**CUT**( )

Selection.Cut

Cuts the selected object to the clipboard, including ranges, chart items, and drawing objects.

**CUT**(from\_reference, to\_reference)

Range(*from\_reference*).Cut *to\_reference*

Cuts a worksheet range from one location and pastes it to another. If from\_reference is omitted, use Selection instead of Range.

## Visual Basic Equivalents for Macro Functions and Commands

Y  
Z



### D

#### DATA.DELETE( )

No direct equivalent.

#### DATA.FIND(logical)

No direct equivalent.

#### DATA.FIND.NEXT( )

No direct equivalent.

#### DATA.FIND.PREV( )

No direct equivalent.

#### DATA.FORM( )

ActiveSheet.ShowDataForm

#### DATA.LABEL(show\_option, , show\_key)

Selection.ApplyDataLabels type := *show\_option*, \_  
legendKey := show\_key

Inserts data labels for the selected point or series. If inserting data labels for the entire chart, use ActiveChart instead of Selection.

#### DATA.LABEL(show\_option, auto\_text, show\_key)

With Selection  
.Type = *show\_option*  
.AutoText = show\_option, auto\_text, show\_key  
.ShowLegendKey = show\_key  
End With

Sets data label properties for the selected label or set of labels.

**DATA.SERIES**(rowcol, type\_num, date\_num, step\_value, stop\_value, trend)

```
Selection.DataSeries rowcol := rowcol, _
type := type_num, date := date_num, _
step := step_value, stop := stop_value, _
trend := trend
```

**DEFINE.NAME**(name\_text, refers\_to, macro\_type, shortcut\_text, hidden, category, FALSE)

```
Names.Add text := ActiveSheet.Name & "!" & name_text, _
refersTo := refers_to, category := category, _
visible := Not hidden, macroType := macro_type, _
shortcutKey := shortcut_text
```

Defines a name for a particular sheet. The Visual Basic Add method defines a local name when the text of the Name begins with a sheet name followed by an exclamation point, for example "Sheet1! SampleName". This example defines the name on the active sheet, which must be a worksheet or macro sheet.

**DEFINE.NAME**(name\_text, refers\_to, macro\_type, shortcut\_text, hidden, category, TRUE)

```
Names.Add text := name_text, refersTo := refers_to, _
category := category, visible := Not hidden, _
macroType := macro_type, _
shortcutKey := shortcut_text
```

Defines a name for all sheets in the workbook. The Visual Basic Add method defines a global name when the text of the Name does not contain a sheet name and exclamation point, for example "SampleName".

**DEFINE.STYLE**(style\_text, number, font, alignment, border, pattern, protection)

```
With ActiveWorkbook.Styles.Add(name := style_text, _
basedOn := ActiveCell)
.IncludeNumber = number
.IncludeFont = font
.IncludeAlignment = alignment
.IncludeBorder = border
.IncludePatterns = pattern
.IncludeProtection = protection
```

End With

Syntax 1: Defines a new style with the attributes of the active cell.

**DEFINE.STYLE**(style\_text, 2, format\_text)

```
With ActiveWorkbook.Styles(style_text)
.NumberFormat = format_text
End With
```

Syntax 2: Modifies the number formatting of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example.

**DEFINE.STYLE**(style\_text, 3, name\_text, size\_num, bold, italic, underline, strike, color, outline, shadow, superscript, subscript)

```
With ActiveWorkbook.Styles(style_text).Font
.Name = name_text
.Size = size_num
.Bold = bold,
.Italic = italic
.Underline = underline
.Strikethrough = strike
.ColorIndex = color
.OutlineFont = outline
.Shadow = shadow
.Superscript = superscript
.Subscript = subscript
```

End With

Syntax 3: Modifies the font settings of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example.

**DEFINE.STYLE**(style\_text, 4, horiz\_align, wrap, vert\_align, orientation)

```
With ActiveWorkbook.Styles(style_text)
 .HorizontalAlignment = horiz_align
 .WrapText = wrap
 .VerticalAlignment = vert_align
 .Orientation = orientation
End With
```

Syntax 4: Modifies the alignment settings of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example.

**DEFINE.STYLE**(style\_text, 5, left, right, top, bottom, left\_color, right\_color, top\_color, bottom\_color)

```
With ActiveWorkbook.Styles(style_text).Borders
 With .Item(xlLeft)
 .LineStyle = left_color
 .ColorIndex = left_color
 End With
 With .Item(xlRight)
 .Weight = right_color
 .ColorIndex = right_color
 End With
 With .Item(xlTop)
 .LineStyle = top_color
 .ColorIndex = top_color
 End With
 With .Item(xlBottom)
 .Weight = bottom_color
 .ColorIndex = bottom_color
 End With
End With
```

Syntax 5: Modifies the border settings of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example. Use either the LineStyle or the Weight property depending on the desired border type.

**DEFINE.STYLE**(style\_text, 6, apattern, afore, aback)

```
With ActiveWorkbook.Styles(style_text).Interior
 .ColorIndex = style_text, 6, apattern, afore, aback)
 .Pattern = style_text, 6, apattern, afore, aback)
 .PatternColorIndex = style_text, 6, apattern, afore, aback)
End With
```

Syntax 6: Modifies the patterns settings of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example.

**DEFINE.STYLE**(style\_text, 7, locked, hidden)

```
With ActiveWorkbook.Styles(style_text)
 .Locked = locked
 .FormulaHidden = hidden
End With
```

Syntax 7: Modifies the protection settings of an existing style. To add a new style in Visual Basic, first use the Add method of the Styles object. See syntax 1 for an example.

**DELETE.ARROW**( )

```
Selection.Delete
```

Deletes the selected arrow object on the active chart.

**DELETE.BAR**(bar\_num)

```
MenuBar(bar_num).Delete
```

**DELETE.CHART.AUTOFORMAT**( name\_text)

```
Application.DeleteChartAutoFormat name := name_text
```

|                                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                      |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| <b><u>DELETE.COMMAND</u></b> (bar_num, menu, command, subcommand)                                                                                                                                                                                                                                                                                                                     | MenuBar( <i>bar_num</i> ).Menus(menu). _<br>MenuItems(subcommand).Delete                             |
| Deletes a menu command from a menu.                                                                                                                                                                                                                                                                                                                                                   |                                                                                                      |
| <b><u>DELETE.COMMAND</u></b> (bar_num, menu, command, subcommand)                                                                                                                                                                                                                                                                                                                     | MenuBar( <i>bar_num</i> ).Menus(menu). _<br>MenuItems(subcommand). _<br>MenuItems(subcommand).Delete |
| Deletes a command from a submenu.                                                                                                                                                                                                                                                                                                                                                     |                                                                                                      |
| <b><u>DELETE.COMMAND</u></b> (bar_num, menu, command)                                                                                                                                                                                                                                                                                                                                 | ShortcutMenus( <i>shortcut_menu</i> ). _<br>MenuItems(command).Delete                                |
| Deletes a command from a shortcut menu. Access shortcut menus in Visual Basic by using the ShortcutMenus method with the appropriate constant.                                                                                                                                                                                                                                        |                                                                                                      |
| <b><u>DELETE.FORMAT</u></b> (format_text)                                                                                                                                                                                                                                                                                                                                             | ActiveWorkbook.DeleteNumberFormat format_text                                                        |
| <b><u>DELETE.MENU</u></b> (bar_num, menu)                                                                                                                                                                                                                                                                                                                                             | MenuBar( <i>bar_num</i> ).Menus(menu).Delete                                                         |
| Deletes a menu.                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                      |
| <b><u>DELETE.MENU</u></b> (bar_num, menu, submenu)                                                                                                                                                                                                                                                                                                                                    | MenuBar( <i>bar_num</i> ).Menus(submenu). _<br>MenuItems(submenu).Delete                             |
| Deletes a submenu.                                                                                                                                                                                                                                                                                                                                                                    |                                                                                                      |
| <b><u>DELETE.NAME</u></b> (name_text)                                                                                                                                                                                                                                                                                                                                                 | Names(name_text).Delete                                                                              |
| Names defined only for the active sheet are accessed by preceeding the Name text with the sheet name followed by an exclamation point, for example "Sheet1!SampleName".                                                                                                                                                                                                               |                                                                                                      |
| <b><u>DELETE.OVERLAY</u></b> ( )                                                                                                                                                                                                                                                                                                                                                      | ActiveChart.ChartGroups(2).Type = _<br>ActiveChart.ChartGroups(1).Type                               |
| Deletes the overlay from the current chart. Microsoft Excel 5.0 uses chart groups instead of overlays, so there is no direct Visual Basic equivalent. The example given merges the second chart group, or overlay, with the first chart group. This example works only if the chart overlay was created using Microsoft Excel 4.0 macros or was read from a Microsoft Excel 4.0 file. |                                                                                                      |
| <b><u>DELETE.STYLE</u></b> (style_text)                                                                                                                                                                                                                                                                                                                                               | ActiveWorkbook.Styles(style_text).Delete                                                             |
| <b><u>DELETE.TOOL</u></b> (bar_id, position)                                                                                                                                                                                                                                                                                                                                          | Toolbars(bar_id).ToolbarButtons(position).Delete                                                     |
| <b><u>DELETE.TOOLBAR</u></b> (bar_name)                                                                                                                                                                                                                                                                                                                                               | Toolbars(bar_name).Delete                                                                            |
| <b><u>DEMOTE</u></b> (1)                                                                                                                                                                                                                                                                                                                                                              | Selection.Rows.Group                                                                                 |
| Groups the rows of the range selection.                                                                                                                                                                                                                                                                                                                                               |                                                                                                      |
| <b><u>DEMOTE</u></b> (2)                                                                                                                                                                                                                                                                                                                                                              | Selection.Columns.Group                                                                              |
| Groups the columns of the range selection.                                                                                                                                                                                                                                                                                                                                            |                                                                                                      |
| <b><u>DEREF</u></b> (reference)                                                                                                                                                                                                                                                                                                                                                       | Range( <i>reference</i> ).Value                                                                      |
| Like references in the Microsoft Excel 4.0 Macro Language, Visual Basic automatically returns the                                                                                                                                                                                                                                                                                     |                                                                                                      |



Value property in most circumstances where a Range object is used and a value is expected. Use the Value property to force a dereference if a Range is expected.

**DIALOG.BOX**(dialog\_ref)

Range(dialog\_ref).DialogBox

Microsoft Excel 5.0 supports a new style of dialog boxes that are visually edited. To use a new dialog, insert a Dialog Sheet by choosing the Dialog command on the Insert Menu, Macro submenu. For information on using dialog sheets, see Chapter 11, "Controls and Dialog Boxes" in the Microsoft Excel Visual Basic User's Guide.

**DIRECTORY**(path\_text)

ChDir path\_text

**DISABLE.INPUT**(logical)

Application.Interactive = Not logical

**DISPLAY**(formulas, gridlines, headings, zeros, color\_num, reserved, outline, page\_breaks, object\_num)

With ActiveWindow

.DisplayFormulas = formulas

.DisplayGridlines = gridlines

.DisplayHeadings = headings

.DisplayZeros = zeros

.GridlineColorIndex = color\_num

.DisplayOutline = outline

.DisplayAutomaticPageBreaks = page\_breaks

End With

ActiveWorkbook.DisplayDrawingObjects = object\_num

Controls view options for the active sheet and workspace.

**DISPLAY**(cell, formula, value, format, protection, names, precedents, dependents, note)

ActiveWindow.SetInfoDisplay cell, formula, value, \_  
mergeformat, protection, names, precedents, \_  
dependents, note

Controls options for the Info Window. To directly access the Info Window in Visual Basic use Windows("Info").

**DOCUMENTS**( )

For Each wb in Workbooks

answer = MsgBox "Print " & wb.Name & "?", vbYesNo

If answer = vbYes Then

wb.PrintOut

End If

Next

Returns an array containing the names of all open workbooks that are not add-ins. In Visual Basic, use the Workbooks collection. Use a For Each ...Next statement to iterate over each workbook in the collection. This example prompts the user whether each currently open workbook should be printed, and prints the workbook if the user responds affirmatively.

**DOCUMENTS**(, match\_text)

For Each wb In Workbooks

If wb.Name Like match\_text Then

' This workbook matches

MsgBox "Match for " & wb.Name

End If

Next

Returns an array containing the names of all open workbooks that are not add-ins with a name matching the pattern given by match\_text. Visual Basic does not have a direct equivalent, but this example simulates the name filtering by performing the filtering inside of a loop.

## **DOCUMENTS**(2)

No direct equivalent. A particular open add-in can be accessed by name using the Workbooks method, but there is no way to get a collection of all open add-ins. Similarly, DOCUMENTS(3) does not have a Visual Basic equivalent.

## **DUPLICATE**( )

Selection.Duplicate

## **E**

### **ECHO**(logical)

Application.ScreenUpdating = logical

### **EDIT.COLOR**(color\_num, red\_value, green\_value, blue\_value)

ActiveWorkbooks.Colors(color\_num) = \_  
RGB(red\_value, green\_value, blue\_value)

### **EDIT.DELETE**(1)

Selection.Delete xlToLeft

Deletes the cells of the selection. The EDIT.DELETE(2) command corresponds to the xlUp argument to the Delete method.

### **EDIT.DELETE**(3)

Selection.EntireRow.Delete

Deletes the rows containing the selection. Use EntireColumn instead of EntireRow for the EDIT.DELETE(4) command.

### **EDIT.OBJECT**(verb\_num, no\_pause)

Selection.Verb verb := *verb\_num*

Edits the embedded or linked object. The no\_pause argument has no equivalent in Visual Basic.

### **EDIT.REPEAT**( )

Application.Repeat

### **EDIT.SERIES**(series\_num, name\_ref, x\_ref, y\_ref, z\_ref, plot\_order)

ActiveChart.SeriesCollection(series\_num). \_  
PlotOrder = plot\_order

No direct equivalent. Adds a new series. Use the Add method of the SeriesCollection object to add a new series, and then set the Formula and PlotOrder properties to change series settings. The example shows how to access an existing series.

### **EDIT.TOOL**(bar\_id, position)

Toolbars(bar\_id).ToolbarButtons(position).Edit

### **EDITION.OPTIONS**(edition\_type, edition\_name, reference, option, appearance, size, formats)

ActiveWorkbook.EditionOptions type := edition\_type, \_  
option := , name := edition\_name, \_  
reference := reference, appearance := \_  
*appearance*, chartSize := *size*, formats := *formats*

### **ELSE**( )

Else keyword, part of an If...Then...Else statement

### **ELSE.IF**(logical\_test)

Elseif keyword, part of an If...Then...Else statment

### **EMBED**(object\_class, item)

No direct equivalent. Displayed in the formula bar when an embedded object is selected, but cannot be entered on a macro sheet or used in a module.

### **ENABLE.COMMAND**(bar\_num, menu, command, enable)

MenuBar(*bar\_num*).Menus(menu). \_  
MenuItems(enable).Enabled = enable

Enables or disables a command on a menu.

|                                                                           |                                                                                                                    |
|---------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| <b><u>ENABLE.COMMAND</u></b> (bar_num, menu, command, enable, subcommand) | MenuBar( <i>bar_num</i> ).Menus(menu). _<br>MenuItems(subcommand). _<br>MenuItems(subcommand).Enabled = subcommand |
|---------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|

Enables or disables a command on a submenu.

|                                                               |                                                                                |
|---------------------------------------------------------------|--------------------------------------------------------------------------------|
| <b><u>ENABLE.COMMAND</u></b> (bar_num, menu, command, enable) | ShortcutMenus( <i>shortcut_menu</i> ). _<br>MenuItems(enable).Enabled = enable |
|---------------------------------------------------------------|--------------------------------------------------------------------------------|

Enables or disables a command on a shortcut menu. Shortcut menus are accessed in Visual Basic by using the ShortcutMenus method with the appropriate constant.

|                                        |                                     |
|----------------------------------------|-------------------------------------|
| <b><u>ENABLE.TIPWIZARD</u></b> (state) | Application.EnableTipWizard = state |
|----------------------------------------|-------------------------------------|

|                                                      |                                                                  |
|------------------------------------------------------|------------------------------------------------------------------|
| <b><u>ENABLE.TOOL</u></b> (bar_id, position, enable) | Toolbars(bar_id).ToolbarButtons(position). _<br>Enabled = enable |
|------------------------------------------------------|------------------------------------------------------------------|

|                          |                                                       |
|--------------------------|-------------------------------------------------------|
| <b><u>END.IF</u></b> ( ) | End If keyword, part of an If...Then...Else statement |
|--------------------------|-------------------------------------------------------|

|                                    |                                            |
|------------------------------------|--------------------------------------------|
| <b><u>ENTER.DATA</u></b> (logical) | Application.DataEntryMode = <i>logical</i> |
|------------------------------------|--------------------------------------------|

|                             |                                                           |
|-----------------------------|-----------------------------------------------------------|
| <b><u>ERROR</u></b> (FALSE) | On Error Resume Next<br>Application.DisplayAlerts = False |
|-----------------------------|-----------------------------------------------------------|

Disables error checking and alert messages. In Visual Basic, the On Error statement causes errors to be ignored and the DisplayAlerts property disables alert messages.

|                            |                                                     |
|----------------------------|-----------------------------------------------------|
| <b><u>ERROR</u></b> (TRUE) | On Error GoTo 0<br>Application.DisplayAlerts = True |
|----------------------------|-----------------------------------------------------|

Enables alert messages and error checking, and execution stops when a runtime error occurs. In Visual Basic, the On Error statement disables all error trapping and restores the default error alert and the DisplayAlerts property enables alert messages.

|                                       |                                                                                                                                                                                                                                                                                                        |
|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>ERROR</u></b> (TRUE, macro_ref) | Sub ErrorsTrapped ( )<br>' Trap all errors, display alerts<br>Application.DisplayAlerts = True<br>On Error GoTo ErrorTrapper<br><br>' Code that potentially causes errors<br>Exit Sub<br><br>ErrorTrapper:<br>' Error recovery code<br>' Use the Resume statement to continue<br>ResumeNext<br>End Sub |
|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Traps errors and calls macro\_text when an error occurs. In Visual Basic error handlers are not a separate procedure but are a labeled portion of the current procedure. The label is specified as the argument to the On Error statement. See Chapter 9 of the Visual Basic User's Guide for detailed information on error trapping.

|                                                         |                                                                                                                 |
|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <b><u>ERRORBAR.X</u></b> (include, type, amount, minus) | Selection.ErrorBar direction := xIX, include := <i>include</i> , _<br>type := <i>type</i> , amount := amount, _ |
|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------|

minusValues := minus

**ERRORBAR.Y**(include, type, amount, minus)

Selection.ErrorBar direction := x\Y, include := *include*, \_  
type := *type*, amount := amount, \_  
minusValues := minus

**EVALUATE**(formula\_text)

Evaluate name := formula\_text

**EXEC**(program\_text, window\_num)

Shell program\_text, window\_num

Starts an application in Microsoft Windows with an initial window state of window\_num.

**EXEC**(program\_text, , TRUE, preferred\_size\_only)

Shell program\_text, 4

Starts an application on the Macintosh in the background. If the background argument is FALSE, specify 1 as the second parameter to the Shell function. The Shell method has no equivalent for the preferred\_size\_only argument.

**EXECUTE**(channel\_num, execute\_text)

DDEExecute channel\_num, execute\_text

**EXTEND.POLYGON**(array)

With Drawings.Add(x1, y1, x2, y2, closed)  
.AddVertex x3, y3  
.AddVertex x4, y4  
End With

Defines a polygon when there are more polygon vertices than can fit in a single CREATE.OBJECT command. In Visual Basic, use the AddVertex method as many times as necessary to add new vertices to a polygon. Note that the AddVertex coordinates are measured relative to the upper left of the sheet or chart, not the upper left of the polygon bounding rectangle.

**EXTRACT**(unique)

No direct equivalent. Use the AdvancedFilter method to perform extractions from of database regions.

## F

**FCLOSE**(file\_num)

Close file\_num

**FILE.CLOSE**(save\_logical, route\_logical)

ActiveWorkbook.Close saveChanges := save\_logical, \_  
routeWorkbook := route\_logical

**FILE.DELETE**(file\_text)

Kill file\_text

**FILES**(directory\_text)

```
Dim arrayFiles() As String
count = 0
f = Dir(directory_text)
Do While Len(f) > 0
 count = count + 1
 ReDim Preserve arrayFiles(1 to count)
 arrayFiles(count) = mergeformat
 f = Dir()
Loop
```

To get a list of all files in a directory you must sequentially call the Dir function in a loop as shown in the example. Since a filename is available at each step of the loop, it may not be necessary for your application to build an array of files. This examples does build an array.

|                                                                                                                                                                              |                                                                                                                                                                                                                                                                                                                                                                               |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>FILL.AUTO</u></b> (destination_ref, copy_only)                                                                                                                         | Selection.AutoFill destination := <i>destination_ref</i> , _<br>type := <i>copy_only</i>                                                                                                                                                                                                                                                                                      |
| <b><u>FILL.DOWN</u></b> ( )                                                                                                                                                  | Selection.FillDown                                                                                                                                                                                                                                                                                                                                                            |
| <b><u>FILL.GROUP</u></b> (type_num)                                                                                                                                          | ActiveWindow.SelectedSheets.FillAcrossSheets _<br>range := Selection, type := <i>type_num</i>                                                                                                                                                                                                                                                                                 |
| <b><u>FILL.LEFT</u></b> ( )                                                                                                                                                  | Selection.FillLeft                                                                                                                                                                                                                                                                                                                                                            |
| <b><u>FILL.RIGHT</u></b> ( )                                                                                                                                                 | Selection.FillRight                                                                                                                                                                                                                                                                                                                                                           |
| <b><u>FILL.UP</u></b> ( )                                                                                                                                                    | Selection.FillUp                                                                                                                                                                                                                                                                                                                                                              |
| <b><u>FILTER</u></b> ( )                                                                                                                                                     | Selection.AutoFilter                                                                                                                                                                                                                                                                                                                                                          |
| <b><u>FILTER</u></b> (field_num, criteria1, operation, criteria2)                                                                                                            | Selection.AutoFilter field := field_num, _<br>criteria1 := <i>criteria1</i> , operator := <i>operation</i> , _<br>criteria2 := <i>criteria2</i>                                                                                                                                                                                                                               |
| <b><u>FILTER.ADVANCED</u></b> (operation, list_ref, criteria_ref, copy_ref, unique)                                                                                          | Range( <i>list_ref</i> ).AdvancedFilter action := <i>operation</i> , _<br>criteria := list_ref, criteria_ref, <i>copy_ref</i> , copyToRef :=<br><i>copy_ref</i> , _<br>unique := unique                                                                                                                                                                                       |
| <b><u>FILTER.SHOW.ALL</u></b> ( )                                                                                                                                            | ActiveSheet.ShowAllData                                                                                                                                                                                                                                                                                                                                                       |
| <b><u>FIND.FILE</u></b> ( )                                                                                                                                                  | Application.FindFile                                                                                                                                                                                                                                                                                                                                                          |
| <b><u>FONT</u></b> (name_text, size_num)                                                                                                                                     | With ActiveWorkbook.Styles("Normal").Font<br>.Name = name_text<br>.Size = size_num<br>End With                                                                                                                                                                                                                                                                                |
| <b><u>FONT.PROPERTIES</u></b> (font, font Style, size, strikethrough, superscript, subscript, outline, shadow, underline, color, normal, background, start_char, char_count) | With Selection.Characters(start_char, char_count).Font<br>.Name = char_count<br>.FontStyle = font_style<br>.Size = size<br>.Strikethrough = strikethrough<br>.Superscript = superscript<br>.Subscript = subscript<br>.OutlineFont = outline<br>.Shadow = shadow<br>.Underline = <i>underline</i><br>.ColorIndex = <i>color</i><br>.Background = <i>background</i><br>End With |

Changes the font of the selected cells or text box. If both of the start\_char and chart\_count parameters are omitted, use Selection.Font in the With statement instead of Selection.Characters(start\_char, char\_count).Font. The normal parameter has no direct equivalent. You can set the font of the Normal style by copying the settings of the Normal style Font object by using ActiveWorkbook.Styles("Normal").Font.

**FOPEN**(file\_text, 1)

```
filenum = FreeFile
Open file_text For Output Access Write As #filenum
```

Opens a file with read/write permissions for text input and output. In Visual Basic, files can be opened only for output or only for input, not both. This Visual Basic example opens the file for output only. Also, Visual Basic will always create the file if it does not exist. Because of this, FOPEN(file\_text, 3) translates to the same Visual Basic code as this example.

Visual Basic offers a richer set of file operations than the Excel 4.0 Macro Language, allowing finer control over file input and output modes and access permissions. See the Visual Basic Language Reference topics "Writing Data To Files" and "Open Statement" for details on the available modes.

**FOPEN**(file\_text, 2)

```
filenum = FreeFile
Open file_text For Input Access Read As #filenum
```

Opens a file with read-only permissions for text input.

**FOR**(counter\_text, start\_num, end\_num, step\_num)

```
For counter = start_num to end_num Step step_num
```

Increments a counter variable in a loop. In Visual Basic, the counter is a variable, not a defined name.

**FOR.CELL**(ref\_name, area\_ref)

```
For Each ref_name In area_ref
```

Iterates over a range of cells. In Visual Basic, the current cell ref\_name is a variable, not a defined name.

**FOR.CELL**(ref\_name, area\_ref, TRUE)

```
For Each ref_name In area_ref
 If Not IsEmpty(ref_name.Value) Then
 ' Cell is non-blank
 End If
Next
```

Iterates over a range of cells, skipping blank cells. The For Each statement has no direct equivalent to the skip\_blanks argument, so each cell is checked within the loop.

