4th Dimension 6.5

Upgrade
Windows®/Mac™OS

4th Dimension 6.5
Upgrade

IMPORTANT LICENSE INFORMATION

Use of this Software is subject to the ACI Product Line License Agreement, which is provided in electronic form with the Software. Please read the ACI Product Line License Agreement carefully before completely installing or using the Software.

Contents

1 Introduction

Welcome to version 6.5 of 4th Dimension
This new version of 4th Dimension and 4D Server brings many new features. Each of the 4D environments are affected by the changes: the Design Mode, the User Mode, the Language, as well as the built-in Web Server.

About this manual

This manual describes the new features and the modifications that were implemented in version 6.5 of 4th Dimension and 4D Server. This manual consists of the following chapters :

- **Design Mode** This chapter describes the new features and the modifications of the 4D Design Mode editors. This includes:
  - a new dialog for creating and opening a database,
  - some modifications to the Structure window and its use,
  - some modifications to the Form Editor,
  - a new Runtime Explorer,
  - a new Find Editor,
  - the capacity to write and edit comments,
  - some new features related to the edition of methods,
  - new database properties,
  - some modifications to the Password Editor,
  - a new Picture Library.
- **User Mode** This chapter describes the new features and the modifications of the 4D User Mode editors, which includes:
  - the redesign of the data import and export functions
  - the modification of the Label Editor,
  - some new options in the Method Execution dialog box.

- **Language** This chapter describes the new features in 4D's language as well as the modifications to the existing commands. This includes:
  - new and modified commands,
  - new form events,
  - enhancements of the Debugger.
- **Web Server** This chapter describes the new features and the improvements of the 4D built-in Web Server. This includes:
  - connection security,
  - customizing sessions,
  - information about the Web site,
  - HTML support
  - Non contextual mode.
- **Optimizations** This chapter describes the optimizations implemented in the 4D database engine, including the indexing processes and the sequential searches and sorts.

In addition, 4D version 6.5 also comes with new features added to the **Network components for 4D Server**, to **4D Tools** and **Customizer Plus**. These new features are described in separate documentations.

Compatibility with 4D 3.x.x/6.0.x databases

The 4D databases created with versions 3.x and 6.0.x of 4D are fully compatible with 4D 6.5.

- **4D 3.x.x Databases**: opening with version 6.5 a database that was created with version 3.x.x will display two successive alerts before 4D converts both the structure file and the data file to the 6.5 format. Once converted to 6.5, you will not be able to reopen the files with 4D version 3.
- **4D 6.0.x Databases**: The 4D databases created with version 6.0.x of 4D can be used directly with 4D 6.5, no conversion is necessary. However, you will not be able to reopen the structure file with 4D version 6.0.x. When you first use a 6.0.x structure file with 4D 6.5, an alert is displayed to make you aware of that restriction.
  On the other side, the data file can either be used by 4D 6.0.x or 4D 6.5 without any modification.

Note In the converted databases, the new features proposed by 4D 6.5 are set so as to keep the initial behavior of the database. These settings are described in this manual.

2 Design Mode

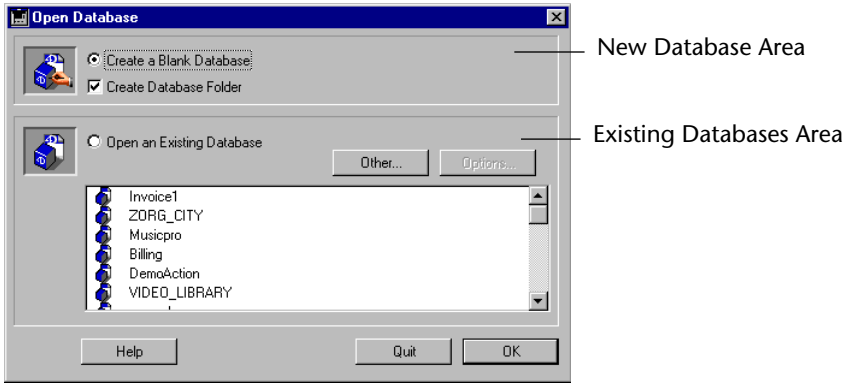
Numerous modifications and features have been added to 4D version 6.5's Design mode. These new features are described in the following sections:

- Creating and Opening Databases
- Structure Window's Modified Interface
- Forms
- Runtime Explorer
- Find Editor
- Comments
- Modifying Methods
- New Database Properties
- Passwords
- Picture Library
- Event Viewer (Windows NT).