**FORMAT.AUTO**(format\_num, number, font, alignment, border, pattern, width)

```
Selection.AutoFormat format := format_num, _
 applyNumber := number, applyFont := font, _
 applyAlignment := alignment, _
 applyBorder := border, applyPattern := pattern, _
 applyWidth := width
```

**FORMAT.CHART**(layer\_num, view, overlap, angle, gap\_width, gap\_depth, chart\_depth, doughnut\_size, axis\_num, drop, hilo, up\_down, series\_line, labels, vary)

```
With ActiveChart.ChartGroups(layer_num)
 .SubType = view
 .Overlap = overlap
 .FirstSliceAngle = angle
 .GapWidth = gap_width
 .DoughnutHoleSize = doughnut_size
 .AxisGroup = doughnut_size, axis_num, drop
 .HasDropLines = drop
 .HasHiLoLines = hilo
 .HasUpDownBars = up_down
 .HasSeriesLines = series_line
 .HasRadarAxisLabels = labels
 .VaryByCategories = vary
End With
With ActiveChart
 .GapDepth = gap_depth
 .DepthPercent = chart_depth
```

## End With

Some arguments and properties may not apply to a particular chart type.

**FORMAT.CHARTTYPE**(1, , dimension, type\_num)      Selection.Type = *charttype*

Sets the chart type of the selected series. Visual Basic does not distinguish 2D and 3D chart types, but instead combines the dimension and type\_num arguments into the Type property. The constants used for the Type property are named after the chart type, for example xlRadar, xlLine, and xl3DArea.

**FORMAT.CHARTTYPE**(2, group\_num, dimension, type\_num)      ActiveChart.ChartGroups(group\_num).Type = *charttype*

Sets the chart type of a particular chart group.

**FORMAT.CHARTTYPE**(3, , dimension, type\_num)      ActiveChart.Type = *charttype*

Sets the chart type for the entire chart.

|                                                                                                   |                                                                                                                                                                                                                            |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>FORMAT.FONT</b> (name_text, size_num, bold, italic, underline, strike, color, outline, shadow) | With Selection.Font<br>.Name = name_text<br>.Size = size_num<br>.Bold = bold<br>.Italic = italic<br>.Underline = underline<br>.Strikethrough = strike<br>.ColorIndex = color<br>.OutlineFont = outline<br>.Shadow = shadow |
|---------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## End With

Changes font settings for cells on worksheets or macrosheets.

**FORMAT.FONT**(name\_text, size\_num, bold, italic, underline, strike, color, outline, shadow, object\_id\_text, start\_num, char\_num)

With

```
ActiveSheet.DrawingObjects(object_id_text_object_id_text). _
 Characters(start_num, char_num).Font
 .Name = name_text
 .Size = size_num
 .Bold = bold
 .Italic = italic
 .Underline = underline
 .Strikethrough = strike
 .ColorIndex = color
 .OutlineFont = outline
 .Shadow = shadow
```

## End With

Changes font settings for textboxes and buttons on worksheets or macrosheets. If `object_id_text` is omitted, use `Selection` instead of `DrawingObjects(object_id_text)`. If both `start_num` and `char_num` are omitted, then you do not need to use `Characters(start_num, char_num)` in the statement.

|                                                                                                                               |                                                                                                                                                               |
|-------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>FORMAT.FONT</u></b> (color, backgd, apply,<br>name_text, size_num, bold, italic,<br>underline, strike, outline, shadow) | With Selection.Font<br>.ColorIndex = <i>color</i><br>.Background = <i>backgd</i><br>.Name = name_text<br>.Size = size_num<br>.Bold = bold<br>.Italic = italic |
|-------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|

```
.Underline = underline
.Strikethrough = strike
.OutlineFont = outline
.Shadow = shadow
```

End With

Changes font settings for unattached text on charts. The apply argument has no direct equivalent, but if TRUE can be simulated by applying settings for the Font property of each Series on the chart. Iterate over all series in the chart by using a For Each loop with the ActiveChart.SeriesCollection object).

**FORMAT.LEGEND**(position\_num)

```
Selection.Position = position_num
```

**FORMAT.MAIN**(type\_num, view, overlap, gap\_width, vary, drop, hilo, angle, gap\_depth, chart\_depth, up\_down, series\_line, labels, doughnut\_size)

```
With ActiveChart.ChartGroups(1)
.Type = type_num
.SubType = view
.Overlap = overlap
.GapWidth = gap_width
.VaryByCategories = vary
.HasDropLines = drop
.HasHiLoLines = hilo
.FirstSliceAngle = angle
.HasUpDownBars = up_down
.HasSeriesLines = series_line
.HasRadarAxisLabels = labels
.DoughnutHoleSize = doughnut_size
End With
With ActiveChart
.GapDepth = gap_depth
.DepthPercent = chart_depth
End With
```

Some arguments and properties may not apply to a particular chart type. Microsoft Excel 5.0 does not use a main chart and an overlay chart, but provides for multiple chart types with chart groups. This example formats the first chart group, which is the number of the main chart of a chart originally saved in Microsoft Excel 4.0.

**FORMAT.MOVE**(x\_offset, y\_offset, reference)

```
With Range(reference)
xpos = .Left + x_offset
ypos = .Top + y_offset
End With
With Selection
.Left = xpos
.Top = ypos
End With
```

Moves a drawing object on a worksheet or macro sheet. Visual Basic coordinates are always relative to the upper left of cell A1. This Visual Basic example adjusts the cell relative coordinates to be relative to A1. If reference is omitted, no adjustment is necessary.

**FORMAT.MOVE**(x\_pos, y\_pos)

```
With Selection
.Left = x_pos
adjVert = ActiveChart.ChartArea.Height - .Height
.Top = adjVert - y_pos
End With
```

Moves a drawing object or text on a chart. Visual Basic coordinates are based on the upper left of the object relative to the upper left of the chart. Since FORMAT.MOVE coordinates are based on the lower left corner of the object relative to the lower left of the chart, a conversion is necessary.



**FORMAT.MOVE**(explosion\_num)

Moves a doughnut or pie chart slice.

Selection.Expllosion = (explosion\_num)

**FORMAT.NUMBER**(format\_text)

Selection.NumberFormat = format\_text

**FORMAT.OVERLAY**(type\_num, view, overlap, gap\_width, vary, drop, hilo, angle, series\_dist, series\_num, up\_down, series\_line, labels)

With ActiveChart.ChartGroups(2)

```
.Type = type_num
.SubType = view
.Overlap = overlap
.GapWidth = gap_width
.VaryByCategories = vary
.HasDropLines = drop
.HasHiLoLines = hilo
.FirstSliceAngle = angle
.HasUpDownBars = up_down
.HasSeriesLines = series_line
.HasRadarAxisLabels = labels
```

End With

Some arguments and properties may not apply to a particular chart type. Microsoft Excel 5.0 does not use a main chart and an overlay chart, but provides for multiple chart types with chart groups. This example formats the second chart group, which is the number of the overlay chart of a chart originally saved in Microsoft Excel 4.0.

The series\_dist and series\_num parameters have no direct equivalent in Visual Basic. Since Microsoft Excel 5.0 can plot series with any chart group, use the Type property of the Series object to determine which chart group the series is plotted with.

**FORMAT.SHAPE**(vertex\_num, insert, , x\_offset, y\_offset)

With Selection

```
xpos = .Left + x_offset
ypos = .Top + y_offset
.Reshape vertex := vertex_num, insert := insert, _
left := xpos, top := ypos
```

End With

Moves or inserts a vertex of a polygon drawing. FORMAT.SHAPE coordinates are relative to the upper left of the polygon bounding box when the reference parameter is omitted as in this example. Visual Basic coordinates are relative to the upper left of cell A1 or the chart. To delete a vertex, specify False for insert and omit the left and top parameters.

**FORMAT.SHAPE**(vertex\_num, insert, reference, x\_offset, y\_offset)

With Range(reference)

```
xpos = .Left + x_offset
ypos = .Top + y_offset
```

End With

```
Selection.Reshape vertex := vertex_num, insert := insert, _
left := xpos, top := ypos
```

Moves or inserts a vertex of a polygon drawing. The FORMAT.SHAPE coordinates are relative to the upper left of reference. Visual Basic coordinates are relative to the upper left of cell A1 or the chart.

**FORMAT.SIZE**(width, height)

With Selection

```
.Width = width
.Height = height
```

End With

Sizes objects in absolute coordinates.

**FORMAT.SIZE**(x\_off, y\_off, reference)

set ref = Range(reference)

```

With Selection
 .Width = (ref.Left + x_off) - .Left
 .Height = (ref.Top + y_off) - .Top
End With

```

Sizes objects relative to a worksheet cell. Visual Basic coordinates are always relative to the upper left of cell A1, so a conversion is required.

**FORMAT.TEXT**(x\_align, y\_align, orient\_num, auto\_text, auto\_size)

```

With Selection
 .HorizontalAlignment = x_align
 .VerticalAlignment = y_align
 .Orientation = orient_num
 .AutoText = auto_text
 .AutoSize = auto_size
End With

```

The show\_key and show\_value arguments are not used by Microsoft Excel 5.0.

**FORMULA**(formula\_text, reference)

Range(*reference*).FormulaR1C1 = formula\_text

Enters a formula using R1C1-style references. To enter formulas using A1-style references in Visual Basic use the Formula property instead of FormulaR1C1.

**FORMULA**(formula\_text)

ActiveCell.FormulaR1C1 = formula\_text

**FORMULA.ARRAY**(formula\_text, reference)

Range(*reference*).FormulaArray = formula\_text

Enters an array formula using R1C1-style references.

**FORMULA.ARRAY**(formula\_text)

Selection.FormulaArray = formula\_text

**FORMULA.CONVERT**(formula\_text, from\_a1, to\_a1, to\_ref\_type, rel\_to\_ref)

```

Application.ConvertFormula formula := formula_text, _
 fromReferenceStyle := from_a1, _
 toReferenceStyle := to_a1, _
 toAbsolute := to_ref_type, relativeTo := rel_to_ref

```

**FORMULA.FILL**(formula\_text)

Selection.FormulaR1C1 = formula\_text

**FORMULA.FILL**(formula\_text, reference)

Range(*reference*) = formula\_text

**FORMULA.FIND**(text, in\_num, at\_num, by\_num, dir\_num, match\_case)

```

Cells.Find(what := text, after := ActiveCell, _
 lookIn := in_num, lookAt := at_num, _
 searchOrder := by_num, _
 searchDirection := dir_num, _
 matchCase := match_case).Activate

```

Searches for text in cells. If more than one cell was selected to restrict the range of the search, use Selection instead of Cells. Note that because the FIND command updates the selection, the result of the Find method is made the active cell.

**FORMULA.FIND.NEXT**( )

Cells.FindNext(after := ActiveCell).Activate

Searches for the next occurrence of text in cells. If more than one cell was selected at the start of the search, use Selection instead of Cells.

**FORMULA.FIND.PREV**( )

Cells.FindPrevious(before := ActiveCell).Activate

Searches for the previous occurrence of text in cells. If more than one cell was selected at the start of the search, use Selection instead of Cells.

**FORMULA.GOTO**(reference, corner)                      Application.Goto reference := *reference*, scroll := corner

**FORMULA.REPLACE**(find\_text, replace\_text, look\_at, look\_by, FALSE, match\_case)                      Cells.Replace what := find\_text, \_ replacement := replace\_text, whole := *look\_at*, \_ searchOrder := *look\_by*, matchCase := match\_case

Replaces text in cells. If more than one cell was selected to restrict the range of the search, use Selection instead of Cells.

**FORMULA.REPLACE**(find\_text, replace\_text, look\_at, look\_by, TRUE, match\_case)                      ActiveCell.Replace what := find\_text, \_ replacement := replace\_text, whole := *look\_at*, \_ searchOrder := *look\_by*, matchCase := match\_case

Replaces text in the active cell only.

**FPOS**(file\_num, position\_num)                      Seek file\_num, position\_num

Sets the current position in the file.

**FPOS**(file\_num)                      Seek(file\_num)

Returns the current position in the file.

**FREAD**(file\_num, num\_chars)                      Input(num\_chars, file\_num)

Reads bytes from a file. Note that the order of parameters is reversed. Unlike FREAD, Input does not have restrictions on how many bytes can be read from a file at once. Microsoft Excel methods and properties, however, can only accept strings shorter than 256 characters. Strings that are longer are truncated when passed as an argument or property value.

**FREADLN**(file\_num)                      Dim resultString as String  
Line Input # file\_num, resultString

Reads a line of text terminated by a newline from a file. The Visual Basic Line Input statement does not return the read line, instead it assigns the result to a variable that must be previously declared.

**FREEZE.PANES**(logical)                      ActiveWindow.FreezePanes = True

**FREEZE.PANES**(TRUE, col\_split, row\_split)                      With ActiveWindow  
                                                 .RowSplit = row\_split  
                                                 .ColumnSplit = col\_split  
                                                 .FreezePanes = True  
End With

**FSIZE**(file\_num)                      LOF(file\_num)

**FULL**(TRUE)                      ActiveWindow.WindowState = xlMaximized

**FULL**(FALSE)                      ActiveWindow.WindowState = xlNormal

**FULL.SCREEN**(logical)                      Application.DisplayFullScreen = logical

**FUNCTION.WIZARD**()                      Selection.FunctionWizard

**FWRITE**(file\_num, text)

Print # file\_num, text

Writes text to a file. Visual Basic does not have a limitation on string length, so the Print statement can write strings that are longer than 255 characters. Note the trailing semicolon character, which is significant to avoid writing out a trailing newline.

**FWRITELN**(file\_num, text)

Print # file\_num, text

Writes text to a file followed by a newline. Note that the Visual Basic equivalent does not have a trailing semicolon.

## Visual Basic Equivalents for Macro Functions and Commands

.....



### G

**GALLERY.3D.AREA**(type\_num)

ActiveChart.AutoFormat gallery := xl3DArea, \_  
format := type\_num

**GALLERY.3D.BAR**(type\_num)

ActiveChart.AutoFormat gallery := xl3DBar, \_  
format := type\_num

**GALLERY.3D.COLUMN**(type\_num)

ActiveChart.AutoFormat gallery := xl3DColumn, \_  
format := type\_num

**GALLERY.3D.LINE**(type\_num)

ActiveChart.AutoFormat gallery := xl3DLine, \_  
format := type\_num

**GALLERY.3D.PIE**(type\_num)

ActiveChart.AutoFormat gallery := xl3DPie, \_  
format := type\_num

**GALLERY.3D.SURFACE**(type\_num)

ActiveChart.AutoFormat gallery := xl3DSurface, \_  
format := type\_num

**GALLERY.AREA**(type\_num)

ActiveChart.AutoFormat gallery := xlArea, \_  
format := type\_num

Applies an autoformat to the active chart. The delete\_overlay parameter has no direct equivalent in Visual Basic and Microsoft Excel 5.0. To format just the selected chart group, see the following example.

**GALLERY.AREA**(type\_num, FALSE)

With Selection.Parent  
.Type = xlArea  
.SubType = type\_num

## End With

Formats just the selected chart group. Use this syntax for all of the `GALLERY.TYPE` commands if the `delete_overlay` parameter is `FALSE`. The Visual Basic equivalent gets the chart group of the selected series with `Selection.Parent`, then changes the type and subtype as appropriate.

**GALLERY.BAR**(type\_num)

```
ActiveChart.AutoFormat gallery := xlBar, _
 format := type_num
```

**GALLERY.COLUMN**(type\_num)

```
ActiveChart.AutoFormat gallery := xlColumn, _
 format := type_num
```

**GALLERY.CUSTOM**(name\_text)

```
ActiveChart.AutoFormat gallery := xlCustom, _
 format := name_text
```

**GALLERY.DOUGHNUT**(type\_num)

```
ActiveChart.AutoFormat gallery := xlDoughnut, _
 format := type_num
```

**GALLERY.LINE**(type\_num)

```
ActiveChart.AutoFormat gallery := xlLine, _
 format := type_num
```

**GALLERY.PIE**(type\_num)

```
ActiveChart.AutoFormat gallery := xlPie, _
 format := type_num
```

### GALLERY.RADAR(type\_num)

```
ActiveChart.AutoFormat gallery := xlRadar, _
 format := type_num
```

### GALLERY.SCATTER(type\_num)

```
ActiveChart.AutoFormat gallery := xlXYScatter, _
 format := type_num
```

## GET.BAR( )

## ActiveMenuBar

**Syntax 1:** Returns the ID of the active menu bar. The `ActiveMenuBar` property returns the `MenuBar` object that is active. Instead of using an ID, the object is manipulated directly.

**GET.BAR**(bar\_num, menu, command)

```
MenuBar(bar_num).Menus(menu). _
 MenuItems(command).Caption
```

Syntax 2: Returns the name, as text, of a menu command by specifying the command argument as a number.

**GET.BAR**(bar\_num, menu, command

```
MenuBar(bar_num).Menus(menu).
 MenuItems(command).Index
```

Syntax 2: Returns the index of a menu command by specifying the command argument as text.

**GET.BAR**(bar\_num, menu, command, subcommand)

```
MenuBar(bar_num).Menus(menu). _
 MenuItems(command). _
 MenuItems(subcommand).Caption
```

**Syntax 2:** Returns the name as text or the index of a command on a submenu. In Visual Basic, use the Caption property of the MenuItem object to return the name as text, or the Index property to return the index number.

**GET.CELL**(type\_num, reference)

Range(*reference*).Resize(1, 1).Property

Returns information about a cell. If reference is specified, the Visual Basic equivalent is to use the

appropriate method or property directly on a Range object constructed using the Range method. If reference is omitted, information is returned using the active cell. For clarity, the remaining translations of GET.CELL assume that reference is omitted.

**GET.CELL(1)**

```
ActiveCell.Address rowAbsolute := True, _
columnAbsolute := True, _
referenceStyle := Application.ReferenceStyle
```

Returns the absolute reference of the upper left cell of the selection. Visual Basic is not dependent on the current reference style; you can pick the reference style you want to use. In this example, GET.CELL(1) is simulated by using the Application.ReferenceStyle style property to determine what style is returned.

**GET.CELL(2)**

```
ActiveCell.Row
```

**GET.CELL(3)**

```
ActiveCell.Column
```

**GET.CELL(4)**

```
valu = ActiveCell.Value
If IsArray(valu) Then
 result = 64
ElseIf IsError(valu) Then
 result = 16
ElseIf TypeName(valu) = "Boolean" Then
 result = 4
ElseIf TypeName(valu) = "String" Then
 result = 2
Else
 result = 1
End If
```

Returns the type of the cell contents as a numeric code. In Visual Basic the use of this entire example is probably not necessary since your code can be constructed to take advantage of the IsError, IsArray, and TypeName type comparison functions.

**GET.CELL(5)**

```
ActiveCell.Value
```

**GET.CELL(6)**

```
If Application.ReferenceStyle = xlA1 Then
 result = ActiveCell.Formula
Else
 result = ActiveCell.FormulaR1C1
End If
```

Returns the formula of the cell, using the current reference style. Visual Basic is not dependent on the current reference style; you can pick the reference style you want to use. In this example, GET.CELL(6) is simulated by using the Application.ReferenceStyle style property to determine what style is returned.

**GET.CELL(7)**

```
ActiveCell.NumberFormat
```

**GET.CELL(8)**

```
ActiveCell.HorizontalAlignment
```

**GET.CELL(9)**

```
ActiveCell.Borders(xlLeft).LineStyle
ActiveCell.Borders(xlLeft).Weight
```

Returns the left border style of cell. In Visual Basic the border style is determined using one of the LineStyle and Weight properties, depending on what border type is selected. The same applies to

GET.CELL(10) through GET.CELL(12) as well.

|                             |                                                                                                                                                                  |
|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>GET.CELL</u></b> (10) | ActiveCell.Borders(xlRight).LineStyle<br>ActiveCell.Borders(xlRight).Weight                                                                                      |
| <b><u>GET.CELL</u></b> (11) | ActiveCell.Borders(xlTop).LineStyle<br>ActiveCell.Borders(xlTop).Weight                                                                                          |
| <b><u>GET.CELL</u></b> (12) | ActiveCell.Borders(xlBottom).LineStyle<br>ActiveCell.Borders(xlBottom).Weight                                                                                    |
| <b><u>GET.CELL</u></b> (13) | ActiveCell.Interior.Pattern                                                                                                                                      |
| <b><u>GET.CELL</u></b> (14) | ActiveCell.Locked                                                                                                                                                |
| <b><u>GET.CELL</u></b> (15) | ActiveCell.FormulaHidden                                                                                                                                         |
| <b><u>GET.CELL</u></b> (16) | cellWidth = ActiveCell.ColumnWidth<br>If cellWidth = ActiveCell.Parent.StandardWidth Then<br>isStandardWidth = True<br>Else<br>isStandardWidth = False<br>End If |

Unlike GET.CELL(16), the Visual Basic ColumnWidth property does not return an array. The example compares the cell width to the StandardWidth property of the worksheet to determine if the cell is sized to the standard width.

|                             |                                                                 |
|-----------------------------|-----------------------------------------------------------------|
| <b><u>GET.CELL</u></b> (17) | ActiveCell.RowHeight                                            |
| <b><u>GET.CELL</u></b> (18) | ActiveCell.Font.Name                                            |
| <b><u>GET.CELL</u></b> (19) | ActiveCell.Font.Size                                            |
| <b><u>GET.CELL</u></b> (20) | ActiveCell.Font.Bold                                            |
| <b><u>GET.CELL</u></b> (21) | ActiveCell.Font.Italic                                          |
| <b><u>GET.CELL</u></b> (22) | ActiveCell.Font.Underline <> xlNone                             |
| <b><u>GET.CELL</u></b> (23) | ActiveCell.Font.Strikethrough                                   |
| <b><u>GET.CELL</u></b> (24) | ActiveCell.Font.ColorIndex                                      |
| <b><u>GET.CELL</u></b> (25) | ActiveCell.Font.OutlineFont                                     |
| <b><u>GET.CELL</u></b> (26) | ActiveCell.Font.Shadow                                          |
| <b><u>GET.CELL</u></b> (27) | result = 0<br>If ActiveCell.EntireRow.PageBreak = xlManual Then |



```

 result = result + 1
 End If
 If ActiveCell.EntireColumn.PageBreak = xlManual Then
 result = result + 2
 End If

```

Returns whether a manual page break exists at the cell. In Visual Basic, rows and columns are individually examined for pagebreaks. The example builds a result value that is equivalent to GET.CELL(27).

**GET.CELL**(28)

ActiveCell.EntireRow.OutlineLevel

**GET.CELL**(29)

ActiveCell.EntireColumn.OutlineLevel

**GET.CELL**(30)

ActiveCell.EntireRow.Summary

**GET.CELL**(31)

ActiveCell.EntireColumn.Summary

**GET.CELL**(32)

```

Set containingBook = ActiveCell.Parent.Parent
If containingBook.Sheets.Count = 1 Then
 ' Get bookname, strip off extension (if any)
 bookName = containingBook.Name
 For i = Len(bookName) To 1 Step -1
 If Mid(bookName, i, 1) = "." Then
 bookName = Mid(bookName, 1, i - 1)
 Exit For
 End If
Next
x = StrComp(bookName, ActiveCell.Parent.Name, 1)
If x <> 0 Then Goto BuildLongName
result = containingBook.Name
Else
BuildLongName:
 result = "[" & containingBook.Name & _
 "]" & ActiveCell.Parent.Name
End If

```

Returns the name of the workbook and sheet containing the cell. The example constructs an exact equivalent of GET.CELL(32), which has special rules to provide backwards compatability with Microsoft Excel 4.0 behavior. In Visual Basic it is not necessary to construct such an equivalent because the Worksheet and Workbook objects containing the cell are used instead of a text name.

**GET.CELL**(33)

ActiveCell.WrapText

**GET.CELL**(34)

ActiveCell.Borders(xlLeft).ColorIndex

**GET.CELL**(35)

ActiveCell.Borders(xlRight).ColorIndex

**GET.CELL**(36)

ActiveCell.Borders(xlTop).ColorIndex

**GET.CELL**(37)

ActiveCell.Borders(xlBottom).ColorIndex

**GET.CELL**(38)

ActiveCell.Interior.PatternColorIndex

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                                                                   |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| <b><u>GET.CELL</u></b> (39)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Interior.ColorIndex                                                    |
| <b><u>GET.CELL</u></b> (40)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Style.Name                                                             |
| <b><u>GET.CELL</u></b> (41)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.FormulaLocal                                                           |
| <b><u>GET.CELL</u></b> (42)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Left -<br>Columns(ActiveWindow.ScrollColumn).Left                      |
| Returns the distance from the left edge of the window to the left edge of the cell. Visual Basic has no direct equivalent because the Left property of the Range object does not include the size of the window frame, the outline symbol area, or row and column headers. The example returns the distance from the left edge of the window not counting such items between the edge of the window and the edge of left-most column. In most cases this difference is not a translation issue since GET.CELL(42) is generally used only to determine the width of the cell using GET.CELL(44) - GET.CELL(42), which is directly equivalent to ActiveCell.Width. |                                                                                   |
| <b><u>GET.CELL</u></b> (43)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Top - Rows(ActiveWindow.ScrollRow).Top                                 |
| No direct equivalent. See remarks for GET.CELL(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                   |
| <b><u>GET.CELL</u></b> (44)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | (ActiveCell.Left + ActiveCell.Width) -<br>Columns(ActiveWindow.ScrollColumn).Left |
| No direct equivalent. See remarks for GET.CELL(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                   |
| <b><u>GET.CELL</u></b> (45)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | (ActiveCell.Top + ActiveCell.Height) -<br>Rows(ActiveWindow.ScrollRow).Top        |
| No direct equivalent. See remarks for GET.CELL(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                   |
| <b><u>GET.CELL</u></b> (46)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | Len(ActiveCell.NoteText()) > 0                                                    |
| <b><u>GET.CELL</u></b> (47)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ExecuteExcel4Macro("GET.CELL(47)")                                                |
| Returns whether a cell contains a sound note. Visual Basic has no direct equivalent.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                                                                                   |
| <b><u>GET.CELL</u></b> (48)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.HasFormula                                                             |
| <b><u>GET.CELL</u></b> (49)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.HasArray                                                               |
| <b><u>GET.CELL</u></b> (50)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.VerticalAlignment                                                      |
| <b><u>GET.CELL</u></b> (51)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Orientation                                                            |
| <b><u>GET.CELL</u></b> (52)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.PrefixCharacter                                                        |
| <b><u>GET.CELL</u></b> (53)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.Text                                                                   |
| <b><u>GET.CELL</u></b> (54)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.PivotTable.Name                                                        |
| <b><u>GET.CELL</u></b> (55)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.LocationInTable                                                        |
| <b><u>GET.CELL</u></b> (56)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ActiveCell.PivotField.Name                                                        |

|                             |                                                                         |
|-----------------------------|-------------------------------------------------------------------------|
| <b><u>GET.CELL</u></b> (57) | ActiveCell.Font.Superscript                                             |
| <b><u>GET.CELL</u></b> (58) | ActiveCell.Font.FontStyle                                               |
| <b><u>GET.CELL</u></b> (59) | ActiveCell.Font.Underline                                               |
| <b><u>GET.CELL</u></b> (60) | ActiveCell.Font.Subscript                                               |
| <b><u>GET.CELL</u></b> (61) | ActiveCell.PivotItem.Name                                               |
| <b><u>GET.CELL</u></b> (62) | "[" & ActiveCell.Parent.Parent.Name & "]" & _<br>ActiveCell.Parent.Name |
| <b><u>GET.CELL</u></b> (63) | ActiveCell.Interior.ColorIndex                                          |
| <b><u>GET.CELL</u></b> (64) | ActiveCell.Interior.PatternColorIndex                                   |
| <b><u>GET.CELL</u></b> (65) | ActiveCell.AddIndent                                                    |
| <b><u>GET.CELL</u></b> (66) | ActiveCell.Parent.Parent.Name                                           |

**GET.CHART.ITEM**(x\_y\_index, point\_index)

Returns position information for the selected chart item. Visual Basic has no equivalent for getting the position of data points or pie slices. If the selected object is a drawing object (such as an Oval or TextBox) use the Left, Top, Width and Height properties to return position information. If the selected object is an arrow, Visual Basic has no means of determining which direction the arrow head is facing, but you can get the position of the end points of the arrow.

**GET.CHART.ITEM**(1, 1)                      Selection.Left

Returns the horizontal coordinate of the left edge of the selected legend, chart area, plot area, or an area on an area chart. The translations for GET.CHART.ITEM(1, 7) and GET.CHART.ITEM(1, 8) are identical.

**GET.CHART.ITEM**(1, 3)                      Selection.Left + Selection.Width

Returns the horizontal coordinate of the right edge of the selected legend, chart area, plot area, or an area on an area chart. The translations for GET.CHART.ITEM(1, 4) and GET.CHART.ITEM(1, 5) are identical.

**GET.CHART.ITEM**(1, 2)                      Selection.Left + (Selection.Width / 2)

Returns the horizontal coordinate of the middle of the selected legend, chart area, plot area, rectangular data point, or an area on an area chart. The translation for GET.CHART.ITEM(1, 6) is identical.

**GET.CHART.ITEM**(2, 1)                      ActiveChart.ChartArea.Height - Selection.Top

Returns the vertical coordinate of the top of the selected legend, chart area, plot area, or an area on an area chart. The translations for GET.CHART.ITEM(2, 2) and GET.CHART.ITEM(2, 3) are identical. Note the Visual Basic coordinate origin is in the upper left, necessitating a conversion in order to return an exact equivalent. Generally, you can write your code to avoid such conversions.

**GET.CHART.ITEM**(2, 7)ActiveChart.ChartArea.Height - Selection.Top - \_  
Selection.Height

Returns the vertical coordinate of the bottom of the selected legend, chart area, plot area, or an area on an area chart. The translations for GET.CHART.ITEM(2, 6) and GET.CHART.ITEM(2, 5) are identical.

**GET.CHART.ITEM**(2, 8)ActiveChart.ChartArea.Height - Selection.Top - \_  
(Selection.Height / 2)

Returns the vertical coordinate of the vertical midpoint of the selected legend, chart area, plot area, or an area on an area chart. The translation for GET.CHART.ITEM(2, 4) is identical.

**GET.DEF**(def\_text, , type\_num)

ActiveWorkbook.Names(refersTo := def\_text).Name

Returns the name defined as the reference. Visual Basic has a Name object, and the Name property of that object is used to return the textual name of the Name. Visual Basic does not have a equivalent to the type\_num argument because all names are equally visible in Visual Basic, they are only hidden to the user. The Hidden property of the Name object indicates whether a particular name is hidden or not.

**GET.DOCUMENT**(type\_num, name\_text)

Sheets(name\_text).Property

Returns information about a sheet. If name\_text is specified, use the appropriate Visual Basic method or property directly on an object obtained using the Sheets method. If name\_text is omitted, information is returned on the active sheet. For clarity, the remaining translations of GET.DOCUMENT assume that name\_text is omitted.

Note that some Visual Basic equivalents use the ActiveWorkbook object instead of a sheet object, reflecting the workbook organization of sheets in Microsoft Excel 5.0. See also GET.WORKBOOK.

**GET.DOCUMENT**(1)

```

If ActiveWorkbook.Sheets.Count = 1 Then
 ' Get bookname, strip off extension (if any)
 bookName = ActiveWorkbook.Name
 For i = Len(bookName) To 1 Step -1
 If Mid(bookName, i, 1) = "." Then
 bookName = Mid(bookName, 1, i - 1)
 Exit For
 End If
Next
x = StrComp(bookName, ActiveSheet.Name, 1)
If x <> 0 Then Goto BuildLongName
result = ActiveWorkbook.Name
Else
BuildLongName:
 result = "[" & ActiveWorkbook.Name & _
 "]" & ActiveSheet.Name
End If

```

Returns the name of the active workbook and sheet. The example constructs an exact equivalent of GET.DOCUMENT(1), which has special rules to provide backwards compatability with Microsoft Excel 4.0 behavior. In Visual Basic it is not necessary to construct such an equivalent because the Worksheet and Workbook objects are used instead of a text name.

**GET.DOCUMENT**(2)

ActiveWorkbook.Path

**GET.DOCUMENT**(3)

```

If ActiveWindow.Type = xlInfo Then
 result = 4

```

```

Else
 Select Case TypeName(ActiveSheet)
 Case "Chart"
 result = 2
 Case "Module"
 result = 6
 Case "DialogSheet"
 result = 7
 Case Else
 If ActiveSheet.Type = xlWorksheet Then
 result = 1
 Else
 result = 3
 End If
 End Select
End If

```

Returns the type of the active sheet as a number. In Visual Basic you can compare types of sheets directly.

#### **GET.DOCUMENT**(4)

No direct equivalent. The GET.DOCUMENT command returns whether changes have been made to the current sheet. Visual Basic can only determine if changes have been made to the workbook, which is determined in the Microsoft Excel 4.0 Macro Language by GET.WORKBOOK(24).

#### **GET.DOCUMENT**(5)

ActiveWorkbook.ReadOnly

#### **GET.DOCUMENT**(6)

ActiveWorkbook.HasPassword

#### **GET.DOCUMENT**(7)

ActiveSheet.ProtectContents

#### **GET.DOCUMENT**(8)

ActiveWorkbook.ProtectWindows

#### **GET.DOCUMENT**(9)

ActiveChart.ChartGroups(1).Type

If the active sheet is a chart, returns the type of the main chart. Instead of using main and overlay charts, Microsoft Excel 5.0 allows multiple chart groups on the same chart. This example returns the type of the first chart group, which is the number of the main chart if the chart was created in Microsoft Excel 4.0.

#### **GET.DOCUMENT**(9)

ActiveSheet.UsedRange.Row

If the active sheet is a worksheet or macro sheet, returns the number of the first used row.

#### **GET.DOCUMENT**(10)

ActiveChart.ChartGroups(2).Type

If the active sheet is a chart, returns the type of the overlay chart. This example returns the type of the second chart group, which is the number of the overlay chart if the chart was created in Microsoft Excel 4.0.

#### **GET.DOCUMENT**(10)

```

With ActiveSheet.UsedRange
 result = .Rows(.Rows.Count).Row
End With

```

If the active sheet is a worksheet or macro sheet, returns the number of the last used row.

#### **GET.DOCUMENT**(11)

ActiveChart.ChartGroups(1).SeriesCollection.Count

If the active sheet is a chart, returns the number of series in the main chart. See remarks for GET.DOCUMENT(9) concerning main charts in Microsoft Excel 5.0.

**GET.DOCUMENT**(11)

ActiveSheet.UsedRange.Column

If the active sheet is a worksheet or macro sheet, returns the number of the first used column.

**GET.DOCUMENT**(12)

ActiveChart.ChartGroups(2).SeriesCollection.Count

If the active sheet is a chart, returns the number of series in the overlay chart. See remarks for GET.DOCUMENT(10) concerning overlay charts in Microsoft Excel 5.0.

**GET.DOCUMENT**(12)

With ActiveSheet.UsedRange  
result = .Columns(.Columns.Count).Column  
End With

If the active sheet is a worksheet or macro sheet, returns the number of the last used column.

**GET.DOCUMENT**(13)

ActiveWorkbook.Windows.Count

Returns the number of windows for the active workbook. Note that in Visual Basic windows are accounted for on a per-workbook basis, not per-sheet.

**GET.DOCUMENT**(14)

Application.Calculation

**GET.DOCUMENT**(15)

Application.Iteration

**GET.DOCUMENT**(16)

Application.MaxIterations

**GET.DOCUMENT**(17)

Application.MaxChange

**GET.DOCUMENT**(18)

ActiveWorkbook.UpdateRemoteReferences

**GET.DOCUMENT**(19)

ActiveWorkbook.PrecisionAsDisplayed

**GET.DOCUMENT**(20)

ActiveWorkbook.Date1904

**GET.DOCUMENT**(21) through  
**GET.DOCUMENT**(29)

Returns attributes of the four default fonts in previous versions of Microsoft Excel. Visual Basic has no direct equivalents for these values, since they are provided for macro compatibility only.

**GET.DOCUMENT**(30)

ActiveSheet.ConsolidationSources

**GET.DOCUMENT**(31)

ActiveSheet.ConsolidationFunction

**GET.DOCUMENT**(32)

ActiveSheet.ConsolidationOptions

**GET.DOCUMENT**(33)

Application.CalculateBeforeSave

**GET.DOCUMENT**(34)

ActiveWorkbook.ReadOnlyRecommended

**GET.DOCUMENT**(35)

ActiveWorkbook.WriteReserved



Returns the number of pages of notes that will be printed. Visual Basic has no equivalent.

**GET.DOCUMENT**(52)

```
With ActiveSheet.PageSetup
 left = .LeftMargin
 right = .RightMargin
 top = .TopMargin
 bottom = .BottomMargin
End With
```

Returns the printing margin settings. In Visual Basic you can access these settings directly as properties. The values are always in points. The example saves the property values but does not create an array.

**GET.DOCUMENT**(53)

```
ActiveSheet.PageSetup.Orientation
```

**GET.DOCUMENT**(54)

```
With ActiveSheet.PageSetup
 result = "&L" & .LeftHeader
 result = result & "&C" & .CenterHeader
 result = result & "&R" & .RightHeader
End With
```

**GET.DOCUMENT**(55)

```
With ActiveSheet.PageSetup
 result = "&L" & .LeftFooter
 result = result & "&C" & .CenterFooter
 result = result & "&R" & .RightFooter
End With
```

**GET.DOCUMENT**(56)

```
Array(ActiveSheet.PageSetup.CenterHorizontally, _
 ActiveSheet.PageSetup.CenterVertically)
```

**GET.DOCUMENT**(57)

```
ActiveSheet.PageSetup.PrintHeadings
```

**GET.DOCUMENT**(58)

```
ActiveSheet.PageSetup.PrintGridlines
```

**GET.DOCUMENT**(59)

```
ActiveSheet.PageSetup.BlackAndWhite
```

**GET.DOCUMENT**(60)

```
ActiveSheet.PageSetup.ChartSize
```

**GET.DOCUMENT**(61)

```
ActiveSheet.PageSetup.Order
```

**GET.DOCUMENT**(62)

```
ActiveSheet.PageSetup.Zoom
```

**GET.DOCUMENT**(63)

```
Array(ActiveSheet.PageSetup.FitToPagesWide, _
 ActiveSheet.PageSetup.FitToPagesTall)
```

**GET.DOCUMENT**(64)

```
Dim rNums() As Integer
count = 0
For Each row In ActiveSheet.UsedRange.Rows
 If row.PageBreak <> xlNone Then
 count = count + 1
 ReDim Preserve rNums(1 To count)
 rNums(count) = row.Column
```



```
End If
Next
```

Returns an array of row numbers corresponding to rows immediately below a manual or automatic page break. Visual Basic has no direct equivalent. The example builds an equivalent array from the used range of the sheet.

#### **GET.DOCUMENT**(65)

```
Dim cNums() As Integer
count = 0
For Each col In ActiveSheet.UsedRange.Columns
 If col.PageBreak <> xlNone Then
 count = count + 1
 ReDim Preserve cNums(1 To count)
 cNums(count) = col.Column
 End If
Next
```

Returns an array of column numbers corresponding to columns immediately to the right of a manual or automatic page break. Visual Basic has no direct equivalent. The example builds an equivalent array from the used range of the sheet.

#### **GET.DOCUMENT**(66)

```
ActiveSheet.TransitionFormEntry
```

#### **GET.DOCUMENT**(67)

```
True
```

Visual Basic always returns True because Microsoft Excel 5.0 does not support unbound workbooks.

#### **GET.DOCUMENT**(68)

```
ActiveSheet.Parent.Name
```

#### **GET.DOCUMENT**(69)

```
ActiveSheet.DisplayAutomaticPageBreaks
```

#### **GET.DOCUMENT**(70)

```
Dim result()
ReDim result(1 To ActiveSheet.PivotTables.Count)
x = 1
For Each t In ActiveSheet.PivotTables
 result(x) = t.Name
 x = x + 1
Next
```

Returns an array of the names of all pivot tables on the active sheet. Visual Basic has no direct equivalent. The PivotTables collection can be used to iterate all pivot tables. The example builds an array of pivot table names by using the PivotTables collection.

#### **GET.DOCUMENT**(71)

```
Dim result()
ReDim result(1 To ActiveWorkbook.Styles.Count)
x = 1
For Each s In ActiveWorkbook.Styles
 result(x) = s.Name
 x = x + 1
Next
```

Returns an array of the names of all styles in the current workbook. Visual Basic has no direct equivalent. The Styles collection can be used to iterate all styles. The example builds an array of style names by using the Styles collection.

#### **GET.DOCUMENT**(72)

```
Dim result()
ReDim result(1 To ActiveChart.ChartGroups.Count)
x = 1
```

```

For Each g In ActiveChart.ChartGroups
 result(x) = g.Type
 x = x + 1
Next

```

Returns an array of the chart group types on the current chart. Visual Basic has no direct equivalent. The ChartGroups collection can be used to iterate all groups. The example builds an array of chart group types by using the ChartGroups collection.

#### **GET.DOCUMENT**(73)

```

Dim result()
ReDim result(1 To ActiveChart.ChartGroups.Count)
x = 1
For Each g In ActiveChart.ChartGroups
 seriesCount = 0
 For Each s In g.SeriesCollection
 seriesCount = seriesCount + 1
 Next
 result(x) = seriesCount
 x = x + 1
Next

```

Returns an array of the count of series in each chart group on the current chart. Visual Basic has no direct equivalent. The example builds an array of the series count for each chart group by iterating the ChartGroup collection.

#### **GET.DOCUMENT**(74)

ActiveSheet.Focus

#### **GET.DOCUMENT**(75)

ActiveSheet.DefaultButton

#### **GET.DOCUMENT**(76)

"[" & ActiveSheet.Parent.Name & "]" & ActiveSheet.Name

#### **GET.DOCUMENT**(77)

ActiveSheet.PageSetup.PaperSize

#### **GET.DOCUMENT**(78)

ActiveSheet.PageSetup.PrintQuality

#### **GET.DOCUMENT**(79)

ActiveSheet.PageSetup.Draft

#### **GET.DOCUMENT**(80)

ActiveSheet.PageSetup.PrintNotes

#### **GET.DOCUMENT**(81)

ActiveSheet.PageSetup.PrintArea

#### **GET.DOCUMENT**(82)

```

Array(ActiveSheet.PageSetup.PrintTitleRows, _
 ActiveSheet.PageSetup.PrintTitleColumns)

```

Returns the reference addresses of the print title rows and columns. Visual Basic always returns the address in A1 style notation.

#### **GET.DOCUMENT**(83)

ActiveSheet.ProtectScenarios

#### **GET.DOCUMENT**(84)

ActiveSheet.CircularReference

#### **GET.DOCUMENT**(85)

ActiveSheet.FilterMode

#### **GET.DOCUMENT**(86)

ActiveSheet.AutoFilterMode

**GET.DOCUMENT**(87)

ActiveSheet.Index

**GET.DOCUMENT**(88)

ActiveSheet.Parent.Name

**GET.FORMULA**(reference)Range(*reference*).Resize(1, 1).FormulaR1C1

Returns the formula of a the upper-left cell of the referenced range.

**GET.FORMULA**("Picture 1")

ActiveSheet.Pictures("Picture 1").Formula

Returns the formula of a linked picture created with the Camera tool. Visual Basic returns the formula using A1 style references.

**GET.FORMULA**("S3")

ActiveChart.SeriesCollection(3).FormulaR1C1

Returns the series formula.

**GET.LINK.INFO**(link\_text, type\_num, type\_of\_link, reference)ActiveWorkbook.LinkInfo(name := link\_text, \_  
linkInfo := *type\_num*, type := *type\_of\_link*, \_  
editionRef := reference)**GET.NAME**("!Profit", 1)

ActiveWorkbook.Names("Profit").RefersToR1C1

Returns the definition of a name defined in the active workbook.

**GET.NAME**("[BOOK1.XLS]Sheet1!Profit", 1)

Workbooks("BOOK1").Names("Profit").RefersToR1C1

Returns the definition of a name defined in a particular workbook.

**GET.NAME**("Sheet1!Profit", 2)

```
Function DefinedLocally(wb As Object, sheet As String, _
 n As String)
 DefinedLocally = True
 On Error Goto NotDefined
 wb.Names(sheet & "!" & n)
 Exit Function
NotDefined:
 DefinedLocally = False
 Exit Function
End Function
```

Returns TRUE if the name is defined only for the specified sheet, or FALSE if defined for the entire workbook. Visual Basic has no direct equivalent, but if the name of the name contains an exclamation point, the name is defined only for the sheet. The example is a function that tries to retrieve the name with an exclamation point. If the name cannot be retrieved with an exclamation point, the name must be defined for the entire workbook.

**GET.NOTE**(cell\_ref, start\_char, num\_chars)Range(*cell\_ref*).NoteText(start := start\_char, \_  
length := num\_chars)

Returns the text of a cell note. Use ActiveCell instead of Range if the cell\_ref argument is omitted.

**GET.OBJECT**(type\_num, object\_id\_text, start\_num, count\_num)

ActiveSheet.DrawingObjects(object\_id\_text).Property

Returns information about a drawing object. If object\_id\_text is specified, use the appropriate Visual Basic method or property directly on an object obtained using the DrawingObjects method. If object\_id\_text is omitted, information is returned on the selected object. For clarity, the remaining

translations of GET.OBJECT assume that object\_id\_text is omitted.

The start\_num, count\_num, and item\_index arguments are used for certain values of type\_num. This translation list includes these arguments only where appropriate.

**GET.OBJECT**(1)

```
Select Case TypeName(Selection)
 Case "GroupObject"
 result = 0
 Case "Line"
 result = 1
 Case "Rectangle"
 result = 2
 Case "Oval"
 result = 3
 Case "Arc"
 result = 4
 Case "ChartObject"
 result = 5
 Case "TextBox"
 result = 6
 Case "Button"
 result = 7
 Case "Picture", "OLEObject"
 result = 8
 Case "Drawing"
 result = 9
 Case "CheckBox"
 result = 11
 Case "OptionButton"
 result = 12
 Case "EditBox"
 result = 13
 Case "Label"
 result = 14
 Case "DialogFrame"
 result = 15
 Case "Spinner"
 result = 16
 Case "ScrollBar"
 result = 17
 Case "ListBox"
 result = 18
 Case "GroupBox"
 result = 19
 Case "DropDown"
 result = 20
 Case Else
 result = CVErr(xlErrValue)
End Select
```

Determines the type of the selected object. Visual Basic has no direct equivalent. Type example computes the equivalent by examining the object type name. Visual Basic cannot detect the difference between open and closed polygons.

**GET.OBJECT**(2)

Selection.Locked

**GET.OBJECT**(3)

Selection.ZOrder

**GET.OBJECT**(4)

Selection.TopLeftCell

**GET.OBJECT**(5)

Selection.Left - Selection.TopLeftCell.Left

Returns the distance from the upper left cell to the left edge of the object. Visual Basic has no direct equivalent because it measures drawing object coordinates relative to the upper left of the sheet. The example converts the sheet-relative coordinates into cell-relative coordinates.

**GET.OBJECT**(6)

Selection.Top - Selection.TopLeftCell.Top

Returns the distance from the upper left cell to the top edge of the object. Visual Basic has no direct equivalent because it measures drawing object coordinates relative to the upper left of the sheet. The example converts the sheet-relative coordinates into cell-relative coordinates.

**GET.OBJECT**(7)

Selection.BottomRightCell

**GET.OBJECT**(8)(Selection.Left + Selection.Width) - \_  
Selection.BottomRightCell.Left

Returns the distance from the upper left corner of the bottom right cell to the right edge of the object. Visual Basic has no direct equivalent because it measures drawing object coordinates relative to the upper left of the sheet. The example converts the sheet-relative coordinates into cell-relative coordinates.

**GET.OBJECT**(9)(Selection.Top + Selection.Height) - \_  
Selection.BottomRightCell.Top

Returns the distance from the upper left corner of the bottom right cell cell to the bottom edge of the object. Visual Basic has no direct equivalent because it measures drawing object coordinates relative to the upper left of the sheet. The example converts the sheet-relative coordinates into cell-relative coordinates.

**GET.OBJECT**(10)

Selection.OnAction

**GET.OBJECT**(11)

Selection.Placement

**GET.OBJECT**(12, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num)

**GET.OBJECT**(13, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.Name

**GET.OBJECT**(14, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.Size

**GET.OBJECT**(15, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.Bold

**GET.OBJECT**(16, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.Italic

**GET.OBJECT**(17, , start\_num, count\_num)Selection.Characters(start\_num, count\_num).Font.Underline \_  
<> xlNone**GET.OBJECT**(18, , start\_num, count\_num)

Selection.Characters(start\_num, ount\_num).Font.Strikethrough

**GET.OBJECT**(19, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.OutlineFont

**GET.OBJECT**(20, , start\_num, count\_num)

Selection.Characters(start\_num, count\_num).Font.Shadow

**GET.OBJECT**(21, , start\_num, count\_num)      Selection.Characters(start\_num, count\_num).Font.ColorIndex

**GET.OBJECT**(22)      Selection.HorizontalAlignment

**GET.OBJECT**(23)      Selection.VerticalAlignment

**GET.OBJECT**(24)      Selection.Orientation

**GET.OBJECT**(25)      Selection.AutoSize

**GET.OBJECT**(26)      Selection.Visible

**GET.OBJECT**(27)      Istyle = Selection.Border.LineStyle  
If Istyle = xlAutomatic Then  
    result = 1  
Elseif Istyle = xlNone Then  
    result = 2  
Else  
    result = 0  
End If

Returns the type of the border or line. In Visual Basic, use the LineStyle property directly.

**GET.OBJECT**(28)      Selection.Border.LineStyle

Returns the border or line style.

**GET.OBJECT**(29)      Selection.Border.ColorIndex

**GET.OBJECT**(30)      Selection.Border.Weight

**GET.OBJECT**(31)      pat = Selection.Interior.Pattern  
If pat = xlAutomatic Then  
    result = 1  
Elseif pat = xlNone Then  
    result = 2  
Else  
    result = 0  
End If

Returns the type of fill. In Visual Basic, use the Pattern property directly.

**GET.OBJECT**(32)      Selection.Interior.Pattern

**GET.OBJECT**(type\_num, object\_id\_text,  
start\_num, count\_num)      Selection.Interior.PatternColorIndex

**GET.OBJECT**(34)      Selection.Interior.ColorIndex

**GET.OBJECT**(35)      Selection.ArrowHeadWidth

**GET.OBJECT**(36)      Selection.ArrowHeadLength

|                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|--------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>GET.OBJECT</u></b> (37)              | Selection.ArrowHeadStyle                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b><u>GET.OBJECT</u></b> (38)              | Selection.RoundedCorners                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b><u>GET.OBJECT</u></b> (39)              | Selection.Shadow                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <b><u>GET.OBJECT</u></b> (40)              | Selection.LockedText                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <b><u>GET.OBJECT</u></b> (41)              | Selection.PrintObject                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <b><u>GET.OBJECT</u></b> (42)              | Selection.Left - Columns(ActiveWindow.ScrollColumn).Left<br>Returns the distance from the left edge of the window to the left edge of the selected object. Visual Basic has no direct equivalent because the Left property of drawing objects does not include the size of the window frame, the outline symbol area, or row and column headers. The example returns the distance from the left edge of the window not counting such items between the edge of the window and the edge of the left-most column. In most cases this difference is not a translation issue since GET.OBJECT(42) is generally used only to determine the width of the object using GET.OBJECT(44) - GET.OBJECT(42), which is directly equivalent to Selection.Width. |
| <b><u>GET.OBJECT</u></b> (43)              | Selection.Top - Rows(ActiveWindow.ScrollRow).Top<br>No direct equivalent. See remarks for GET.OBJECT(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b><u>GET.OBJECT</u></b> (44)              | (Selection.Left + Selection.Width) - _<br>Columns(ActiveWindow.ScrollColumn).Left<br>No direct equivalent. See remarks for GET.OBJECT(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b><u>GET.OBJECT</u></b> (45)              | (Selection.Top + Selection.Height) - _<br>Rows(ActiveWindow.ScrollRow).Top<br>No direct equivalent. See remarks for GET.OBJECT(42).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b><u>GET.OBJECT</u></b> (46)              | UBound(Selection.Vertices, 1)<br>Returns the number of vertices in a polygon. Visual Basic cannot determine the number of vertices if there are more than 255 vertices in a polygon, in which case 255 is always returned. In such cases, use ExecuteExcel4Macro("GET.OBJECT(46)").                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b><u>GET.OBJECT</u></b> (47, , start_num) | Selection.Vertices<br>Returns an array of polygon vertices starting at vertex start_num. Visual Basic has no equivalent for the start_num parameter, which is assumed to be 1 at all times. Also, the Visual Basic Vertices property cannot return more than 255 vertices as an array. To access the vertices numbered beyond 255, retrieve a single coordinate by passing row and column index parameters to the Vertices property. For example, Selection.Vertices(500, 1) retrieves the x coordinate of the five hundredth vertex.                                                                                                                                                                                                             |
| <b><u>GET.OBJECT</u></b> (48)              | Selection.Formula<br>Returns the formula of the cell linked to a text box or button. Visual Basic always returns an A1-style reference.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b><u>GET.OBJECT</u></b> (48)              | Selection.LinkedCell<br>Returns the formula of the cell linked to an on-sheet control. Visual Basic always returns an A1-style reference.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |





|                               |                                     |
|-------------------------------|-------------------------------------|
| <b><u>GET.OBJECT</u></b> (70) | Selection.LinkObject                |
| <b><u>GET.OBJECT</u></b> (71) | Selection.ListCount                 |
| <b><u>GET.OBJECT</u></b> (72) | Selection.List(Selection.ListIndex) |
| <b><u>GET.OBJECT</u></b> (73) | Selection.ListFillRange             |
| <b><u>GET.OBJECT</u></b> (74) | Selection.DropDownLines             |
| <b><u>GET.OBJECT</u></b> (75) | Selection.Display3DShading          |
| <b><u>GET.OBJECT</u></b> (76) | Selection.PhoneticAccelerator       |
| <b><u>GET.OBJECT</u></b> (77) | Selection.MultiSelect               |
| <b><u>GET.OBJECT</u></b> (78) | Selection.Selected                  |
| <b><u>GET.OBJECT</u></b> (79) | Selection.AddIndent                 |

**GET.PIVOT.FIELD**(type\_num, pivot\_field\_name, pivot\_table\_name)      ActiveSheet.PivotTables(pivot\_table\_name). \_  
PivotFields(pivot\_field\_name).*Property*

Returns information about a pivot field in a pivot table. If pivot\_field\_name and pivot\_table\_name are specified, use the appropriate Visual Basic method or property directly on an object returned by the PivotTables and PivotFields methods. If omitted, returns information on the pivot field containing the active cell. For clarity, the remaining translations of GET.PIVOT.FIELD assume that pivot\_field\_name and pivot\_table\_name are omitted.

**GET.PIVOT.FIELD**(1)

```

Dim result
Set pvtlItems = ActiveCell.PivotField.PivotlItems
If pvtlItems.Count = 0 Then
 result = CVErr(xlErrNA)
Else
 ReDim result(1 To pvtlItems.Count)
 x = 1
 For Each i In pvtlItems
 result(x) = i.Name
 x = x + 1
 Next
End If

```

Returns an array of pivot item names for the items making up the field. Visual Basic has no direct equivalent; however, the PivotlItems method returns a collection which can be iterated to examine the items. The example builds an equivalent array by iterating the collection.

Replace the PivotlItems method with VisiblelItems, HiddenlItems, ParentlItems, or ChildlItems to make an equivalent for GET.PIVOT.FIELD() type\_num values of 2, 3, 5, and 17 respectively.

**GET.PIVOT.FIELD**(2)

Returns an array of pivot item names for the visible items in the field. Visual Basic has no direct equivalent; however, the VisiblelItems method returns a collection which can be iterated to examine the items. See the example for GET.PIVOT.FIELD(1) and use the VisiblelItems method.

**GET.PIVOT.FIELD**(3)

Returns an array of pivot item names for the hidden items in the field. Visual Basic has no direct equivalent; however, the HiddenItems method returns a collection which can be iterated to examine the items. See the example for GET.PIVOT.FIELD(1) and use the HiddenItems method.

**GET.PIVOT.FIELD**(4)

ActiveCell.PivotField.Orientation

**GET.PIVOT.FIELD**(5)

Returns an array of pivot item names for all items in the field that are group parents. Visual Basic has no direct equivalent; however, the ParentItems method returns a collection which can be iterated to examine the items. See the example for GET.PIVOT.FIELD(1) and use the ParentItems method.

**GET.PIVOT.FIELD**(6)

ActiveCell.PivotField.Subtotals

No direct equivalent. Visual Basic returns an array of boolean values instead of a bitstring value. Examine each entry of the array to determine if a particular subtotal type is enabled.

**GET.PIVOT.FIELD**(7)

ActiveCell.PivotField.DataType

**GET.PIVOT.FIELD**(8)

With ActiveCell.PivotField  
     result = Array(.Function, .Calculation, \_  
                                                             .BaseField, .Baseltem, .NumberFormat)  
 End With

Returns an array of settings for the pivot field. Visual Basic allows examination of individual properties, but the example constructs an array equivalent to GET.PIVOT.FIELD(8).

**GET.PIVOT.FIELD**(9)

ActiveCell.PivotField.DataRange

**GET.PIVOT.FIELD**(10)

ActiveCell.PivotField.LabelRange

**GET.PIVOT.FIELD**(11)

ActiveCell.PivotField.TotalLevels

**GET.PIVOT.FIELD**(12)

ActiveCell.PivotField.GroupLevel

**GET.PIVOT.FIELD**(13)

ActiveCell.PivotField.ParentField

**GET.PIVOT.FIELD**(type\_num,  
 pivot\_field\_name, pivot\_table\_name)

ActiveCell.PivotField.ChildField

**GET.PIVOT.FIELD**(15)

ActiveCell.PivotField.SourceName

**GET.PIVOT.FIELD**(16)

ActiveCell.PivotField.Position

**GET.PIVOT.FIELD**(17)

Returns an array of pivot item names for all items in the field that are group children. Visual Basic has no direct equivalent; however, the ChildItems method returns a collection which can be iterated to examine the items. See the example for GET.PIVOT.FIELD(1) and use the ChildItems method.

**GET.PIVOT.ITEM**(type\_num,

ActiveSheet.PivotTables(pivot\_table\_name). \_

pivot\_item\_name, pivot\_field\_name,  
pivot\_table\_name)

PivotFields(pivot\_field\_name).\_  
PivotItems(pivot\_item\_name).Property

Returns information about a pivot item in a pivot table. If pivot\_item\_name, pivot\_field\_name and pivot\_table\_name are specified, use the appropriate Visual Basic method or property directly on an object returned by the PivotTables, PivotFields and PivotItems methods. If omitted, returns information on the pivot item containing the active cell. For clarity, the remaining translations of GET.PIVOT.ITEM assume that pivot\_item\_name, pivot\_field\_name and pivot\_table\_name are omitted.

**GET.PIVOT.ITEM**(1)

ActiveCell.PivotItem.Position

**GET.PIVOT.ITEM**(2)

ActiveCell.PivotItem.LabelRange

**GET.PIVOT.ITEM**(3)

ActiveCell.PivotItem.DataRange

**GET.PIVOT.ITEM**(4)

```
Dim result
Set chldItems = ActiveCell.PivotItem.ChildItems
If chldItems.Count = 0 Then
 result = CVErr(xlErrNA)
Else
 ReDim result(1 To chldItems.Count)
 x = 1
 For Each i In chldItems
 result(x) = i.Name
 x = x + 1
 Next
End If
```

Returns an array of pivot item names for all child items of the current item if the item is a parent. Visual Basic has no direct equivalent; however, the ChildItems method returns a collection which can be iterated to examine the items. The example builds an equivalent array by iterating the collection.

**GET.PIVOT.ITEM**(5)

ActiveCell.PivotItem.ParentItem

**GET.PIVOT.ITEM**(6)

ActiveCell.PivotItem.ParentShowDetail

**GET.PIVOT.ITEM**(7)

ActiveCell.PivotItem.ShowDetail

**GET.PIVOT.ITEM**(8)

ActiveCell.PivotItem.Visible

**GET.PIVOT.ITEM**(9)

ActiveCell.PivotItem.SourceName

**GET.PIVOT.TABLE**(type\_num,  
pivot\_table\_name)

ActiveSheet.PivotTables(pivot\_table\_name).Property

Returns information about a pivot table. If pivot\_table\_name is specified, use the appropriate Visual Basic method or property directly on an object returned by the PivotTables method. If omitted, returns information about the pivot table containing the active cell. For clarity, the remaining translations of GET.PIVOT.TABLE assume that pivot\_table\_name is omitted.

**GET.PIVOT.TABLE**(1)

ActiveCell.PivotTable.RefreshName

**GET.PIVOT.TABLE**(2)

ActiveCell.PivotTable.RefreshDate

### **GET.PIVOT.TABLE(3)**

```
Dim result
Set flds = ActiveCell.PivotTable.PivotFields
If flds.Count = 0 Then
 result = CVErr(xlErrNA)
Else
 ReDim result(1 To flds.Count)
 x = 1
 For Each f In flds
 result(x) = f.Name
 x = x + 1
 Next
End If
```

Returns an array of pivot field names for all fields of the pivot table. Visual Basic has no direct equivalent; however, the PivotFields method returns a collection which can be iterated to examine the fields. The example builds an equivalent array by iterating the collection.

Replace the PivotFields method with either VisibleFields, HiddenFields, RowFields, ColumnFields, PageFields, or DataFields to make an equivalent for GET.PIVOT.TABLE() type\_num values of 5, 6, 7, 8, 9, and 10 respectively.

### **GET.PIVOT.TABLE(4)**

```
ActiveCell.PivotTable.PivotFields.Count
```

### **GET.PIVOT.TABLE(5)**

Returns an array of pivot field names for all visible fields of the pivot table. Visual Basic has no direct equivalent; however, the VisibleFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the VisibleFields method.

### **GET.PIVOT.TABLE(6)**

Returns an array of pivot field names for all hidden fields of the pivot table. Visual Basic has no direct equivalent; however, the HiddenFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the HiddenFields method.

### **GET.PIVOT.TABLE(7)**

Returns an array of pivot field names for all row fields of the pivot table. Visual Basic has no direct equivalent; however, the RowFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the RowFields method.

### **GET.PIVOT.TABLE(8)**

Returns an array of pivot field names for all column fields of the pivot table. Visual Basic has no direct equivalent; however, the ColumnFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the ColumnFields method.

### **GET.PIVOT.TABLE(9)**

Returns an array of pivot field names for all page fields of the pivot table. Visual Basic has no direct equivalent; however, the PageFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the PageFields method.

### **GET.PIVOT.TABLE(10)**

Returns an array of pivot field names for all data fields of the pivot table. Visual Basic has no direct equivalent; however, the DataFields method returns a collection which can be iterated to examine the fields. See the example for GET.PIVOT.TABLE(3) and use the DataFields method.

|                                              |                                                                                                                                                                                                                                                                               |
|----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>GET.PIVOT.TABLE</u></b> (11)           | ActiveCell.PivotTable.TableRange1                                                                                                                                                                                                                                             |
| <b><u>GET.PIVOT.TABLE</u></b> (12)           | ActiveCell.PivotTable.TableRange2                                                                                                                                                                                                                                             |
| <b><u>GET.PIVOT.TABLE</u></b> (13)           | ActiveCell.PivotTable.RowRange                                                                                                                                                                                                                                                |
| <b><u>GET.PIVOT.TABLE</u></b> (14)           | ActiveCell.PivotTable.ColumnRange                                                                                                                                                                                                                                             |
| <b><u>GET.PIVOT.TABLE</u></b> (15)           | ActiveCell.PivotTable.DataLabelRange                                                                                                                                                                                                                                          |
| <b><u>GET.PIVOT.TABLE</u></b> (16)           | ActiveCell.PivotTable.PageRange                                                                                                                                                                                                                                               |
| <b><u>GET.PIVOT.TABLE</u></b> (17)           | ActiveCell.PivotTable.DataBodyRange                                                                                                                                                                                                                                           |
| <b><u>GET.PIVOT.TABLE</u></b> (18)           | ActiveCell.PivotTable.RowGrand                                                                                                                                                                                                                                                |
| <b><u>GET.PIVOT.TABLE</u></b> (19)           | ActiveCell.PivotTable.ColumnGrand                                                                                                                                                                                                                                             |
| <b><u>GET.PIVOT.TABLE</u></b> (20)           | ActiveCell.PivotTable.SaveData                                                                                                                                                                                                                                                |
| <b><u>GET.PIVOT.TABLE</u></b> (21)           | ActiveCell.PivotTable.HasAutoFormat                                                                                                                                                                                                                                           |
| <b><u>GET.PIVOT.TABLE</u></b> (22)           | ActiveCell.PivotTable.SourceData                                                                                                                                                                                                                                              |
| <b><u>GET.TOOL</u></b> (1, bar_id, position) | Toolbars(bar_id).ToolbarButtons(position).Id                                                                                                                                                                                                                                  |
| <b><u>GET.TOOL</u></b> (2, bar_id, position) | Toolbars(bar_id).ToolbarButtons(position).OnAction                                                                                                                                                                                                                            |
| <b><u>GET.TOOL</u></b> (3, bar_id, position) | Toolbars(bar_id).ToolbarButtons(position).Pushed                                                                                                                                                                                                                              |
| <b><u>GET.TOOL</u></b> (4, bar_id, position) | Toolbars(bar_id).ToolbarButtons(position).Enabled                                                                                                                                                                                                                             |
| <b><u>GET.TOOL</u></b> (5, bar_id, position) | Toolbars(bar_id).ToolbarButtons(position).BuiltInFace                                                                                                                                                                                                                         |
| <b><u>GET.TOOL</u></b> (6, bar_id, position) | ExecuteExcel4Macro("GET.TOOL(6, bar_id, position)")<br>Returns the help text associated with the toolbar button. Visual Basic has no direct equivalent. Instead, help text is associated with the macro assigned to the toolbar button.                                       |
| <b><u>GET.TOOL</u></b> (7, bar_id, position) | ExecuteExcel4Macro("GET.TOOL(7, bar_id, position)")<br>Returns the ballon help associated with the toolbar button. Visual Basic has no equivalent.                                                                                                                            |
| <b><u>GET.TOOLBAR</u></b> (1, bar_id)        | Dim result()<br>ReDim result(1 To Toolbars(bar_id).ToolbarButtons.Count<br>x = 1<br>For Each t In Toolbars(bar_id).ToolbarButtons<br>result(x) = t.Id<br>x = x + 1<br>Next<br>Returns an array of ID numbers corresponding to the buttons on the toolbar. Visual Basic has no |

direct equivalent; however, the Toolbars collection can be iterated to examine toolbars. The example builds an equivalent array by iterating the collection.

**GET.TOOLBAR**(2, bar\_id)                      Toolbars(bar\_id).Left

**GET.TOOLBAR**(3, bar\_id)                      Toolbars(bar\_id).Top

**GET.TOOLBAR**(4, bar\_id)                      Toolbars(bar\_id).Width

**GET.TOOLBAR**(5, bar\_id)                      Toolbars(bar\_id).Height

**GET.TOOLBAR**(6, bar\_id)                      Toolbars(bar\_id).Position

**GET.TOOLBAR**(7, bar\_id)                      Toolbars(bar\_id).Visible

**GET.TOOLBAR**(8)                      Dim result()  
ReDim result(1 To Toolbars.Count)  
x = 1  
For Each t In Toolbars  
    result(x) = t.Name  
    x = x + 1  
Next

Returns an array of toolbar names for all toolbars visible and hidden. Visual Basic has no direct equivalent; however, the Toolbars collection can be iterated to examine toolbars. The example builds an equivalent array by iterating the collection.

**GET.TOOLBAR**(9)                      Dim result()  
ReDim result(1 To Toolbars.Count)  
x = 1  
For Each t In Toolbars  
    If t.Visible Then  
        result(x) = t.Name  
        x = x + 1  
    End If  
Next  
ReDim result(1 To (x - 1))

Returns an array of toolbar names for all visible toolbars. Visual Basic has no direct equivalent; however, the Toolbars collection can be iterated to examine toolbars. The example builds an equivalent array by iterating the collection and filtering for visible toolbars.

**GET.TOOLBAR**(10, bar\_id)                      ExecuteExcel4Macro("GET.TOOLBAR(10, bar\_id)")  
Returns whether a toolbar is visible in full screen mode. Visual Basic has no direct equivalent.

**GET.WINDOW**(type\_num, window\_text)                      Windows(window\_text).Property  
Returns information about a window. If window\_text is specified, use the appropriate Visual Basic method or property directly on an object returned by the Windows method. If window\_text is omitted, returns information about the active window. For clarity, the remaining translations of GET.WINDOW assume that window\_text is omitted.

**GET.WINDOW**(1)                      Set containingBook = ActiveWindow.ActiveSheet.Parent  
If containingBook.Sheets.Count = 1 Then  
    ' Get bookname, strip off extension (if any)

```

bookName = containingBook.Name
For i = Len(bookName) To 1 Step -1
 If Mid(bookName, i, 1) = "." Then
 bookName = Mid(bookName, 1, i - 1)
 Exit For
 End If
Next
x = StrComp(bookName, _
 ActiveWindow.ActiveSheet.Name, 1)
If x <> 0 Then Goto BuildLongName
result = containingBook.Name
Else
BuildLongName:
 result = "[" & containingBook.Name & _
 "]" & ActiveWindow.ActiveSheet.Name
End If

```

Returns the name of the workbook and sheet in the window. The example constructs an exact equivalent of GET.WINDOW(1), which has special rules to provide backwards compatability with Microsoft Excel 4.0. In Visual Basic it is not necessary to construct such an equivalent because the Worksheet and Workbook objects are used instead of a text name.

|                               |                                    |
|-------------------------------|------------------------------------|
| <b><u>GET.WINDOW</u></b> (2)  | ActiveWindow.WindowNumber          |
| <b><u>GET.WINDOW</u></b> (3)  | ActiveWindow.Left                  |
| <b><u>GET.WINDOW</u></b> (4)  | ActiveWindow.Top                   |
| <b><u>GET.WINDOW</u></b> (5)  | ActiveWindow.Width                 |
| <b><u>GET.WINDOW</u></b> (6)  | ActiveWindow.Height                |
| <b><u>GET.WINDOW</u></b> (7)  | ActiveWindow.Visible               |
| <b><u>GET.WINDOW</u></b> (8)  | ActiveWindow.DisplayFormulas       |
| <b><u>GET.WINDOW</u></b> (9)  | ActiveWindow.DisplayGridlines      |
| <b><u>GET.WINDOW</u></b> (10) | ActiveWindow.DisplayHeadings       |
| <b><u>GET.WINDOW</u></b> (11) | ActiveWindow.DisplayZeros          |
| <b><u>GET.WINDOW</u></b> (12) | ActiveWindow.GridlineColorIndex    |
| <b><u>GET.WINDOW</u></b> (13) | ActiveWindow.Panes(1).ScrollColumn |

Returns an array containing the leftmost column numbers for all panes. Visual Basic has no direct equivalent. Use the Panes collection to examine the properties of a particular pane. If required, construct an array of result values by iterating all panes in the collection.

|                               |                                 |
|-------------------------------|---------------------------------|
| <b><u>GET.WINDOW</u></b> (14) | ActiveWindow.Panes(1).ScrollRow |
|-------------------------------|---------------------------------|

Returns an array containing the topmost row numbers for all panes. Visual Basic has no direct equivalent. Use the Panes collection to examine the properties of a particular pane. If required,

construct an array of result values by iterating all panes in the collection.

|                                                                                                                                                                                                                                                                 |                                                       |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <b><u>GET.WINDOW</u></b> (15)                                                                                                                                                                                                                                   | ExecuteExcel4Macro("INDEX(GET.WINDOW(15),,pane_num)") |
| No direct equivalent. The VisibleRange property returns a range containing all visible cells, even if the cell is only partially visible. This range could be converted into the fractional width values returned by GET.WINDOW(15), or use ExecuteExcel4Macro. |                                                       |

|                                                                                                                                                                                                                                                                       |                                                       |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------|
| <b><u>GET.WINDOW</u></b> (16)                                                                                                                                                                                                                                         | ExecuteExcel4Macro("INDEX(GET.WINDOW(16),,pane_num)") |
| <p>No direct equivalent. The VisibleRange property returns a range containing all visible cells, even if the cell is only partially visible. This range can be converted into the fractional height values returned by GET.WINDOW(16), or use ExecuteExcel4Macro.</p> |                                                       |

```
GET.WINDOW(17)
result = ActiveForm.ActivePane.Index
If ActiveForm.Panes.Count = 2 Then
 If ActiveForm.SplitVertical <> 0 Then
 If result = 2 Then result = 3
 End If
End If
```

Returns the index of the pane. In Visual Basic the panes are renumber depending on the split configuration, so conversion is necessary to return an exact equivalent.

**GET.WINDOW**(18)      ActiveWindow.SplitVertical <> 0

**GET.WINDOW**(19)      ActiveWindow.SplitHorizontal <> 0

**GET.WINDOW**(20)      ActiveWindow.WindowState = xlMaximized

**GET.WINDOW**(22) ActiveWindow.DisplayOutline

**GET.WINDOW**(23)      *ActiveWindow.WindowState*

**GET.WINDOW**(24)      ActiveWindow.FreezePanes

**GET.WINDOW**(25)      ActiveWindow.Zoom

**GET.WINDOW**(26)      ActiveWindow.DisplayHorizontalScrollBars

**GET.WINDOW**(27)      ActiveWindow.DisplayVerticalScrollBars

**GET.WINDOW**(28)      ActiveWindow.TabRatio

**GET.WINDOW**(29)      ActiveWindow.DisplayWorkbookTabs

GET.WINDOW(30) "I" & ActiveWindow.ActiveSheet.Parent.Name & \_  
"J" & ActiveWindow.ActiveSheet.Name

**GET.WINDOW**(31)      ActiveWindow.ActiveSheet.Parent.Name

**GET.WORKBOOK**(type num, name text)      Workbooks(name text).Property



Returns information about a workbook. If name\_text is specified, use the appropriate Visual Basic method or property directly on a workbook object returned by the Workbooks method. If name\_text is omitted, returns information on the active workbook. For clarity, the remaining translations of GET.WORKBOOK assume that name\_text is omitted.

**GET.WORKBOOK(1)**

```
Dim result()
ReDim result(1 To ActiveWorkbook.Sheets.Count)
bkName = "[" & ActiveWorkbook.Name & "]"
x = 1
For Each s In ActiveWorkbook.Sheets
 result(x) = bkName & s.Name
 x = x + 1
Next
```

Returns an array of sheet names for all sheets in the workbook. Visual Basic has no direct equivalent; however, the Sheets collection can be iterated to examine sheets. The example builds an equivalent array by iterating the collection.

**GET.WORKBOOK(2)**

CVErr(xlErrNA)

In Microsoft Excel 5.0 this function always returns the #N/A error value.

**GET.WORKBOOK(3)**

```
Dim result()
ReDim result(1 To ActiveWindow.SelectedSheets.Count)
bkName = "[" & ActiveWorkbook.Name & "]"
x = 1
For Each s In ActiveWindow.SelectedSheets
 result(x) = bkName & s.Name
 x = x + 1
Next
```

Returns an array of sheet names for the selected sheets in the workbook. Visual Basic has no direct equivalent; however, the SelectedSheets method returns a collection which can be iterated to examine the selected sheets. The example builds an equivalent array by iterating the collection.

**GET.WORKBOOK(4)**

ActiveWorkbook.Sheets.Count

**GET.WORKBOOK(5)**

ActiveWorkbook.HasRoutingSlip

**GET.WORKBOOK(6)**

ActiveWorkbook.RoutingSlip.Recipients

**GET.WORKBOOK(7)**

ActiveWorkbook.RoutingSlip.Subject

**GET.WORKBOOK(8)**

ActiveWorkbook.RoutingSlip.Message

**GET.WORKBOOK(9)**

ActiveWorkbook.RoutingSlip.Delivery

**GET.WORKBOOK(10)**

ActiveWorkbook.RoutingSlip.ReturnWhenDone

**GET.WORKBOOK(11)**

ActiveWorkbook.Routed

**GET.WORKBOOK(12)**

ActiveWorkbook.RoutingSlip.TrackStatus

**GET.WORKBOOK(13)**

ActiveWorkbook.RoutingSlip.Status

|                                 |                                     |
|---------------------------------|-------------------------------------|
| <b><u>GET.WORKBOOK</u></b> (14) | ActiveWorkbook.ProtectStructure     |
| <b><u>GET.WORKBOOK</u></b> (15) | ActiveWorkbook.ProtectWindows       |
| <b><u>GET.WORKBOOK</u></b> (16) | ActiveWorkbook.Name                 |
| <b><u>GET.WORKBOOK</u></b> (17) | ActiveWorkbook.ReadOnly             |
| <b><u>GET.WORKBOOK</u></b> (18) | ActiveWorkbook.WriteReserved        |
| <b><u>GET.WORKBOOK</u></b> (19) | ActiveWorkbook.WriteReservedBy      |
| <b><u>GET.WORKBOOK</u></b> (20) | ActiveWorkbook.FileFormat           |
| <b><u>GET.WORKBOOK</u></b> (21) | ActiveWorkbook.CreateBackup         |
| <b><u>GET.WORKBOOK</u></b> (22) | ActiveWorkbook.SaveLinkValues       |
| <b><u>GET.WORKBOOK</u></b> (23) | ActiveWorkbook.HasMailer            |
| <b><u>GET.WORKBOOK</u></b> (24) | Not ActiveWorkbook.Saved            |
| <b><u>GET.WORKBOOK</u></b> (25) | ActiveWorkbook.Mailer.ToRecipients  |
| <b><u>GET.WORKBOOK</u></b> (26) | ActiveWorkbook.Mailer.CCRecipients  |
| <b><u>GET.WORKBOOK</u></b> (27) | ActiveWorkbook.Mailer.BCCRecipients |
| <b><u>GET.WORKBOOK</u></b> (28) | ActiveWorkbook.Mailer.Subject       |
| <b><u>GET.WORKBOOK</u></b> (29) | ActiveWorkbook.Mailer.Enclosures    |
| <b><u>GET.WORKBOOK</u></b> (30) | ActiveWorkbook.Mailer.Received      |
| <b><u>GET.WORKBOOK</u></b> (31) | ActiveWorkbook.Mailer.SendDateTime  |
| <b><u>GET.WORKBOOK</u></b> (32) | ActiveWorkbook.Mailer.Sender        |
| <b><u>GET.WORKBOOK</u></b> (33) | ActiveWorkbook.Title                |
| <b><u>GET.WORKBOOK</u></b> (34) | ActiveWorkbook.Subject              |
| <b><u>GET.WORKBOOK</u></b> (35) | ActiveWorkbook.Author               |
| <b><u>GET.WORKBOOK</u></b> (36) | ActiveWorkbook.Keywords             |
| <b><u>GET.WORKBOOK</u></b> (37) | ActiveWorkbook.Comments             |

|                                  |                                                      |
|----------------------------------|------------------------------------------------------|
| <b><u>GET.WORKBOOK</u></b> (38)  | ActiveWorkbook.ActiveSheet.Name                      |
| <b><u>GET.WORKSPACE</u></b> (1)  | Application.OperatingSystem                          |
| <b><u>GET.WORKSPACE</u></b> (2)  | Application.Version                                  |
| <b><u>GET.WORKSPACE</u></b> (3)  | Application.FixedDecimals                            |
| <b><u>GET.WORKSPACE</u></b> (4)  | Application.ReferenceStyle = xlR1C1                  |
| <b><u>GET.WORKSPACE</u></b> (5)  | Application.DisplayScrollBars                        |
| <b><u>GET.WORKSPACE</u></b> (6)  | Application.DisplayStatusBar                         |
| <b><u>GET.WORKSPACE</u></b> (7)  | Application.DisplayFormulaBar                        |
| <b><u>GET.WORKSPACE</u></b> (8)  | Not Application.IgnoreRemoteRequests                 |
| <b><u>GET.WORKSPACE</u></b> (9)  | Application.TransitionMenuKey                        |
| <b><u>GET.WORKSPACE</u></b> (10) | Application.DataEntryMode<br>Application.CutCopyMode |

Returns a number indicating the current user mode. Use either of the listed properties. Microsoft Excel 5.0 has no equivalent for Data Find mode.

|                                  |                                                                                                                                                        |
|----------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>GET.WORKSPACE</u></b> (11) | Application.Left                                                                                                                                       |
| <b><u>GET.WORKSPACE</u></b> (12) | Application.Top                                                                                                                                        |
| <b><u>GET.WORKSPACE</u></b> (13) | Application.UsableWidth                                                                                                                                |
| <b><u>GET.WORKSPACE</u></b> (14) | Application.UsableHeight                                                                                                                               |
| <b><u>GET.WORKSPACE</u></b> (15) | Application.WindowState                                                                                                                                |
| <b><u>GET.WORKSPACE</u></b> (16) | Application.MemoryFree                                                                                                                                 |
| <b><u>GET.WORKSPACE</u></b> (17) | Application.MemoryTotal                                                                                                                                |
| <b><u>GET.WORKSPACE</u></b> (18) | Application.MathCoprocessorAvailable                                                                                                                   |
| <b><u>GET.WORKSPACE</u></b> (19) | Application.MouseAvailable                                                                                                                             |
| <b><u>GET.WORKSPACE</u></b> (20) | Dim result<br>If ActiveWindow.SelectedSheets.Count = 1 Then<br>result = CVErr(xlErrNA)<br>Else<br>ReDim result(1 To ActiveWindow.SelectedSheets.Count) |

```

bkName = "[" & ActiveWorkbook.Name & "]"
x = 1
For Each s In ActiveWindow.SelectedSheets
 result(x) = bkName & s.Name
 x = x + 1
Next
End If

```

Returns an array containing the sheet names of the group edit. In Microsoft Excel 5.0, groups are created by selecting multiple sheet tabs. The example constructs an array using the collection of sheets returned by the SelectedSheets method.

|                                                                                                                                       |                                                     |
|---------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|
| <b><u>GET.WORKSPACE</u></b> (21)                                                                                                      | Toolbars("Standard").Visible                        |
| <b><u>GET.WORKSPACE</u></b> (22)                                                                                                      | Application.DDEAppReturnCode                        |
| <b><u>GET.WORKSPACE</u></b> (type_num)                                                                                                | Application.StartupPath                             |
| <b><u>GET.WORKSPACE</u></b> (24)                                                                                                      | Application.AltStartupPath                          |
| <b><u>GET.WORKSPACE</u></b> (25)                                                                                                      | Application.RecordRelative                          |
| <b><u>GET.WORKSPACE</u></b> (26)                                                                                                      | Application.UserName                                |
| <b><u>GET.WORKSPACE</u></b> (27)                                                                                                      | Application.OrganizationName                        |
| <b><u>GET.WORKSPACE</u></b> (28)                                                                                                      | Application.TransitionMenuKeyAction                 |
| <b><u>GET.WORKSPACE</u></b> (29)                                                                                                      | Application.TransitionNavigKeys                     |
| <b><u>GET.WORKSPACE</u></b> (30)                                                                                                      | ExecuteExcel4Macro("INDEX(GET.WORKSPACE(30),,num)") |
| No direct equivalent. Since ExecuteExcel4Macro cannot return an array, use the INDEX function to retrieve each setting one at a time. |                                                     |
| <b><u>GET.WORKSPACE</u></b> (31)                                                                                                      | No equivalent in Visual Basic.                      |
| <b><u>GET.WORKSPACE</u></b> (32)                                                                                                      | Application.Path                                    |
| <b><u>GET.WORKSPACE</u></b> (33)                                                                                                      | No direct equivalent.                               |
| <b><u>GET.WORKSPACE</u></b> (34)                                                                                                      | No direct equivalent.                               |
| <b><u>GET.WORKSPACE</u></b> (35)                                                                                                      | No equivalent in Visual Basic.                      |
| <b><u>GET.WORKSPACE</u></b> (36)                                                                                                      | Application.CellDragAndDrop                         |

|                                                                                                                                                                                                                                                                                                                             |                                         |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|
| <b><u>GET.WORKSPACE</u></b> (37)                                                                                                                                                                                                                                                                                            | Application.International               |
| <b><u>GET.WORKSPACE</u></b> (38)                                                                                                                                                                                                                                                                                            | Application.DisplayAlerts               |
| No direct equivalent. The DisplayAlerts property indicates whether alerts are turned on or off, but does not indicate if error trapping is completely disabled. The status of Visual Basic error trapping, as set by the On Error statement, is not available.                                                              |                                         |
| <b><u>GET.WORKSPACE</u></b> (39)                                                                                                                                                                                                                                                                                            |                                         |
| No equivalent in Visual Basic. The location of the currently active error handler set by the On Error statement is not available.                                                                                                                                                                                           |                                         |
| <b><u>GET.WORKSPACE</u></b> (40)                                                                                                                                                                                                                                                                                            | Application.ScreenUpdating              |
| <b><u>GET.WORKSPACE</u></b> (41)                                                                                                                                                                                                                                                                                            | Application.PreviousSelections          |
| <b><u>GET.WORKSPACE</u></b> (42)                                                                                                                                                                                                                                                                                            | Application.CanPlaySounds               |
| <b><u>GET.WORKSPACE</u></b> (43)                                                                                                                                                                                                                                                                                            | Application.CanRecordSounds             |
| <b><u>GET.WORKSPACE</u></b> (44)                                                                                                                                                                                                                                                                                            | Application.RegisteredFunctions         |
| Returns an array of information on DLL or code resource functions registered for use by worksheets and macro sheets. Note that these functions are not directly callable by Visual Basic and are not related to the DLL and code resources declared within Visual Basic by the Declare Function and Declare Sub statements. |                                         |
| <b><u>GET.WORKSPACE</u></b> (45)                                                                                                                                                                                                                                                                                            | Application.WindowsForPens              |
| <b><u>GET.WORKSPACE</u></b> (46)                                                                                                                                                                                                                                                                                            | Application.MoveAfterReturn             |
| <b><u>GET.WORKSPACE</u></b> (48)                                                                                                                                                                                                                                                                                            | Application.LibraryPath                 |
| <b><u>GET.WORKSPACE</u></b> (49)                                                                                                                                                                                                                                                                                            | Application.MailSession                 |
| <b><u>GET.WORKSPACE</u></b> (50)                                                                                                                                                                                                                                                                                            | Application.DisplayFullScreen           |
| <b><u>GET.WORKSPACE</u></b> (51)                                                                                                                                                                                                                                                                                            | ExecuteExcel4Macro("GET.WORKSPACE(51)") |
| No direct equivalent.                                                                                                                                                                                                                                                                                                       |                                         |
| <b><u>GET.WORKSPACE</u></b> (52)                                                                                                                                                                                                                                                                                            | ExecuteExcel4Macro("GET.WORKSPACE(51)") |
| No direct equivalent.                                                                                                                                                                                                                                                                                                       |                                         |
| <b><u>GET.WORKSPACE</u></b> (53)                                                                                                                                                                                                                                                                                            | Application.ActiveDialog.Name           |
| <b><u>GET.WORKSPACE</u></b> (54)                                                                                                                                                                                                                                                                                            | Application.EditDirectlyInCell          |
| <b><u>GET.WORKSPACE</u></b> (55)                                                                                                                                                                                                                                                                                            | Application.AlertBeforeOverwriting      |
| <b><u>GET.WORKSPACE</u></b> (56)                                                                                                                                                                                                                                                                                            | Application.StandardFont                |

|                                                                                                                                                                            |                                                                                                                                                 |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>GET.WORKSPACE</u></b> (57)                                                                                                                                           | Application.StandardFontSize                                                                                                                    |
| <b><u>GET.WORKSPACE</u></b> (58)                                                                                                                                           | Application.DisplayRecentFiles                                                                                                                  |
| <b><u>GET.WORKSPACE</u></b> (59)                                                                                                                                           | Application.DisplayExcel4Menus                                                                                                                  |
| <b><u>GET.WORKSPACE</u></b> (60)                                                                                                                                           | Application.EnableTipWizard                                                                                                                     |
| <b><u>GET.WORKSPACE</u></b> (61)                                                                                                                                           | Application.CustomListCount                                                                                                                     |
| <b><u>GET.WORKSPACE</u></b> (62)                                                                                                                                           | Application.FileConverters                                                                                                                      |
| <b><u>GET.WORKSPACE</u></b> (63)                                                                                                                                           | Application.MailSystem                                                                                                                          |
| <b><u>GET.WORKSPACE</u></b> (64)                                                                                                                                           | Application.AskToUpdateLinks                                                                                                                    |
| <b><u>GET.WORKSPACE</u></b> (65)                                                                                                                                           | Application.CopyObjectsWithCells                                                                                                                |
| <b><u>GET.WORKSPACE</u></b> (66)                                                                                                                                           | Application.SheetsInNewWorkbook                                                                                                                 |
| <b><u>GET.WORKSPACE</u></b> (67)                                                                                                                                           | Application.DefaultFilePath                                                                                                                     |
| <b><u>GET.WORKSPACE</u></b> (68)                                                                                                                                           | Application.ShowToolTips                                                                                                                        |
| <b><u>GET.WORKSPACE</u></b> (69)                                                                                                                                           | Application.LargeButtons                                                                                                                        |
| <b><u>GET.WORKSPACE</u></b> (70)                                                                                                                                           | Application.PromptForSummaryInfo                                                                                                                |
| <b><u>GET.WORKSPACE</u></b> (71)                                                                                                                                           | ExecuteExcel4Macro("GET.WORKSPACE(71)")                                                                                                         |
| <b><u>GET.WORKSPACE</u></b> (72)                                                                                                                                           | Application.ColorButtons                                                                                                                        |
| <b><u>GOAL.SEEK</u></b> (target_cell, target_value, variable_cell)                                                                                                         | Range( <i>target_cell</i> ).GoalSeek goal := target_value, _<br>changingCell := <i>variable_cell</i>                                            |
| <b><u>GOTO</u></b> (reference)                                                                                                                                             | GoTo TargetLabel<br><br>' code that is skipped here<br><br>TargetLabel:<br>' execution resumes here, following the label                        |
| Use the Visual Basic GoTo statement to branch unconditionally. Unlike the GOTO() function, Visual Basic requires hardcoded label values that cannot be changed at runtime. |                                                                                                                                                 |
| <b><u>GRIDLINES</u></b> (x_major, x_minor, y_major, y_minor)                                                                                                               | With ActiveChart.Axes(xlCategory)<br>.HasMajorGridlines = x_major<br>.HasMinorGridlines = x_minor<br>End With<br>With ActiveChart.Axes(xlValue) |

```
.HasMajorGridlines = y_major
.HasMinorGridlines = y_minor
End With
```

Controls gridline settings for 2D charts.

**GRIDLINES**(x\_major, x\_minor, y\_major,  
y\_minor, z\_major, z\_minor, 2D\_effect)

```
With ActiveChart.Axes(xlCategory)
.HasMajorGridlines = x_major
.HasMinorGridlines = x_minor
End With
With ActiveChart.Axes(xlSeries)
.HasMajorGridlines = y_major
.HasMinorGridlines = y_minor
End With
With ActiveChart.Axes(xlValue)
.HasMajorGridlines = z_major
.HasMinorGridlines = z_minor
End With
ActiveChart.WallsAndGridlines2D = 2D_effect
```

Controls gridline settings for 3D charts.

**GROUP**( )

Groups the selected drawing objects.

```
Selection.Group.Select
```

.....



The `cancel_close` argument has no equivalent.

In Microsoft Excel 4.0 Macros, the help reference is specified as a string containing the help file name followed by the help context number, for example "PROJECT.HLP!255". In Visual Basic the help file and the help context are specified separately instead of in a single string. The example divides the help\_ref string argument into its string and numeric components.

Hides or unhides the selected drawing object.

Hides or unhides a group of specific drawing objects that are not selected. Use ActiveChart instead of



ActiveSheet if the drawing objects are on a chart.

**HLINE**(num\_columns)      ActiveWindow.SmallScroll toRight := num\_columns

**HPAGE**(num\_windows)      ActiveWindow.LargeScroll toRight := num\_windows

|                                                   |                                      |
|---------------------------------------------------|--------------------------------------|
| <b><u>HSCROLL</u></b> (position, TRUE)            | ActiveWindow.ScrollColumn = position |
| Scrolls horizontally to a specific column number. |                                      |

|                                                                                                                                                                              |                                            |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
| <b><u>HSCROLL</u></b> (position, FALSE)                                                                                                                                      | ActiveWindow.ScrollColumn = position * 256 |
| <p>Scrolls horizontally to a fractional position. In Visual Basic you can scroll to specific columns only. The example converts a fraction to an absolute column number.</p> |                                            |

```
IF(logical_test) If logical_test Then
 ' code for True condition
 Else
 ' code for False condition
 End If
```

Evaluates the logical\_test expression and, based on the results, runs code between the IF and ELSE, ELSE.IF, or END.IF functions. The Visual Basic examples illustrates the complete If statement syntax.

|                                                            |                                                                                             |
|------------------------------------------------------------|---------------------------------------------------------------------------------------------|
| <u>IF</u> (logical_test, value_if_true,<br>value_if_false) | If logical_test Then<br>result = value_if_true<br>Else<br>result = value_if_false<br>End If |
|------------------------------------------------------------|---------------------------------------------------------------------------------------------|

Evaluates the logical\_test expression and, depending on the test value, returns the value of either the true\_value or the false\_value expressions. Visual Basic has no direct equivalent; use a temporary variable instead.

**INITIATE**(app\_text, topic\_text)      Application.DDEInitiate app := app\_text, topic := topic\_text

**INPUT**(message\_text, type\_num, title\_text, default, x\_pos, y\_pos, help\_ref)

Application.InputBox prompt := message\_text, title := title\_text,  
 — default := default, left := x\_pos, top := y\_pos, \_  
 helpFile := *help\_ref*, helpContextID := *help\_file*, \_  
 type := type\_num

Presents a dialog box with an edit field to get user input. The `help_ref` argument is split into a separate help file name and help context number in Visual Basic. See the `HELP` function for an example of how to convert the `help_ref` argument into the two components. The `InputBox` function in the Visual Basic runtime library provides similar functionality but does not support the `type_num` argument. If you plan to run your code in other Visual Basic environments, use the `InputBox` function.

INSERT(1) Selection.Insert shift := xlToRight

**INSERT**(2)                      Selection.Insert shift := xlDown

**INSERT(3)** Selection.EntireRow.Insert

## **INSERT**(4)

**INSERT.OBJECT**(object\_class, file\_name, link\_logical, display\_icon\_logical, icon\_file, icon\_number, icon\_label)

Selection.EntireColumn.Insert

```
ActiveSheet.OLEObjects.Add classType := object_class, _
 fileName := file_name, link := link_logical, _
 displayAsIcon := display_icon_logical, _
 iconFileName := icon_file, iconIndex := icon_number, _
 iconLabel := icon_label
```

**INSERT.PICTURE**(file\_name, filter\_number)

```
ActiveSheet.Pictures.Insert filename := file_name, _
 converter := filter_number
```

**INSERT.TITLE**(chart, y\_primary, x\_primary, y\_secondary, x\_secondary)

```
With ActiveChart
 .HasTitle = chart
 .Axes(xlCategory, xlPrimary).HasTitle = x_primary
 .Axes(xlCategory, xlSecondary).HasTitle = x_secondary
 .Axes(xlValue, xlPrimary).HasTitle = y_primary
 .Axes(xlValue, xlSecondary).HasTitle = y_secondary
End With
```

Attaches text to a 2-D chart.

**INSERT.TITLE**(chart, z\_primary, x\_primary, y\_primary)

```
With ActiveChart
 .HasTitle = chart
 .Axes(xlValue).HasTitle = chart, z_primary, x_primary
 .Axes(xlCategory).HasTitle = x_primary
 .Axes(xlSeries).HasTitle = x_primary, y_primary
End With
```

Attaches text to a 3-D chart.

## **J**

### **JUSTIFY**()

Selection.Justify

## **K**

## Visual Basic Equivalents for Macro Functions and Commands

.....



### L

**LABEL.PROPERTIES**(accel\_text,  
accel2\_text, 3d\_shading)

With Selection  
    .Accelerator = accel\_text  
    .PhoneticAccelerator = accel2\_text  
    .Display3DShading = 3d\_shading  
End With

**LAST.ERROR**( )

Erl

No direct equivalent. Use the Erl function to return the line number of the Visual Basic expression that caused the last Visual Basic error.

**LEGEND**(logical)

ActiveChart.HasLegend = logical

**LINE.PRINT**( )

ExecuteExcel4Macro("LINE.PRINT(...)")

No direct equivalent. Use either ExecuteExcel4Macro or a combination of the Visual Basic Write and Print statements to create output files.

**LINK.COMBO**(link\_logical)

Selection.LinkCombo link := link\_logical

**LINK.FORMAT**( )

Selection.TickLabels.NumberFormatLinked = True

Use this syntax if axis tick labels are selected.

**LINK.FORMAT**( )

Selection.NumberFormatLinked = True

Use this syntax if objects other than axis tick labels are selected.

**LINKS**(document\_text, type\_num)

Workbooks(document\_text).LinkSources type := *type\_num*

Use ActiveWorkbook instead of Workbooks(document\_text) if the document\_text argument is omitted.

## **LIST.NAMES**( )

Selection.ListNames

**LISTBOX.PROPERTIES**(range, link,  
drop\_size, multi\_select, 3d\_shading)

With Selection  
.ListFillRange = range  
.LinkedCell = link  
.DropDownLines = drop\_size  
.MultiSelect = *multi\_select*  
.Display3DShading = 3d\_shading  
End With

## **M**

### **MAIL.ADD.MAILER**( )

ActiveWorkbook.HasMailer = True

### **MAIL.DELETE.MAILER**( )

ActiveWorkbook.HasMailer = False

**MAIL.EDIT.MAILER**(to\_recipients,  
cc\_recipients, bcc\_recipients, subject,  
enclosures, which\_address)

With ActiveWorkbook.Mailer  
.ToRecipients = *to\_recipients*  
.CCRecipients = *bcc\_recipients*  
.BCCRecipients = *bcc\_recipients*  
.Subject = subject  
.Enclosures = *enclosures*  
End With

The which\_address argument has no direct equivalent in Visual Basic.

### **MAIL.FORWARD**( )

ActiveWorkbook.ForwardMailer

### **MAIL.LOGOFF**( )

Application.MailLogoff

**MAIL.LOGON**(name\_text, password\_text,  
download\_logical)

Application.MailLogon name := name\_text, \_  
password := password\_text, \_  
downloadNewMail := download\_logical

### **MAIL.NEXT.LETTER**( )

Application.NextLetter

### **MAIL.REPLY**( )

ActiveWorkbook.Reply

### **MAIL.REPLY.ALL**( )

ActiveWorkbook.ReplyAll

**MAIN.CHART**(type\_num, stack, 100, vary,  
overlap, drop, hilo, overlap%, cluster,  
angle)

With ActiveChart  
.Type = *type\_num*  
.Subtype = 100  
End With  
With ActiveChart.ChartGroups(1)  
.VaryByCategories = vary  
.Overlap = overlap%  
.HasDropLines = drop  
.HasHiLoLines = hilo  
.FirstSliceAngle = angle  
End With

This function is used only for backwards compatability with Microsoft Excel 2.2 or earlier. Some

arguments do not apply to all chart types, and the stack, overlap, and cluster arguments have no direct equivalent in Microsoft Excel 5.0.

**MAIN.CHART.TYPE**(type\_num)

ActiveChart.Type = *type\_num*

**MENU.EDITOR**()

Application.Dialogs(xlDialogMenuEditor).Show

**MERGE.STYLES**(document\_text)

ActiveWorkbook.Styles.Merge \_  
workbook := Workbooks(document\_text)

**MESSAGE**(TRUE, text)

Application.StatusBar = text

Displays text in the status bar.

**MESSAGE**(FALSE)

Application.StatusBar = False

Restores the status bar to its default message display.

**MOVE.TOOL**(from\_bar\_id,  
from\_bar\_position, to\_bar\_id,  
to\_bar\_position, TRUE)

With Toolbars(from\_bar\_id).ToolbarButtons(from\_bar\_position)  
.Copy toolbar := Toolbars(to\_bar\_id), \_  
before := to\_bar\_position  
End With

Copies a button from one toolbar to another.

**MOVE.TOOL**(from\_bar\_id,  
from\_bar\_position, to\_bar\_id,  
to\_bar\_position, FALSE)

With Toolbars(from\_bar\_id).ToolbarButtons(from\_bar\_position)  
.Move toolbar := Toolbars(to\_bar\_id), \_  
before := to\_bar\_position  
End With

Moves a button from one toolbar to another.

**MOVE.TOOL**(from\_bar\_id,  
from\_bar\_position, , , width)

With Toolbars(from\_bar\_id).ToolbarButtons(from\_bar\_position)  
.Width = width  
End With

Resizes a toolbar button that displays a drop-down list.

## N

**NAMES**(document\_text, 3)

For Each n In ActiveWorkbook.Names  
MsgBox n.Name  
Next

Returns an array of all range names. In Visual Basic the Names collection provides direct access to Name objects and does not require using text names. The example displays the name of each range name in a message box. If 1 or 2 is specified for type\_num, examine the Visible property of the Name object to determine whether the name should be included.

**NAMES**(document\_text, 2, match\_text)

Dim result() As String  
numNames = 0  
For Each n In ActiveWorkbook.Names  
If n.Visible = False And n.Name Like match\_text Then  
numNames = numNames + 1  
ReDim Preserve result(1 To numNames)  
result(numNames) = n.Name  
End If

Next

Returns an array of range names with titles matching a pattern, with additional filtering for hidden names only. The example constructs an array of range names that match the pattern and are hidden. Note the use of the Names collection in a For Each...Next statement.

**NEW**(type\_num, xy\_series, add\_logical)      Workbooks.Add template := type\_num

Adds a new workbook based on a template of type\_num if type\_num is text. If type\_num is omitted, a default workbook is created. The xy\_series and add\_logical arguments have no direct equivalent in Visual Basic.

**NEW**(1)                      Workbooks.Add template := xlWorksheet

Adds a new workbook containing a single worksheet.

**NEW**(2)                      Workbooks.Add template := xlChart

Adds a new workbook containing a single chart based on the current selection.

**NEW**(3)      Workbooks.Add template := xlExcel4MacroSheet

Adds a new workbook containing a single Microsoft Excel 4.0 macro sheet.

**NEW**(4)                      Workbooks.Add template := xlExcel4IntlMacroSheet

Adds a new workbook containing a single Microsoft Excel 4.0 international macro sheet.

**NEW**(5)                      Workbooks.Add

Adds a new workbook containing the default number of worksheets or based on the default workbook template.

NEW.WINDOW( )                      ActiveWindow.NewWindow

```
NEXT() For x = 1 To 100
 ' loop body
Next
```

Ends a loop. The example illustrates the full Visual Basic syntax.

**NOTE**(, cell\_ref)      Range(*cell\_ref*).ClearNotes

Deletes a cell note.

**NOTE**(add\_text, cell\_ref, start\_char, num\_chars) Range(*cell\_ref*).NoteText text := add\_text, start := start\_char, length := num\_chars

Adds text to a cell note.

## Visual Basic Equivalents for Macro Functions and Commands

.....



### O

**OBJECT.PROPERTIES**(placement\_type,  
print\_object)

With Selection  
    .Placement = *placement\_type*  
    .PrintObject = print\_object  
End With

**OBJECT.PROTECTION**(locked, lock\_text)

With Selection  
    .Locked = locked  
    .LockedText = lock\_text  
End With

**ON.DATA**(document\_text, macro\_text)

ActiveWorkbook.SetLinkOnData name := document\_text, \_  
    procedure := macro\_text

**ON.DOUBLECLICK**(sheet\_text,  
macro\_text)

Sheets(*sheet\_text*).OnDoubleClick = macro\_text

Runs the specified macro when a particular sheet is double-clicked.

**ON.DOUBLECLICK**(, macro\_text)

Application.OnDoubleClick = macro\_text

Runs the specified macro when a double-click occurs on any sheet that does not already have a double-click macro defined.

**ON.ENTRY**(sheet\_text, macro\_text)

Sheets(*sheet\_text*).OnEntry = macro\_text

Runs the specified macro whenever data is entered into a cell on a particular sheet.

**ON.ENTRY**(, macro\_text)

Application.OnEntry = macro\_text

Runs the specified macro whenever data is entered into a cell on any sheet that does not already have a data entry macro defined.





**OPEN**(file\_text, , , , write\_res\_pwd, , , , , 2, notify\_logical)

Workbooks(file\_text).ChangeFileAccess \_  
mode := xlReadWrite, notify := notify\_logical, \_  
writePassword := write\_res\_pwd

Changes a workbook to read-write access.

**OPEN**(file\_text, , , , , , , , , 3)

Workbooks(file\_text).ChangeFileAccess mode := xlReadOnly

Changes access to a workbook to read-only.

**OPEN**(file\_text, , TRUE, , , , , , 1)

Workbooks(file\_text).UpdateFromFile

Updates a workbook with the most recently saved disk copy, possibly discarding changes made in memory.

**OPEN.DIALOG**(file\_filter, button\_text, title, filter\_index)

Application.GetOpenFilename fileFilter := file\_filter, \_  
filterIndex := filter\_index, title := title, \_  
buttonText := button\_text

**OPEN.LINKS**(document\_text1, document\_text2, ..., read\_only, type\_of\_link)

ActiveWorkbook.OpenLinks \_  
name := Array(document\_text1, document\_text2), \_  
readOnly = read\_only, type := type\_of\_link

Opens the specified supporting documents. If multiple documents are specified, Visual Basic requires an array for the name argument. The example uses the Array function to construct an array.

**OPEN.MAIL**(subject, comments)

ExecuteExcel4Macro("comments.comments(subject, comments)")

No direct equivalent.

**OPEN.TEXT**(file\_name, file\_origin, start\_row, file\_type, text\_qualifier, consecutive\_delim, tab, semicolon, comma, space, other, other\_char, {field\_info; field\_info2;...})

Workbooks.OpenText filename := file\_name, \_  
origin := file\_origin, startRow := start\_row, \_  
dataType := file\_type, textQualifier := text\_qualifier, \_  
consecutiveDelimiter := consecutive\_delim, \_  
tab := tab, semicolon := semicolon, comma := comma, \_  
space := space, other := other\_char, \_  
otherChar := other\_char, fieldInfo := field\_info2

**OPTIONS.CALCULATION**(type\_num, iter, max\_num, max\_change, update, precision, date\_1904, calc\_save, save\_values)

With Application  
.Calculation = type\_num  
.Iteration = iter  
.MaxIterations = max\_num  
.MaxChange = max\_change  
.CalculateBeforeSave = calc\_save

End With

With ActiveWorkbook

.UpdateRemoteReferences = update  
.PrecisionAsDisplayed = precision  
.Date1904 = date\_1904  
.SaveLinkValues = save\_values

End With

**OPTIONS.CHART**(display\_blanks, plot\_visible, size\_with\_window)

With ActiveChart  
.DisplayBlanksAs = display\_blanks  
.PlotVisibleOnly = plot\_visible  
.SizeWithWindow = size\_with\_window

End With

**OPTIONS.EDIT**(incell\_edit, drag\_drop, alert, entermove, fixed, decimals,copy\_objects, update\_links)

With Application

.EditDirectlyInCell = incell\_edit  
.CellDragAndDrop = drag\_drop  
.AlertBeforeOverwriting = alert  
.MoveAfterReturn = entermove  
.FixedDecimal = fixed  
.FixedDecimalPlaces = decimals  
.CopyObjectsWithCells = copy\_objects  
.AskToUpdateLinks = update\_links

End With

**OPTIONS.GENERAL**(R1C1\_mode, dde\_on, sum\_info, tips, recent\_files, old\_menus, user\_info, font\_name, font\_size, default\_location, alternate\_location, sheet\_num, enable\_under)

With Application

.ReferenceStyle = *R1C1\_mode*  
.IgnoreRemoteRequests = dde\_on  
.PromptForSummaryInfo = sum\_info  
.DisplayRecentFiles = recent\_files  
.DisplayExcel4Menus = old\_menus  
.UserName = user\_info  
.StandardFont = font\_name  
.StandardFontSize = font\_size  
.DefaultFilePath = default\_location  
.AltStartupPath = alternate\_location  
.SheetsInNewWorkbook = sheet\_num  
.CommandUnderlines = *enable\_under*

End With

Application.ResetTipWizard

If the tips argument is FALSE or omitted, do not include the ResetTipWizard method.

**OPTIONS.LISTS.ADD**(string\_array)

Application.AddCustomList listArray := string\_array

Adds the contents of an array as a custom list . In Visual Basic, use the Array function to easily create arrays of string values.

**OPTIONS.LISTS.ADD**(import\_ref, by\_row)

Application.AddCustomList listArray := *import\_ref*, \_  
byRow := by\_row

Adds the contents from a cell range as a custom list.

**OPTIONS.LISTS.DELETE**(list\_num)

Application.DeleteCustomList listNum = list\_num

**OPTIONS.LISTS.GET**(list\_num)

Application.GetCustomListContents(listNum := list\_num)

**OPTIONS.TRANSITION**(menu\_key, menu\_key\_action, nav\_keys, trans\_eval, trans\_entry )

With Application

.TransitionMenuKey = menu\_key\_action  
.TransitionMenuKeyAction = *menu\_key\_action*  
.TransitionNavigKeys = nav\_keys

End With

With ActiveSheet

.TransitionExpEval = trans\_eval  
.TransitionFormEntry = trans\_entry

End With

**OPTIONS.VIEW**(formula, status, notes, show\_info, object\_num, page\_breaks,

With Application

.DisplayFormulaBar = formulas

formulas, gridlines, color\_num, headers,  
display\_outline, zeros, hor\_scroll,  
vert\_scroll, sheet\_tabs)

```
.DisplayStatusBar = status
.DisplayNoteIndicator = notes
.DisplayInfoWindow = show_info
```

End With

```
ActiveWorkbook.DisplayDrawingObjects = object_num
```

```
ActiveSheet.DisplayAutomaticPageBreaks = page_breaks
```

With ActiveWindow

```
.DisplayFormulas = formulas
.DisplayGridlines = gridlines
.GridlineColorIndex = color_num
.DisplayHeadings = headers
.DisplayOutline = display_outline
.DisplayZeros = zeros
.DisplayHorizontalScrollBar = hor_scroll
.DisplayVerticalScrollBar = vert_scroll
.DisplayWorkbookTabs = sheet_tabs
```

End With

**OUTLINE**(auto\_styles, row\_dir, col\_dir)

With ActiveSheet.Outline

```
.AutomaticStyles = auto_styles
.SummaryRow = row_dir
.SummaryColumn = col_dir
```

End With

Changes outline settings.

**OUTLINE**(, , , 1)

Selection.AutoOutline

Creates an outline with the current outline settings.

**OUTLINE**(, , , 2)

Selection.ApplyOutlineStyles

Applies outline styles to the outlined selection.

**OVERLAY**(type\_num, stack, 100, vary,  
overlap, drop, hilo, overlap%, cluster,  
angle, series\_num, auto)

With ActiveChart.ChartGroups(2)

```
.Type = type_num
.SubType = 100
.VaryByCategories = vary
.Overlap = overlap%
.HasDropLines = drop
.HasHiLoLines = hilo
.FirstSliceAngle = angle
```

End With

This function is used only for backwards compatability with Microsoft Excel 2.2 or earlier. Some arguments do not apply to all chart types. The stack, overlap, cluster, series\_num, and auto arguments have no direct equivalent in Microsoft Excel 5.0.

## P

**PAGE.SETUP**(head, foot, left, right, top,  
bot, hdng, grid, h\_cntr, v\_cntr, orient,  
paper\_size, scale, pg\_num, pg\_order,  
bw\_cells, quality, head\_margin,  
foot\_margin, notes, draft)

With ActiveSheet.PageSetup

```
.LeftHeader = head
.CenterHeader = head
.RightHeader = head
. = foot
.CenterFooter = foot
.RightFooter = foot
.LeftMargin = Application.InchesToPoints(left)
```

```

.RightMargin = Application.InchesToPoints(right)
.TopMargin = Application.InchesToPoints(top)
.BottomMargin = Application.InchesToPoints(bot)
.PrintHeadings = hdng
.PrintGridlines = grid
.CenterHorizontally = h_cntr
.CenterVertically = v_cntr
.Orientation = orient
.PaperSize = paper_size
If scale = True Then
 .Zoom = False
 .FitToPagesWide = 1
 .FitToPagesTall = 1
ElseIf IsArray(scale) Then
 .Zoom = False
 If IsError(scale(1)) Then
 .FitToPagesWide = False
 Else
 .FitToPagesWide = scale(1)
 End If
 If IsError(scale(2)) Then
 .FitToPagesTall = False
 Else
 .FitToPagesTall = scale(2)
 End If
Else
 .Zoom = scale
End If
.FirstPageNumber = pg_num
.Order = pg_order
.BlackAndWhite = bw_cells
.PrintQuality = quality
.HeaderMargin = _
 Application.InchesToPoints(head_margin)
.FooterMargin = Application.InchesToPoints(foot_margin)
.PrintNotes = notes
.Draft = draft

```

End With

Controls page settings for worksheets and macro sheets. In Visual Basic the head and foot arguments are split into separate left, center, and right format strings. Note also the translation required for the scale argument.

**PAGE.SETUP**(head, foot, left, right, top, bot, size, h\_cntr, v\_cntr, orient, paper\_size, scale, pg\_num, bw\_chart, quality, head\_margin, foot\_margin, draft)

```

With ActiveSheet.PageSetup
.LeftHeader = head_margin
.CenterHeader = head_margin
.RightHeader = head_margin
.LeftFooter = foot_margin
.CenterFooter = foot_margin
.RightFooter = foot_margin
.LeftMargin = Application.InchesToPoints(left)
.RightMargin = Application.InchesToPoints(right)
.TopMargin = Application.InchesToPoints(top)
.BottomMargin = Application.InchesToPoints(bot)
.ChartSize = paper_size
.CenterHorizontally = h_cntr
.CenterVertically = v_cntr
.Orientation = orient
.PaperSize = paper_size
.Zoom = scale

```

```
.FirstPageNumber = pg_num
.BlackAndWhite = bw_chart
.PrintQuality = quality
.HeaderMargin = _
 Application.InchesToPoints(head_margin)
.FooterMargin = Application.InchesToPoints(foot_margin)
.Draft = draft
```

End With

Controls page settings for charts. In Visual Basic the head and foot arguments are split into separate left, center, and right format strings.

**PAGE.SETUP**(*head*, *foot*, *left*, *right*, *top*,  
*bot*, *orient*, *paper\_size*, *scale*, *quality*,  
*head\_margin*, *foot\_margin*, *pg\_num*)

With ActiveSheet.PageSetup

```
.LeftHeader = head
.CenterHeader = head
.RightHeader = head
.LeftFooter = foot
.CenterFooter = foot
.RightFooter = foot
.LeftMargin = Application.InchesToPoints(left)
.RightMargin = Application.InchesToPoints(right)
.TopMargin = Application.InchesToPoints(top)
.BottomMargin = Application.InchesToPoints(bot)
.Orientation = orient
.PaperSize = paper_size
.Zoom = scale
.PrintQuality = quality
.HeaderMargin = _
 Application.InchesToPoints(head_margin)
.FooterMargin = Application.InchesToPoints(foot_margin)
.FirstPageNumber = pg_num
```

End With

Controls page settings for Visual Basic modules. In Visual Basic the head and foot arguments are split into separate left, center, and right format strings.

**PARSE**(*parse\_text*, *destination\_ref*)

```
Selection.Parse parseLine := parse_text, _
 destination := destination_ref
```

**PASTE**(*to\_reference*)

```
ActiveSheet.Paste destination := to_reference
```

Pastes data onto a sheet or chart. If a chart is active use ActiveChart instead of ActiveSheet and omit the destination argument. Note that the Paste method is dependent on the current selection.

**PASTE**( )

```
Selection.Paste
```

Pastes an image onto the selected point or series of a picture chart.

**PASTE**( )

```
ActiveChart.Paste
```

Pastes an entire chart onto the active chart.

**PASTE.LINK**( )

```
ActiveSheet.Paste link := True
```

Establishes a link with the source of the data being pasted.

**PASTE.PICTURE**( )

```
ActiveSheet.Pictures.Paste
```

Pastes a picture of the data on the clipboard. If the active sheet is a chart, use ActiveChart instead of ActiveSheet.

**PASTE.PICTURE.LINK**( )

ActiveSheet.Pictures.Paste link := True

Pastes a picture which is linked to the source of the data on the clipboard. If the active sheet is a chart, use ActiveChart instead of ActiveSheet.

**PASTE.SPECIAL**(paste\_num, operation\_num, skip\_blanks, transpose)

```
Selection.PasteSpecial paste := paste_num, _
operation := operation_num, skipBlanks := skip_blanks, _
transpose := transpose
```

Pastes worksheet or macro sheet range data.

**PASTE.SPECIAL**(rowcol, titles, categories, replace, series)

```
ActiveChart.SeriesCollection.Paste rowcol := rowcol, _
seriesLabels := titles, categoryLabels := categories, _
replace := replace, newSeries := series
```

Pastes data points or data series information onto a chart.

**PASTE.SPECIAL**(paste\_num)ActiveChart.Paste type := *paste\_num*

Pastes chart information onto a chart.

**PASTE.SPECIAL**(format\_text, pastelink\_logical, display\_icon\_logical, icon\_file, icon\_number, icon\_label)

```
ActiveSheet.PasteSpecial format := format_text, _
link := pastelink_logical, _
displayAsIcon := display_icon_logical, _
iconFileName := icon_file, iconIndex := icon_number, _
iconLabel := icon_label
```

Pastes data from another application.

**PASTE.TOOL**(bar\_id, position)

Toolbars(bar\_id).ToolbarButtons(position).PasteFace

**PATTERNS**(apattern, afore, aback, TRUE)

```
With Selection.Interior
 If apattern = 0 Then
 .ColorIndex = xlNone
 Else
 .ColorIndex = aback
 .Pattern = apattern
 .PatternColorIndex = afore
 End If
End With
```

Syntax 1: Sets pattern and color information for the selected range. Visual Basic has no direct equivalent if the newui argument is FALSE, which indicates Microsoft Excel 4.0 compatibility mode.

**PATTERNS**(lauto, lstyle, lcolor, lwt, hwidth, hlength, htype)

```
With Selection.Border
 If lauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf lauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = lstyle
 .ColorIndex = lcolor
 .Weight = lwt
 End If
End With
With Selection
 .ArrowHeadWidth = hwidth
 .ArrowHeadStyle = htype
 .ArrowHeadLength = hlength
```

End With

Syntax 2: Sets color and line style information for lines and arrows on worksheets and charts.

**PATTERNS**(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, rounded, TRUE)

```
With Selection.Border
 If bauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf bauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = bstyle
 .ColorIndex = bcolor
 .Weight = bwt
 End If
End With
Selection.Shadow = shadow
With Selection.Interior
 If aauto = 1 Then
 .Pattern = xlAutomatic
 ElseIf aauto = 2 Then
 .Pattern = xlNone
 Else
 .ColorIndex = aback
 .Pattern = apattern
 .PatternColorIndex = afore
 End If
End With
Selection.RoundedCorners = rounded
```

Syntax 3: Sets pattern and border information for filled drawing objects on worksheets and charts.

Visual Basic has no direct equivalent if the newui argument is FALSE, which indicates Microsoft Excel 4.0 compatibility mode.

**PATTERNS**(bauto, bstyle, bcolor, bwt, shadow, aauto, apattern, afore, aback, invert, apply, TRUE)

```
With Selection.Border
 If bauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf bauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = bstyle
 .ColorIndex = bcolor
 .Weight = bwt
 End If
End With
Selection.Shadow = shadow
With Selection.Interior
 If aauto = 1 Then
 .Pattern = xlAutomatic
 ElseIf aauto = 2 Then
 .Pattern = xlNone
 Else
 .ColorIndex = aback
 .Pattern = apattern
 .PatternColorIndex = afore
 End If
End With
Selection.InvertIfNegative = invert
```

Syntax 4: Sets pattern and border information for chart bars, columns, areas, and text labels. Visual

Basic has no direct equivalent if the newfill argument is FALSE, which indicates Microsoft Excel 4.0 compatibility mode. The apply argument has no equivalent in Visual Basic.

**PATTERNS**(lauto, lstyle, lcolor, lwt, tmajor, tminor, tlabel)

```
With Selection.Border
 If bauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf bauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = bstyle
 .ColorIndex = bcolor
 .Weight = bwt
 End If
End With
With Selection
 .MajorTickMark = tmajor
 .MinorTickMark = tminor
 .TickLabelPosition = tlabel
End With
```

Syntax 5: Sets border and style information for chart axes.

**PATTERNS**(lauto, lstyle, lcolor, lwt, apply, smooth)

```
With Selection.Border
 If lauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf lauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = lstyle
 .ColorIndex = lcolor
 .Weight = lwt
 End If
End With
Selection.Smooth = smooth
```

Syntax 6: Sets border information for chart lines other than chart data lines. The apply argument has no equivalent in Visual Basic.

**PATTERNS**(lauto, lstyle, lcolor, lwt, mauto, mstyle, mfore, mback, apply, smooth)

```
With Selection.Border
 If lauto = 1 Then
 .LineStyle = xlAutomatic
 ElseIf lauto = 2 Then
 .LineStyle = xlNone
 Else
 .LineStyle = lstyle
 .ColorIndex = lcolor
 .Weight = lwt
 End If
End With
With Selection
 If mauto = 1 Then
 .MarkerStyle = xlAutomatic
 ElseIf mauto = 2 Then
 .MarkerStyle = xlNone
 Else
 .MarkerStyle = mstyle
 .MarkerForegroundColorIndex = mfore
 .MarkerBackgroundColorIndex = mback
 End If
End With
```



```
.Smooth = smooth
End With
```

Syntax 7: Sets border information for chart data lines. The apply argument has no equivalent in Visual Basic.

```
PATTERNS(type, picture_units, apply) With Selection
 .PictureType = type
 .PictureUnit = picture_units
End With
```

Syntax 8: Sets picture chart information for picture chart data markers. The apply argument has no equivalent in Visual Basic.

**PAUSE**(no\_tool)

No direct equivalent. Use a breakpoint or the Stop statement to halt execution and show the Debug window. Once halted, use the Immediate pane to examine and set variables.

```
PIVOT.ADD.DATA(name, With ActiveSheet.PivotTables(name).
pivot_field_name, new_name, position, PivotFields(pivot_field_name)
function, calculation, base_field, .Orientation = xlDataField
base_item, format_text) .Name = new_name
 .Position = position
 .Function = function
 .Calculation = calculation
 .BaseField = base_field
 .BaseItem = base_item
 .NumberFormat = format_text
End With
```

If the name argument is omitted, use ActiveCell.PivotTable instead of ActiveSheet.PivotTables(name). The same replacement can be used for all of the following pivot table functions that have a pivot table name argument.

```
PIVOT.ADD.FIELDS(name, row_array, ActiveSheet.PivotTables(name).AddFields
column_array, page_array, add_to_table) rowFields := row_array, columnFields := column_array, _
 pageFields := page_array, addToTable := add_to_table
```

```
PIVOT.FIELD(name, pivot_field_name, With ActiveSheet.PivotTables(name).
orientation, position) PivotFields(pivot_field_name)
 .Orientation = orientation
 .Position = position
End With
```

Pivots a specified field within a specified pivot table.

```
PIVOT.FIELD(, , orientation, position) With ActiveCell.PivotField
 .Orientation = orientation
 .Position = position
End With
```

Pivots the field that contains the active cell.

```
PIVOT.FIELD.GROUP(start, end, by, Selection.Group start := start, end := end, by := by, _
periods) periods := periods
```

Creates groups within a pivot field. In Visual Basic the periods argument is an array of logical values specifying the date periods to group by. The Microsoft Excel 4.0 macro period argument must be converted from a summed number into an array of logical values.

**PIVOT.FIELD.PROPERTIES**(name,  
pivot\_field\_name, new\_name, orientation,  
function, formats)

```
With ActiveSheet.PivotTables(name). _
 PivotFields(pivot_field_name)
 .Name = new_name
 .Orientation = orientation
 .Subtotals = function
 For x = 1 To UBound(formats, 1)
 .PivotItems(formats(x, 1)).Visible = formats(x, 2)
 Next
End With
```

Changes the properties of a pivot table header field. In Visual Basic the Subtotals property is an array of logical values specifying the subtotals calculated for the field. The PIVOT.FIELD.PROPERTIES function argument must be converted from a summed number into an array of logical values. The formats argument is an array specifying the visibility of items in the field. Visual Basic requires that the properties of pivot items be set individually for each field. The example iterates over the array to set the Visible property according to the array contents.

**PIVOT.FIELD.PROPERTIES**(name,  
pivot\_field\_name, new\_name, orientation,  
function, formats)

```
With ActiveSheet.PivotTables(name). _
 PivotFields(pivot_field_name)
 .Name = new_name
 .Orientation = orientation
 .Function = function
 .Calculation = formats(1)
 .BaseField = formats(2)
 .BaseItem = formats(3)
 .NumberFormat = formats(4)
End With
```

Changes the properties of a pivot table data field. In Visual Basic the function argument is converted from an array into four distinct properties.

**PIVOT.FIELD.UNGROUP**( )

```
Selection.Ungroup
```

**PIVOT.ITEM**(name, pivot\_field\_name,  
pivot\_item\_name, position)

```
ActiveSheet.PivotTables(name). _
 PivotFields(pivot_field_name). _
 PivotItems(pivot_item_name).Position = position
```

Moves the specified item in a pivot table.

**PIVOT.ITEM**(, , , position)

```
ActiveCell.PivotItem.Position = position
```

Moves the pivot item containing the active cell.

**PIVOT.ITEM.PROPERTIES**(name,  
pivot\_field\_name, pivot\_item\_name,  
new\_name, position, show, TRUE)

```
With ActiveSheet.PivotTables(name). _
 PivotFields(pivot_field_name). _
 PivotItems(pivot_item_name)
 .Name = new_name
 .Position = position
 .Visible = show
End With
ActiveSheet.PivotTables(name).PivotFields(pivot_field_name).
 - CurrentPage = new_name
```

Changes the name and properties of a pivot item in a header field, and makes the item become the active item in the page field.

**PIVOT.ITEM.PROPERTIES**(name,

```
With ActiveSheet.PivotTables(name). _
```

|                                                                       |                                                                                                                                                                                                                        |
|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>pivot_field_name, pivot_item_name, ,<br/>position, show, TRUE)</p> | <pre> PivotFields(pivot_field_name). _ PivotItems(pivot_item_name) .Position = position .Visible = show End With ActiveSheet.PivotTables(name). _ PivotFields(pivot_field_name).CurrentPage = _ pivot_item_name </pre> |
|-----------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Changes the properties of a pivot item in a header field and makes the item the active item in the page field.

|                                                                                                                              |                                                                                                                                                                                |
|------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b><u>PIVOT.ITEM.PROPERTIES</u></b>(name,<br/>pivot_field_name, pivot_item_name,<br/>new_name, position, show, FALSE)</p> | <pre> With ActiveSheet.PivotTables(name). _ PivotFields(pivot_field_name). _ PivotItems(pivot_item_name) .Name = new_name .Position = position .Visible = show End With </pre> |
|------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Changes the properties of a pivot item in a header field without changing the active item in the page field.

|                                          |                                                         |
|------------------------------------------|---------------------------------------------------------|
| <p><b><u>PIVOT.REFRESH</u></b>(name)</p> | <pre> ActiveSheet.PivotTables(name).RefreshTable </pre> |
|------------------------------------------|---------------------------------------------------------|

|                                                         |                                                                                |
|---------------------------------------------------------|--------------------------------------------------------------------------------|
| <p><b><u>PIVOT.SHOW.PAGES</u></b>(name, page_field)</p> | <pre> ActiveSheet.PivotTables(name).ShowPages _ pageField := page_field </pre> |
|---------------------------------------------------------|--------------------------------------------------------------------------------|

|                                                                                                                                                |                                                                                                                                                                                                                                                                                    |
|------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b><u>PIVOT.TABLE.WIZARD</u></b>(type, source,<br/>destination, name, row_grand, col_grand,<br/>save_data, apply_auto_format, autopage)</p> | <pre> ActiveSheet.PivotTableWizard sourceType := type, _ SourceData := source, tableDestination := destination, _ tableName := name, rowGrand := row_grand, _ columnGrand := col_grand, saveData := save_data, _ hasAutoFormat := apply_auto_format, _ autoPage := autopage </pre> |
|------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                             |                                                                                                    |
|-------------------------------------------------------------|----------------------------------------------------------------------------------------------------|
| <p><b><u>POKE</u></b>(channel_num, item_text, data_ref)</p> | <pre> Application.DDEPoke channel := channel_num, _ item := item_text, data_ref := data_ref </pre> |
|-------------------------------------------------------------|----------------------------------------------------------------------------------------------------|

|                                         |                                                         |
|-----------------------------------------|---------------------------------------------------------|
| <p><b><u>PRECISION</u></b>(logical)</p> | <pre> Application.PrecisionAsDisplayed = logical </pre> |
|-----------------------------------------|---------------------------------------------------------|

|                                   |                                                                    |
|-----------------------------------|--------------------------------------------------------------------|
| <p><b><u>PREFERRED</u></b>( )</p> | <pre> ActiveChart.AutoFormat gallery := xlDefaultAutoFormat </pre> |
|-----------------------------------|--------------------------------------------------------------------|

|                                                         |                                                                      |
|---------------------------------------------------------|----------------------------------------------------------------------|
| <p><b><u>PRESS.TOOL</u></b>(bar_id, position, down)</p> | <pre> Toolbars(bar_id).ToolbarButtons(position).Pushed = down </pre> |
|---------------------------------------------------------|----------------------------------------------------------------------|

|                                                                                                        |                                                                      |
|--------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| <p><b><u>PRINT</u></b>(1, , , copies, , preview, , , , , 1)<br/>Prints all pages of the selection.</p> | <pre> Selection.PrintOut copies := copies, preview := preview </pre> |
|--------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|

|                                                                                                                        |                                                                                                |
|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|
| <p><b><u>PRINT</u></b>(2, from, to, copies, , preview, , , , , 1)<br/>Prints the specified pages of the selection.</p> | <pre> Selection.PrintOut from := from, to := to, copies := copies, _ preview := preview </pre> |
|------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|

|                                                                                                                      |                                                                                                                      |
|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| <p><b><u>PRINT</u></b>(1, , , copies, draft, preview,<br/>print_what, color, feed, quality,<br/>y_resolution, 1)</p> | <pre> ' Save the page setup variables to restore after override With ActiveSheet.PageSetup saveDraft = .Draft </pre> |
|----------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|

Prints all pages of the selection, temporarily overriding the draft quality and print notes page setup settings. The color, feed, quality, and y\_resolution arguments have no direct equivalents in Visual Basic. A print\_what value of 2, which prints only notes, has no equivalent in Visual Basic.

Prints all pages of the selected sheets.

Prints all pages of the Info window, if the Info window is the active window.

Prints all pages of the active workbook.

Previews printing of the active sheet.

Previews printing of the Info window, if the Info window is the active window.

```
PRINTER.SETUP(printer_text) Application.ActivePrinter = printer_text
```

Promotes, or ungroups, the rows of the selection. The parameter 1 is optional.

Promotes, or ungroups, the columns of the selection.

Protects the active sheet. The windows argument has no direct equivalent in Microsoft Excel 5.0.

**PROTECT.DOCUMENT**(FALSE, windows,      ActiveSheet.Unprotect password := password  
password, FALSE, FALSE)

Unprotects the active sheet. The windows argument has no direct equivalent in Microsoft Excel 5.0.

**PUSHBUTTON.PROPERTIES**( default\_log      With Selection  
ical, cancel\_logical, dismiss\_logical,      .DefaultButton = default\_logical  
help\_logical, accel\_text, accel\_text2)      .CancelButton = cancel\_logical  
                                                  .DismissButton = dismiss\_logical  
                                                  .HelpButton = help\_logical  
                                                  .Accelerator = accel\_text  
                                                  .PhoneticAccelerator = accel\_text2  
End With

**Q**

**QUIT**( )      Application.Quit

.....



No direct equivalent. In Visual Basic the Range object is more flexible than the REFTEXT function and should not require you to manipulate references as text. The example returns the range address as an absolute A1-style reference in the form of text (the REFTEXT function always preserves the absolute or relative nature of the original reference).

No direct equivalent. The example returns the range address as an absolute R1C1-style reference in the form of text.

Registers a function in a DLL or code resource so that it can be called from the Microsoft Excel 4.0 Macro Language. In Visual Basic the Declare statement registers DLL or code resource functions. For a full description of the Declare statement, see Chapter 10, "Controlling and Communicating with Other Applications," in the Microsoft Excel Visual Basic User's Guide. The example shows registration of a single function in SAMPLE.DLL returning a Boolean value and taking an integer and a string parameter.

Automatically registers all worksheet and Microsoft Excel 4.0 Macro Language functions within a DLL or code resource.

**RELREF**(reference, rel\_to\_ref)

Range(*reference*).Address rowAbsolute := False, \_  
columnAbsolute := False, referenceStyle := xlR1C1, \_  
relativeTo := Range(*rel\_to\_ref*)

**REMOVE.LIST.ITEM**(index\_num,  
count\_num)

Selection.RemoveItem index := index\_num, count :=  
count\_num

**REMOVE.PAGE.BREAK**( )

ActiveCell.PageBreak = xlNone

Removes manual page breaks from the left or above the active cell.

**REMOVE.PAGE.BREAK**( )

Cells.PageBreak = xlNone

Removes manual page breaks from the entire sheet if all cells are selected.

**RENAME.COMMAND**(bar\_num, menu,  
command, name\_text)

Menubars(*bar\_num*).Menus(menu). \_  
MenuItems(name\_text).Caption = name\_text

Renames a command on a menu.

**RENAME.COMMAND**(bar\_num, menu,  
command, name\_text, position)

Menubars(*bar\_num*).Menus(menu). \_  
MenuItems(position). \_  
MenuItems(position).Caption = name\_text

Renames a common on a submenu.

**RENAME.COMMAND**(bar\_num, menu,  
command, name\_text)

ShortcutMenus(*shortcut\_menu*). \_  
MenuItems(name\_text).Caption = name\_text

Renames a command on a shortcut menu. Shortcut menus are accessed in Visual Basic by using the ShortcutMenus method with the appropriate constant.

**RENAME.OBJECT**(new\_name)

Selection.Name = new\_name

**REPLACE.FONT**(font\_num, name\_text,  
size\_num, bold, italic, underline, strike,  
color, outline, shadow)

This function is provided only for macro compatability with previous versions of Microsoft Excel. It has no direct equivalent in Visual Basic. See the FONT.PROPERTIES function for information on how to change fonts.

**REQUEST**(channel\_num, item\_text)

Application.DDERequest channel := channel\_num, \_  
item := item\_text

**RESET.TOOL**(bar\_id, position)

Toolbars(bar\_id).ToolBarButtons(position).Reset

**RESET.TOOLBAR**(bar\_id)

Toolbar(bar\_id).Reset

**RESTART**(level\_num)

No direct equivalent. In Visual Basic you can use GoTo statements to explicitly jump to a particular piece of code, but you cannot arbitrarily return from a called routine to a particular caller.

**RESULT**(type\_num)

Function MyProcedure ( ) As String

Specifies the return type of a function. In Visual Basic the result of a function is declared as a part of

the function header. The example shows a procedure returning a Text or String value.

If the type\_num parameter specifies more than one return type--for example, you use the return type\_num 5 to specify either a number (1) or a logical (4) return value--in Visual Basic you need to specify the return type as a Variant.

### **RESUME**(type\_num)

No direct equivalent. You cannot pause Visual Basic macros. Instead, to assist in debugging a macro Visual Basic provides the Debug Window and breakpoints or watchpoints.

### **RETURN**(value)

```
Function MyFunction ()
 ' Often a function returns before reaching the normal
 ' end of the function, for example after an error occurs.
 If AnErrorOccurred Then
 MyFunction = value
 Exit Function
 End If

 ' No Exit Function is necessary at the end of the function
 ' because the End Function statement implies a return
 MyFunction = value
End Function
```

Returns from a subroutine or function and specifies the return value. In Visual Basic you specify the return value of a function by assigning the return value to the name of the function. A function returns when the end of the function is reached at the End Function statement, or when an Exit Function statement is executed. The example shows assignment of the return value and illustrates use of both forms of returning from a function.

### **ROUTE.DOCUMENT**( )

ActiveWorkbook.Route

**ROUTING.SLIP**(recipients,subject,  
message, route\_num, return\_logical,  
status\_logical)

```
ActiveWorkbook.HasRoutingSlip = True
With ActiveWorkbook.RoutingSlip
 .Recipients = recipients
 .Subject = subject
 .Message = message
 .Delivery = route_num
 .ReturnWhenDone = return_logical
 .TrackStatus = status_logical
End With
```

### **ROW.HEIGHT**(, reference, , 1)

Rows(*reference*).Hidden = True

Hides rows. If hiding the selection, use Selection.EntireRow instead of the Rows method.

### **ROW.HEIGHT**(, reference, , 2)

Rows(*reference*).Hidden = False

Unhides rows.

### **ROW.HEIGHT**(, reference, , 3)

Rows(*reference*).AutoFit

Sizes the rows in reference to the height of the largest item in each row.

### **ROW.HEIGHT**(height\_num, reference)

Rows(*reference*).RowHeight = height\_num

Sets the row height to the specified value.

### **ROW.HEIGHT**(, reference, TRUE)

Rows(*reference*).RowHeight = ActiveSheet.StandardHeight



Sets the row height to the standard row height value.

**RUN**(reference, step)

Application.Run macro := Range(*reference*)

Runs a particular macro. Visual Basic has no equivalent for the step parameter. Note that the Visual Basic Run method can pass parameters and return values of the called macro. To run a Microsoft Excel 4-style macro from Visual Basic, use the Application.ExecuteExcel4Macro method.

**RUN**(1)

ActiveWorkbook.RunAutoMacros which := xlAutoOpen

**RUN**(2)

ActiveWorkbook.RunAutoMacros which := xlAutoClose

**RUN**(3)

ActiveWorkbook.RunAutoMacros which := xlAutoActivate

**RUN**(4)

ActiveWorkbook.RunAutoMacros which := xlAutoDeactivate

## Visual Basic Equivalents for Macro Functions and Commands

.....



### S

**SAVE**( )

ActiveWorkbook.Save

**SAVE.AS**(document\_text, type\_num, prot\_pwd, backup, write\_res\_pwd, read\_only\_rec)

ActiveWorkbook.SaveAs filename := document\_text, \_  
fileFormat := *type\_num*, password := prot\_pwd, \_  
writeResPassword := write\_res\_pwd, \_  
readOnlyRecommended := read\_only\_rec, \_  
createBackup := backup

**SAVE.AS**(, 0)

ActiveWorkbook.Saved = True

Changes the workbook so that Microsoft Excel will not prompt to save the workbook when it is closed.

**SAVE.COPY.AS**(document\_text)

ActiveWorkbook.SaveCopyAs filename := document\_text

**SAVE.DIALOG**(init\_filename, title, button\_text, file\_filter, filter\_index)

Application.GetSaveAsFilename  
initialFilename := init\_filename, fileFilter := file\_filter, \_  
filterIndex := filter\_index, title := title, \_  
buttonText := button\_text

**SAVE.TOOLBAR**(bar\_id, filename)

ExecuteExcel4Macro("SAVE.TOOLBAR(...)")

No direct equivalent.

**SAVE.WORKBOOK**(document\_text, type\_num, prot\_pwd, backup, write\_res\_pwd, read\_only\_rec)

ActiveWorkbook.SaveAs filename := document\_text, \_  
fileFormat := *type\_num*, password := prot\_pwd, \_  
writeResPassword := write\_res\_pwd, \_  
readOnlyRecommended := read\_only\_rec, \_  
createBackup := backup

**SAVE.WORKSPACE**(name\_text)

Application.Save filename := name\_text

**SCALE**(cross, cat\_labels, cat\_marks,  
between, max, reverse)

```
With ActiveChart.Axes(xlCategory, xlPrimary)
 If max = True Then
 .Crosses = xlMaximum
 Else
 .CrossesAt = cross
 End If
 .TickLabelSpacing = cat_labels
 .TickMarkSpacing = cat_marks
 .ReversePlotOrder = reverse
 .AxisBetweenCategories = between
End With
```

Syntax 1: Controls settings for the category axes on a 2-D chart. If the secondary axis is selected use xlSecondary instead of xlPrimary.

**SCALE**(min\_num, max\_num, major, minor,  
cross, logarithmic, reverse, max)

```
With ActiveChart.Axes(xlValue, xlPrimary)
 If min_num = True Then
 .MinimumScaleIsAuto = True
 Else
 .MinimumScale = min_num
 End If
 If max_num = True Then
 .MaximumScaleIsAuto = True
 Else
 .MaximumScale = max_num
 End If
 If major = True Then
 .MajorUnitIsAuto = True
 Else
 .MajorUnit = major
 End If
 If minor = True Then
 .MinorUnitIsAuto = True
 Else
 .MinorUnit = minor
 End If
 If max = True Then
 .Crosses = xlMaximum
 Else
 If cross = True Then
 .Crosses = xlAutomatic
 Else
 .Crosses = xlCustom
 .CrossesAt = cross
 End If
 End If
 .ScaleType = logarithmic
 .ReversePlotOrder = reverse
End With
```

Syntax 2: Controls settings for the value axes on a 2-D chart. If the secondary axis is selected use xlSecondary instead of xlPrimary.

**SCALE**(cat\_labels, cat\_marks, reverse,  
between)

```
With ActiveChart.Axes(xlCategory)
 .TickLabelSpacing = cat_labels
 .TickMarkSpacing = cat_marks
```

```

.ReversePlotOrder = reverse
.AxisBetweenCategories = between
End With

```

Syntax 3: Controls settings for the category axis on a 3-D chart.

```

SCALE(series_labels, series_marks, reverse)
With ActiveChart.Axes(xlSeries)
.TickLabelSpacing = series_labels
.TickMarkSpacing = series_marks
.ReversePlotOrder = reverse
End With

```

Syntax 4: Controls settings for the series axis on a 3-D chart.

```

SCALE(min_num, max_num, major, minor, cross, logarithmic, reverse, min)
With ActiveChart.Axes(xlValue)
If min_num = True Then
.MinimumScaleIsAuto = True
Else
.MinimumScale = min_num
End If
If max_num = True Then
.MaximumScaleIsAuto = True
Else
.MaximumScale = max_num
End If
If major = True Then
.MajorUnitIsAuto = True
Else
.MajorUnit = major
End If
If minor = True Then
.MinorUnitIsAuto = True
Else
.MinorUnit = minor
End If
If min = True Then
.Crosses = xlMinimum
Else
If cross = True Then
.Crosses = xlAutomatic
Else
.Crosses = xlCustom
.CrossesAt = cross
End If
End If
.ScaleType = logarithmic
.ReversePlotOrder = reverse
End With

```

Syntax 5: Controls settings for the value axis on a 3-D chart.

```

SCENARIO.ADD(scen_name, value_array, changing_ref, scen_comment, locked, hidden)
ActiveSheet.Scenarios.Add name := scen_name, _
changingCells := changing_ref, values := value_array, _
comment := scen_comment, locked := locked, _
hidden := hidden

```

```

SCENARIO.CELLS(changing_ref)
For Each scen In ActiveSheet.Scenarios
scen.ChangeScenario changingCells := changing_ref

```

Next

No direct equivalent. Scenarios in Microsoft Excel 5.0 can have different sets of changing cells; however, the SCENARIO.CELLS command assumes that there is only one set of changing cells. To simulate having only a single set of changing cells the example sets the changing cells of all scenarios by using a loop.

**SCENARIO.DELETE**(scen\_name)

ActiveSheet.Scenarios(scen\_name).Delete

**SCENARIO.EDIT**(scen\_name,  
new\_scenname, value\_array,  
changing\_ref, scen\_comment, locked,  
hidden)

```
ActiveSheet.Scenarios.ChangeScenario _
 changingCells := changing_ref, values := value_array
With ActiveSheet.Scenarios(scen_name)
 .Name = new_scenname
 .Comment = scen_comment
 .Locked = locked
 .Hidden = hidden
End With
```

**SCENARIO.GET**(1)

```
Dim result() As String
count = 0
For Each scen In ActiveSheet.Scenarios
 count = count + 1
 ReDim Preserve result(1 To count)
 result(count) = scen.Name
Next
```

Returns an array of the scenario names defined on the current sheet. Visual Basic uses the Scenarios collection for accessing scenarios, allowing direct iteration. The example builds an array of the scenario names by using the Scenarios collection.

**SCENARIO.GET**(2, scen\_name)

ActiveSheet.Scenarios(scen\_name).ChangingCells

**SCENARIO.GET**(3)

ExecuteExcel4Macro("SCENARIO.GET(3)")

No direct equivalent.

**SCENARIO.GET**(4, scen\_name)

ActiveSheet.Scenarios(scen\_name).Values

**SCENARIO.GET**(5, scen\_name)

ActiveSheet.Scenarios(scen\_name).Comment

**SCENARIO.GET**(6, scen\_name)

ActiveSheet.Scenarios(scen\_name).Locked

**SCENARIO.GET**(7, scen\_name)

ActiveSheet.Scenarios(scen\_name).Hidden

**SCENARIO.GET**(8, scen\_name)

ExecuteExcel4Macro("SCENARIO.GET(8, scen\_name)")

No direct equivalent

**SCENARIO.MERGE**(source\_file)

ActiveSheet.Scenarios.Merge source := source\_file

**SCENARIO.SHOW**(scen\_name)

ActiveSheet.Scenarios(scen\_name).Show

**SCENARIO.SHOW.NEXT**( )

ExcecuteExcel4Macro("SCENARIO.SHOW.NEXT()")

No direct equivalent. Microsoft Excel 5.0 does not have a next scenario button in the Scenarios dialog, so Visual Basic does not have a Next property. You can recreated the functionality of

SCENARIO.SHOW.NEXT in Visual Basic by showing a specific scenario and maintaining a counter to remember which scenario is currently visible. Show the next scenario by incrementing the counter.

**SCENARIO.SUMMARY**(result\_ref,  
report\_type)

ActiveSheet.Scenarios.CreateSummary \_  
reportType := *report\_type*, resultCells := *result\_ref*

**SCROLLBAR.PROPERTIES**(value, min,  
max, inc, page, link, 3d\_shading)

With Selection  
.Value = value  
.Min = min  
.Max = max  
.SmallChange = inc  
.LargeChange = page  
.LinkedCell = link  
.Display3DShading = 3d\_shading  
End With

**SELECT**(selection, active\_cell)

Range(*selection*).Select  
Range(*active\_cell*).Activate

Selects cell ranges or changes the active cell.

**SELECT**("Oval 1", replace)

ActiveSheet.DrawingObjects("Oval 1").Select replace := replace

Selects a single drawing object on a sheet or chart. Use ActiveChart instead of ActiveSheet if the object is on a chart.

**SELECT**("Oval 1, Rectangle 2", replace)

ActiveSheet.DrawingObjects( \_  
Array("Oval 1", "Rectangle 2")).Select replace := replace

Selects multiple drawing objects on a sheet or chart. Use ActiveChart instead of ActiveSheet if the object is on a chart.

**SELECT**(item\_text, single\_point)

Selects items on a chart. The syntax of charting selection varies widely. To see the Visual Basic equivalent of selecting a particular charting item, use the Record Macro command on the Tools menu .

**SELECT.ALL**( )

Set curSheet = ActiveSheet  
ActiveWorkbook.Sheets.Select  
curSheet.Activate

**SELECT.CHART**( )

ActiveChart.ChartArea.Select

**SELECT.END**(direction\_num)

Selection.End(*direction\_num*).Select

**SELECT.LAST.CELL**( )

ActiveCell.SpecialCells(xlLastCell).Select

**SELECT.LIST.ITEM**(index\_num)

Selection.Value = index\_num

Selects an item in a single-selection list box.

**SELECT.LIST.ITEM**(index\_num,  
selected\_logical)

Selection.Selected(index\_num) = selected\_logical

Selects, or cancels selection of, an item in a multiple-selection list box.

|                                                                 |                                                                                                                                                                                                                                                                                  |
|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>SELECT.PLOT.AREA</u></b> ( )                              | ActiveChart.PlotArea.Select                                                                                                                                                                                                                                                      |
| <b><u>SELECT.SPECIAL</u></b> (1)                                | Selection.SpecialCells(xlNotes).Select                                                                                                                                                                                                                                           |
| <b><u>SELECT.SPECIAL</u></b> (2, value_type)                    | Selection.SpecialCells(xlFormulas, value_type).Select                                                                                                                                                                                                                            |
| <b><u>SELECT.SPECIAL</u></b> (3, value_type)                    | Selection.SpecialCells(xlFormulas, value_type).Select                                                                                                                                                                                                                            |
| <b><u>SELECT.SPECIAL</u></b> (4)                                | Selection.SpecialCells(xlBlanks).Select                                                                                                                                                                                                                                          |
| <b><u>SELECT.SPECIAL</u></b> (5)                                | Selection.CurrentRegion.Select                                                                                                                                                                                                                                                   |
| <b><u>SELECT.SPECIAL</u></b> (6)                                | Selection.CurrentArray.Select                                                                                                                                                                                                                                                    |
| <b><u>SELECT.SPECIAL</u></b> (7)                                | Selection.RowDifferences(ActiveCell).Select                                                                                                                                                                                                                                      |
| <b><u>SELECT.SPECIAL</u></b> (8)                                | Selection.ColumnDifferences(ActiveCell).Select                                                                                                                                                                                                                                   |
| <b><u>SELECT.SPECIAL</u></b> (9, , 1)                           | Selection.DirectPrecedents.Select                                                                                                                                                                                                                                                |
| <b><u>SELECT.SPECIAL</u></b> (9, , 2)                           | Selection.Precedents.Select                                                                                                                                                                                                                                                      |
| <b><u>SELECT.SPECIAL</u></b> (10, , 1)                          | Selection.DirectDependents.Select                                                                                                                                                                                                                                                |
| <b><u>SELECT.SPECIAL</u></b> (10, , 2)                          | Selection.Dependents.Select                                                                                                                                                                                                                                                      |
| <b><u>SELECT.SPECIAL</u></b> (11)                               | Selection.SpecialCells(xlLastCell).Select                                                                                                                                                                                                                                        |
| <b><u>SELECT.SPECIAL</u></b> (12)                               | Selection.SpecialCells(xlVisible).Select                                                                                                                                                                                                                                         |
| <b><u>SELECT.SPECIAL</u></b> (13)                               | ActiveSheet.DrawingObjects.Select                                                                                                                                                                                                                                                |
| <b><u>SELECTION</u></b> ( )                                     | Selection                                                                                                                                                                                                                                                                        |
| <b><u>SEND.KEYS</u></b> (key_text, wait_logical)                | SendKeys key_text, wait_logical                                                                                                                                                                                                                                                  |
| <b><u>SEND.MAIL</u></b> (recipients, subject, return_receipt)   | ActiveWorkbook.SendMail recipients := recipients, _<br>subject := subject, returnReceipt := return_receipt                                                                                                                                                                       |
| <b><u>SEND.TO.BACK</u></b> ( )                                  | Selection.SendToBack                                                                                                                                                                                                                                                             |
| <b><u>SERIES</u></b> (name_ref, categories, values, plot_order) | No direct equivalent. The SERIES function represents the formula of a chart series and cannot be entered in a macro sheet cell. In Visual Basic use the Name, Values, XValues, and PlotOrder properties of the Series object, and the CategoryNames property of the Axis object. |
| <b><u>SERIES.AXES</u></b> (axis)                                | Selection.AxisGroup = axis                                                                                                                                                                                                                                                       |

**SERIES.ORDER**(chart\_num,  
old\_series\_num, new\_series\_num)

ActiveChart.ChartGroups(chart\_num).  
SeriesCollection(old\_series\_num).PlotOrder = \_  
new\_series\_num

**SERIES.X**(x\_ref)

Selection.XValues = x\_ref

**SERIES.Y**(name\_ref, y\_ref)

With Selection  
.Name = name\_ref  
.Values = y\_ref  
End With

**SET.CONTROL.VALUE**(value)

Selection.Value = value

**SET.CRITERIA**( )

ExecuteExcel4Macro("SET.CRITERIA()")

No direct equivalent. The AdvancedFilter method in Visual Basic allows the criteria to be contained in any arbitrary range. Explicitly setting the criteria range is not required.

**SET.DATABASE**( )

ExecuteExcel4Macro("SET.DATABASE()")

No direct equivalent. In Microsoft Excel 5.0 the database commands automatically detect and select database lists.

**SET.DIALOG.DEFAULT**(object\_id\_text)

ActiveDialog.DefaultButton = object\_id\_text

**SET.DIALOG.FOCUS**(object\_id\_text)

ActiveDialog.Focus = object\_id\_text

**SET.EXTRACT**( )

ExecuteExcel4Macro("SET.EXTRACT()")

No direct equivalent. The AdvancedFilter method in Visual Basic allows the extract to be contained in any arbitrary range. Explicitly setting the extract range is not required.

**SET.LIST.ITEM**(text, index\_num)

Selection.List(index\_num) = text

**SET.NAME**(name\_text, value)

Dim name\_text

name\_text = value

Defines a temporary name while a macro is executing. In Visual Basic use a variable instead of a name to hold a value. The example illustrates declaring a variable and assigning a value to it. Declaration of variables is not necessary unless Option Explicit is used for the Visual Basic module.

**SET.PAGE.BREAK**( )

ActiveCell.PageBreak = xlManual

**SET.PREFERRED**(format)

Application.SetDefaultChart formatName := format

Sets the default chart format to the name of a custom autofformat.

**SET.PREFERRED**("Built-in")

Application.SetDefaultChart formatName := xlBuiltIn

Sets the default chart format to the standard built-in format.

**SET.PRINT.AREA**( )

ActiveSheet.PageSetup.PrintArea = Selection.Address

Sets the print area to match the current selection.



**SET.PRINT.AREA**(range)

Sets the print area to the specified range.

ActiveSheet.PageSetup.PrintArea = *range*

**SET.PRINT.TITLES**(titles\_for\_columns\_ref, titles\_for\_rows\_ref)

With ActiveSheet.PageSetup  
     .PrintTitleColumns = titles\_for\_columns\_ref  
     .PrintTitleRows = titles\_for\_rows\_ref  
 End With

**SET.UPDATE.STATUS**(link\_text, status, type\_of\_link)

No direct equivalent.

ExecuteExcel4Macro("SET.UPDATE.STATUS(...)")

**SET.VALUE**(reference, values)

Changes the value of a name defined for the macro. In Visual Basic use a variable instead of a name to hold a value while a macro is executing. The example illustrates assigning a value to a variable.

myVariable = values

**SHORT.MENUS**(TRUE)

Displays short menus. Visual Basic does not have a direct equivalent. This command is used for compatibility with Microsoft Excel 3.0. The example displays the short worksheet menu bar.

MenuBar(xlWorksheetShort).Activate

**SHOW.ACTIVE.CELL**( )

ActiveCell.Activate

**SHOW.BAR**(bar\_num)

MenuBar(*bar\_num*).Activate

**SHOW.CLIPBOARD**( )

Application.DisplayClipboardWindow = True

**SHOW.DETAIL**(1, rowcol\_num, expand)

Shows or hides detail for an outline row.

Rows(rowcol\_num).ShowDetail = expand

**SHOW.DETAIL**(2, rowcol\_num, expand)

Shows or hides detail for an outline column.

Columns(rowcol\_num).ShowDetail = expand

**SHOW.DETAIL**(3, , expand, show\_field)

Shows or hides detail for a range in a pivot table.

ActiveCell.PivotTable.InnerDetail = show\_field  
 Selection.ShowDetail = expand

**SHOW.DIALOG**(dialog\_sheet)

DialogSheets(dialog\_sheet).Show

**SHOW.INFO**(logical)

Application.DisplayInfoWindow = logical

**SHOW.LEVELS**(row\_level, col\_level)

ActiveSheet.Outline.ShowLevels rowLevels := row\_level, \_  
     columnLevels = col\_level

**SHOW.TOOLBAR**(bar\_id, visible, dock, x\_pos, y\_pos, width, protect)

With Toolbars(bar\_id)  
     .Visible = visible  
     .Position = *dock*  
     .Left = x\_pos  
     .Top = y\_pos  
     .Width = width

End With

Sets attributes of a toolbar. The protect argument has no equivalent in Visual Basic.

**SHOW.TOOLBAR**(0, , , , , , tool\_tips,  
large\_buttons, color\_buttons)

With Application  
    .ShowToolTips = tool\_tips  
    .LargeButtons = large\_buttons  
    .ColorButtons = color\_buttons  
End With

Sets toolbar workspace properties.

**SORT**(orientation, key1, order1, key2,  
order2, key3, order3, header, custom,  
case)

Selection.Sort key1 := key1, order1 := *order1*, \_  
key2 := key2, order2 := *order2*, \_  
key3 := key3, order3 := *order3*, \_  
header := *header*, orderCustom := custom, \_  
matchCase := case, orientation := *orientation*

Syntax 1: Sorts a range of cells.

**SORT**(orientation, key1, order1, type,  
custom)

Selection.Sort key1 := key1, order1 := *order1*, type := type \_  
orderCustom := custom, orientation := orientation

Syntax 2: Sorts a pivot table.

**SOUND.NOTE**(cell\_ref, FALSE)

Range(cell\_ref).SoundNote.Record

Records a sound by displaying the sound recording dialog box.

**SOUND.NOTE**(cell\_ref, TRUE)

Range(cell\_ref).SoundNote.Delete

Deletes a sound note.

**SOUND.NOTE**(cell\_ref, file\_text, resource)

Range(cell\_ref).SoundNote.Import filename := file\_text \_  
resource := resource

Imports a sound note from a file. The resource argument is used Microsoft Excel for the Macintosh only.

**SOUND.PLAY**(cell\_ref)

Range(*cell\_ref*).SoundNote.Play

Plays a sound note contained in a cell. If cell\_ref is omitted, use ActiveCell instead of Range(cell\_ref).

**SOUND.PLAY**(, file\_text, resource)

ExecuteExcel4Macro("SOUND.PLAY(, *file\_text*, *resource*)")

Plays a sound contained in a file. Visual Basic has no direct equivalent.

**SPELLING**(custom\_dic, ignore\_uppercase,  
always\_suggest)

Selection.CheckSpelling customDictionary := custom\_dic, \_  
ignoreUppercase := ignore\_uppercase, \_  
alwaysSuggest := always\_suggest

**SPELLING.CHECK**(word\_text,  
custom\_dic, ignore\_uppercase)

Application.CheckSpelling word := word\_text, \_  
customDictionary := custom\_dic, \_  
ignoreUppercase := ignore\_uppercase

**SPLIT**(col\_split, row\_split)

With ActiveWindow  
    .SplitColumn = col\_split  
    .SplitRow = row\_split  
End With

**STANDARD.FONT**(name\_text, size\_num,  
bold, italic, underline, strike, color, outline,  
shadow)

This function is used for macro compatability with previous versions of Microsoft Excel. See the [DEFINE.STYLE](#) and [APPLY.STYLE](#) functions.

**STANDARD.WIDTH**(standard\_num)

ActiveSheet.StandardWidth = standard\_num

**STEP( )**

Stop

**STYLE**(bold, italic)

This function is used for macro compatability with previous versions of Microsoft Excel. See the FONT.PROPERTIES function.

**SUBSCRIBE.TO**(file\_text, format\_num)

Selection.SubscribeTo edition := file\_text, format := format\_num

**SUBTOTAL.CREATE**(at\_change\_in,  
function\_num, total, replace, pagebreaks,  
summary\_below)

```
Selection.Subtotal groupBy := at_change_in, _
 function := function_num, totalList := total, _
 replace := replace, pageBreaks := pagebreaks, _
 summaryBelowData := summary_below
```

**SUBTOTAL.REMOVE()**

Selection.RemoveSubtotal

**SUMMARY.INFO**(title, subject, author, keywords, comments)

```
With ActiveWorkbook
 .Title = title
 .Subject = subject
 .Author = author
 .Keywords = keywords
 .Comments = comments
End With
```

.....

Y  
Z

## T

Dialogs(xlDialogTabOrder).Show

No direct equivalent. Use the `Show` method of the `Dialogs` collection to bring up the Tab Order dialog, or use the `BringToFront` and `SendToBack` methods to control tab order of controls on dialogs.

Selection.Table rowInput := *row\_ref*, columnInput := *column\_ref*

DDETerminate channel := channel\_num

```
Selection.Characters(start_num, num_chars).Text := add_text
```

Replaces text in the selected text box with add\_text.

```
ActiveSheet.DrawingObjects(object_id_text). _
 Characters(start_num, num_chars).Text = add_text
```

Replaces text in a named text box or button with add text.

```
Selection.TextToColumns destination := destination_ref, _
 dataType := data_type, textQualifier := text_delim, _
 consecutiveDelimiter := consecutive_delim, _
 tab := tab, semicolon := semicolon, comma := comma, _
 space := space, other := other_char, _
 otherChar := other_char, fieldInfo := field_info
```

Range(text)

Converts A1 style text into a reference. In Visual Basic the Range method always takes A1 style text and returns a Range object.

**TEXTREF**(text, FALSE)

```
a1Text = Application.ConvertFormula(formula := "=" & text, _
 fromReferenceStyle := xlR1C1, _
 toReferenceStyle := xlA1, toAbsolute := xlAbsolute)
Set result = Range(Right(a1Text, Len(a1Text) - 1))
```

No direct equivalent. Converts R1C1 style text into a reference. In Visual Basic the Range method takes A1-style text only. The example translates R1C1-style text to A1-style text by treating the text as a formula.

R1C1-style text is often used in Microsoft Excel 4.0 Macros to construct references from row and column indexes. In Visual Basic use the Cells method, which constructs a Range directly from numeric row and column index parameters.

**TRACER.CLEAR**( )

```
ActiveSheet.ClearArrows
```

**TRACER.DISPLAY**(FALSE, FALSE)

```
Selection.ShowDependents remove := True
```

Removes one level of dependent tracer arrows

**TRACER.DISPLAY**(FALSE, TRUE)

```
Selection.ShowDependents remove := False
```

Adds one level of dependent tracer arrows

**TRACER.DISPLAY**(TRUE, FALSE)

```
Selection.ShowPrecedents remove := True
```

Removes one level of precedent tracer arrows

**TRACER.DISPLAY**(TRUE, TRUE)

```
Selection.ShowPrecedents remove := False
```

Adds one level of precedent tracer arrows

**TRACER.ERROR**( )

```
Selection.ShowErrors.Select
```

**TRACER.NAVIGATE**(direction,  
arrow\_num, ref\_num)

```
ActiveCell.NavigateArrow towardPrecedent := direction, _
 arrowNumber := arrow_num, linkNumber := ref_num
```

## U

**UNDO**( )

```
Application.Undo
```

**UNGROUP**( )

```
Selection.Ungroup
```

**UNHIDE**(window\_text)

```
Windows(window_text).Visible = False
```

Unhides a normal workbook window.

**UNHIDE**("[Book1]Sheet1 Chart1")

```
ActiveSheet.ChartObjects("Chart 1").Activate
```

Activates an embedded chart for editing.

**UNLOCKED.NEXT**( )

```
ActiveCell.Next.Activate
```

**UNLOCKED.PREV**( )

```
ActiveCell.Previous.Activate
```

**UNREGISTER**(register\_id)

```
ExecuteExcel4Macro("UNREGISTER(register_id)")
```

No direct equivalent. In Visual Basic, functions in DLLs and code resources are explicitly declared before use. At run time Visual Basic loads, registers, and unregisters the functions automatically. For more information on calling functions in DLLs, see Chapter 10, "Controlling and Communicating with Other Applications," in the Microsoft Excel Visual Basic User's Guide.

|                                                     |                                                                               |
|-----------------------------------------------------|-------------------------------------------------------------------------------|
| <b><u>UPDATE.LINK</u></b> (link_text, type_of_link) | ActiveWorkbook.UpdateLink name := link_text, _<br>type := <i>type_of_link</i> |
|-----------------------------------------------------|-------------------------------------------------------------------------------|

## V

|                                  |                                      |
|----------------------------------|--------------------------------------|
| <b><u>VBA.MAKE.ADDIN</u></b> ( ) | ExecuteExcel4Macro("VBA.MAKE.ADDIN") |
| No direct equivalent.            |                                      |

|                                               |                                                  |
|-----------------------------------------------|--------------------------------------------------|
| <b><u>VBA.INSERT.FILE</u></b> (filename_text) | ActiveSheet.InsertFile filename := filename_text |
|-----------------------------------------------|--------------------------------------------------|

|                                                                                    |                                                                                                                                                                                              |
|------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b><u>VIEW.3D</u></b> (elevation, perspective, rotation, axes, height%, autoscale) | With ActiveChart<br>.Elevation = elevation<br>.Perspective = autoscale<br>.Rotation = rotation<br>.RightAngleAxes = axes<br>.HeightPercent = height%<br>.AutoScaling = autoscale<br>End With |
|------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                |                                           |
|--------------------------------|-------------------------------------------|
| <b><u>VLINE</u></b> (num_rows) | ActiveWindow.SmallScroll down := num_rows |
|--------------------------------|-------------------------------------------|

|                                  |                                          |
|----------------------------------|------------------------------------------|
| <b><u>VOLATILE</u></b> (logical) | Application.Volatile volatile := logical |
|----------------------------------|------------------------------------------|

|                                   |                                              |
|-----------------------------------|----------------------------------------------|
| <b><u>VPAGE</u></b> (num_windows) | ActiveWindow.LargeScroll down := num_windows |
|-----------------------------------|----------------------------------------------|

|                                              |                                   |
|----------------------------------------------|-----------------------------------|
| <b><u>VSCROLL</u></b> (position, TRUE)       | ActiveWindow.ScrollRow = position |
| Scrolls vertically to a specific row number. |                                   |

|                                                                                                                                                               |                                           |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------|
| <b><u>VSCROLL</u></b> (position, FALSE)                                                                                                                       | ActiveWindow.ScrollRow = position * 16384 |
| Scrolls vertically to a fractional position. In Visual Basic you can scroll to specific rows only. The example converts a fraction to an absolute row number. |                                           |

## W

|                                    |                                        |
|------------------------------------|----------------------------------------|
| <b><u>WAIT</u></b> (serial_number) | Application.Wait time := serial_number |
|------------------------------------|----------------------------------------|

|                                    |                                                           |
|------------------------------------|-----------------------------------------------------------|
| <b><u>WHILE</u></b> (logical_test) | Do While logical_test<br>' statements within loop<br>Loop |
|------------------------------------|-----------------------------------------------------------|

In Visual Basic use a Do...Loop statement to build a loop. The Do...Loop statement offers more flexibility than the WHILE command.

|                                       |                                        |
|---------------------------------------|----------------------------------------|
| <b><u>WINDOW.MAXIMIZE</u></b> ( )     | ActiveWindow.WindowState = xlMaximized |
| Maximizes the active document window. |                                        |

**WINDOW.MAXIMIZE**(window\_text)      Windows(window\_text).WindowState = xlMaximized

Maximizes the specified window. For all of the following WINDOW functions that have a `window_text` argument, to manipulate a particular window use `Windows(window_text)` instead of `ActiveWindow`.

**WINDOW.MINIMIZE()**      ActiveWindow.WindowState = xlMinimized

```
WINDOW.MOVE(x_pos, y_pos) With ActiveForm
 .Left = x_pos
 .Top = y_pos
 End With
```

**WINDOW.RESTORE**( )      ActiveWindow.WindowState = xlNormal

```
WINDOW.SIZE(width, height) With ActiveForm
 .Width = width
 .Height = height
 End With
```

|                                     |                             |
|-------------------------------------|-----------------------------|
| <b><u>WINDOW.TITLE</u></b> (text)   | ActiveWindow.Caption = text |
| Changes the title text of a window. |                             |

|                                                                                 |                                     |
|---------------------------------------------------------------------------------|-------------------------------------|
| <b><u>WINDOW.TITLE</u></b> ( )                                                  | ExecuteExcel4Macro "WINDOW.TITLE()" |
| Restores the title text of a window to the default title. No direct equivalent. |                                     |

```
WINDOWS() For Each w In Windows
 MsgBox w.Caption
Next
```

Returns an array of window names. In Visual Basic the Windows collection provides direct access to Window objects and does not require using text names. The example displays the name of each window in a message box.

```
WINDOWS(, match_text)
 Dim result() As String
 numWindows = 0
 For Each w In Windows
 If w.Caption Like match_text Then
 numWindows = numWindows + 1
 ReDim Preserve result(1 To numWindows)
 result(numWindows) = w.Caption
 End If
 Next
```

Returns an array of window names with titles matching a pattern. The example constructs an array of window names that match the pattern. Note the use of the Windows collection in a For Each...Next statement.

**WINDOWS**(2, match\_text)  
No equivalent. In Visual Basic you cannot access the windows of loaded add-in workbooks.

**WINDOWS**(3, match\_text)  
No exact equivalent. In Visual Basic you cannot access the windows of loaded add-in workbooks. The windows of other workbooks are accessible using the Windows collection.

**WORKBOOK.ACTIVATE**(sheet\_name)      ActiveWorkbook.Sheets(sheet\_name).Activate

**WORKBOOK.ADD**(name\_array,  
dest\_book, position\_num)

Equivalent to WORKBOOK.MOVE in Microsoft Excel 5.0. See WORKBOOK.MOVE for Visual Basic equivalents.

**WORKBOOK.COPY**(name\_array)      Sheets(name\_array).Copy

Copies the specified sheets into a new workbook.

**WORKBOOK.COPY**(name\_array,      Sheets(name\_array).Copy before := Sheets(position\_num)  
dest\_book, position\_num)

Copies the specified sheets and inserts the copies within the current workbook, assuming that dest\_book is the name of the current workbook.

**WORKBOOK.COPY**(name\_array,      Sheets(name\_array).Copy after := Sheets(Sheets.Count)  
dest\_book)

Copies the specified sheets and inserts the copies at the end of the current workbook, assuming that dest\_book is the name of the current workbook.

**WORKBOOK.COPY**(name\_array,      Sheets(name\_array).Copy before := \_  
dest\_book, position\_num)      Workbooks(dest\_book).Sheets(position\_num)

Copies the specified sheets and inserts the copies within a different workbook, assuming that dest\_book is not the name of the current workbook.

**WORKBOOK.COPY**(name\_array,      numSheets = Workbooks(dest\_book).Sheets.Count  
dest\_book)      Set LastInBook = Workbooks(dest\_book).Sheets(numSheets)  
Sheets(name\_array).Copy after := LastInBook

Copies the specified sheets and inserts the copies at the end of a different workbook, assuming that dest\_book is not the name of the current workbook.

**WORKBOOK.DELETE**( )      ActiveWindow.SelectedSheets.Delete

Deletes the selected sheets.

**WORKBOOK.DELETE**(sheet\_text)      Sheets(sheet\_text).Delete

Deletes the specified sheet.

**WORKBOOK.DELETE**({"Sheet1",      Sheets(Array("Sheet1", "Sheet2")).Delete  
"Sheet2"})

Deletes a set of sheets.

**WORKBOOK.HIDE**( )      ActiveWindow.SelectedSheets.Visible = False

Hides the selected sheets in the current workbook window.

**WORKBOOK.HIDE**(sheet\_text)      Sheets(sheet\_text).Visible = False

Hides the specified sheet.

**WORKBOOK.HIDE**({"Sheet1","Sheet2"})      Sheets(Array("Sheet1", "Sheet2")).Visible = False

Hides the set of sheets specified as an array.



**WORKBOOK.HIDE**(sheet\_text, TRUE)      Sheets(sheet\_text).Visible = xlVeryHidden  
Hides the specified sheet so that it is not displayed in the unhide list.

**WORKBOOK.INSERT**(1)      Worksheets.Add

**WORKBOOK.INSERT**(2)      Charts.Add

**WORKBOOK.INSERT**(3)      Worksheets.Add type := xlExcel4MacroSheet

**WORKBOOK.INSERT**(4)      Worksheets.Add type := xlExcel4IntlMacroSheet

**WORKBOOK.INSERT**(6)      Modules.Add

**WORKBOOK.INSERT**(7)      DialogSheets.Add

**WORKBOOK.MOVE**(name\_array)      Sheets(name\_array).Move  
Moves the specified sheets into a new workbook.

**WORKBOOK.MOVE**(name\_array,      Sheets(name\_array).Move before := Sheets(position\_num)  
dest\_book, position\_num)  
Moves the specified sheets within the current workbook, assuming that dest\_book is the name of the current workbook.

**WORKBOOK.MOVE**(name\_array,      Sheets(name\_array).Move after := Sheets(Sheets.Count)  
dest\_book)  
Moves the specified sheets to the end of the current workbook, assuming that dest\_book is the name of the current workbook.

**WORKBOOK.MOVE**(name\_array,      Sheets(name\_array).Move before := \_  
dest\_book, position\_num)      Workbooks(dest\_book).Sheets(position\_num)  
Moves the specified sheets to a different workbook, assuming that dest\_book is not the name of the current workbook.

**WORKBOOK.MOVE**(name\_array,      numSheets = Workbooks(dest\_book).Sheets.Count  
dest\_book)      Set LastInBook = Workbooks(dest\_book).Sheets(numSheets)  
      Sheets(name\_array).Move after := LastInBook  
Moves the specified sheets to the end of a different workbook, assuming that dest\_book is not the name of the current workbook.

**WORKBOOK.NAME**(oldname\_text,      Sheets((oldname\_text, newname\_text)).Name = (oldname\_text,  
newname\_text)      newname\_text)

**WORKBOOK.NEW**( )  
No direct equivalent. This function adds a new sheet to a workbook after prompting the user for the type. To add a particular type of sheet, use the Add method of the Sheets object.

**WORKBOOK.NEXT**( )      ActiveSheet.Next.Select

**WORKBOOK.OPTIONS**(sheet\_name, , Sheets(sheet\_name).Name = new\_name  
new\_name)

The second argument of WORKBOOK.OPTIONS has no equivalent in Microsoft Excel 5.0.

**WORKBOOK.PREV**( ) ActiveSheet.Previous.Select

**WORKBOOK.PROTECT**(structure, ActiveWorkbook.Protect password := password, \_  
windows, password) structure := structure, windows := windows

Protects a workbook if either structure or windows are TRUE.

**WORKBOOK.PROTECT**(FALSE, FALSE, ActiveWorkbook.Unprotect password := password  
password)

Unprotects a workbook.

**WORKBOOK.SCROLL**(num\_sheets) ActiveWindow.ScrollWorkbookTabs sheets := num\_sheets

Scrolls the workbook tabs.

**WORKBOOK.SCROLL**(, TRUE) ActiveWindow.ScrollWorkbookTabs position := xlLast

Scrolls the workbook tabs to the last tab in the workbook.

**WORKBOOK.SCROLL**(, FALSE) ActiveWindow.ScrollWorkbookTabs position := xlFirst

Scrolls the workbook tabs to the first tab in the workbook.

**WORKBOOK.SELECT**("Sheet1", Sheets("Sheet1").Select  
"Sheet1")

Selects a single sheet in the active workbook. This is the most common use of WORKBOOK.SELECT.

**WORKBOOK.SELECT**({"Sheet1", Sheets(Array("Sheet1", "Sheet2")).Select  
"Sheet2"}, "Sheet2") Sheets("Sheet2").Activate

Selects a set of sheets in the active workbook and makes Sheet2 the active sheet in the group selection. In Visual Basic this action requires first selecting the sheets and then specifying which sheet should be active.

**WORKBOOK.SELECT**("Sheet1", Sheets("Sheet1").Select replace := False  
"Sheet1", FALSE)

Adds a single sheet to the sheet selection in a workbook. With the Sheets.Select method the replace argument works exactly like it does for the WORKBOOK.SELECT function.

**WORKBOOK.TAB.SPLIT**(ratio\_num) ActiveWindow.TabRatio = (ratio\_num)

**WORKBOOK.UNHIDE**(sheet\_text) Sheets(sheet\_text).Visible = True

**WORKGROUP**({"Sheet1", "Sheet2"}) Sheets(Array("Sheet1", "Sheet2")).Select

Selects the specified sheets as part of a group edit. Visual Basic has no direct equivalent for this command when the sheet array is omitted.

**WORKSPACE**(fixed, decimals, r1c1, scroll, With Application  
status, formula, menu\_key, remote, .FixedDecimal = fixed)

entermove, underlines, tools, notes,  
nav\_keys, menu\_key\_action, drag\_drop,  
show\_info, incell\_edit)

```
.FixedDecimalPlaces = decimals
.ReferenceStyle = r1c1
.DisplayStatusBar = status
.DisplayFormulaBar = formula
.TransitionMenuKey = menu_key_action
.IgnoreRemoteRequests = remote
.MoveAfterReturn = entermove
.CommandUnderlines = underlines
.DisplayNoteIndicator = notes
.TransitionNavigationKeys = nav_keys
.TransitionMenuKeyAction = menu_key_action
.CellDragAndDrop = drag_drop
.DisplayInfoWindow = show_info
.EditDirectlyInCell = incell_edit
End With
With ActiveWindow
 .DisplayHorizontalScrollBar = scroll
 .DisplayVerticalScrollBar = scroll
End With
```

The tools argument has no direct equivalent in Microsoft Excel 5.0.

**X**

**Y**

**Z**

**ZOOM**(TRUE)

ActiveWindow.Zoom = xlZoomToSelection

**ZOOM**(FALSE)

ActiveWindow.Zoom = 100

**ZOOM**(magnification)

ActiveWindow.Zoom = magnification