Creating and Opening Databases

With version 6.5 of 4D, you create and open databases through the same dialog box :

New Dialog Box for Creating and Opening 4D Databases



This dialog box allows you to:

- create a blank database,
- open an existing database. Additional options are available that allow you to check the structure file or to select another data file.

Creating a Blank Database

► To create a blank database:

1 Select the Create a Blank Database radio button :

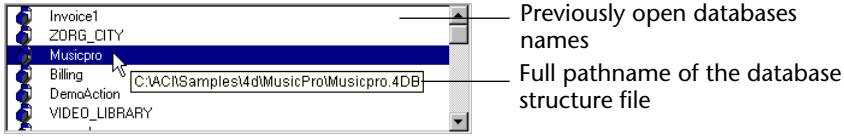
If you check the box **Create Database Folder**, all the database files will be saved into a new folder. The name of this folder will be identical to the database's name.

2 Click OK.

A standard save file dialog box appears. You can specify the name and the location of the new 4D structure file.
When you validate the dialog, the database is immediately created. Unlike earlier versions of 4D, version 6.5 creates both structure and data file at the same time.

Opening an Existing Database

4D now stores both the names and access pathes of all the previously opened databases. The names of those databases are displayed in the lower area of the Open database dialog box. The names are sorted in chronological order, the most recently opened databases being displayed at the top of the list.
The complete access path to the currently selected database structure file is displayed in a tip when you move the mouse cursor over the display area:



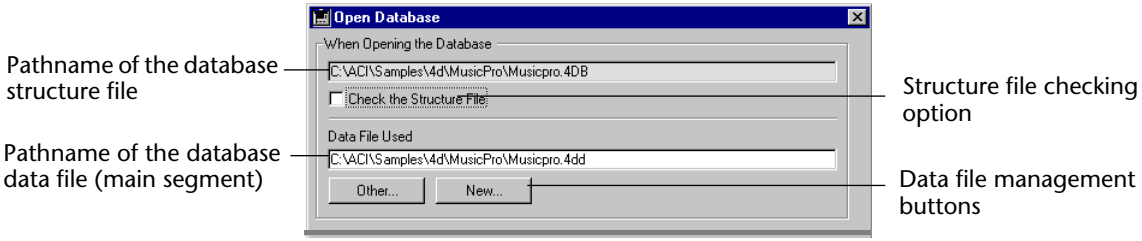
- To open a database from the list, double-click the database's name, or select the database's name and click **OK**.
- To open a database that is not already displayed in the list, click the **Other...** button. A standard Open file dialog box appears, allowing you to select the structure file you wish to open.
- To remove a database from the list, select the database's name and press the **Delete** or **Backspace** key.

Notes

• The list is updated by 4th Dimension. Attempting to open a database whose file have been deleted, moved or renamed at the OS level will result in the removal of the database from the list.
• For more information about how to open databases created with version 6.0.x or earlier of 4D, please refer to chapter "Introduction", page 9.

Options

The **Options...** button allows you to set additional options while opening a database. When you click this button, the following dialog box appears:



This dialog box displays the current name and access path of both the structure and data files of the database. If the data file is segmented, the information applies to the main segment.

- **Check the Structure File**: when this option is checked, the logical integrity of the structure file (tables, forms, menu bars, and so on) is checked when the database is launched. Once the check is performed, a "Check and Recover" window is displayed. That windows displays in the Report area the results of the tests performed.
  For more information about this option, please refer to the *4D Tools Reference Guide*.
- **Other...**: clicking on this button displays a standard Open file dialog box. You can then select a different data file that will be opened with the structure file of the database currently selected.
- **New...**: when you click on this button, a standard Save file dialog box is displayed. You can then create a new data file that will be opened with the structure file of the database currently selected.

Sharing Plug-ins

4D 6.5 allows you to share plug-in files between all your databases, by placing a Win4DX or a Mac4DX[1] folder in the ACI subfolder of your System folder[2]. The plug-ins located in this folder can then be used in all your 4D databases. This new feature makes both installation and upgrade of plug-ins used in several databases easier. It is available in 4th Dimension and 4D Server.

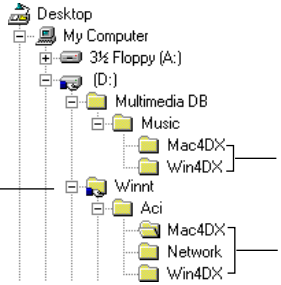
This installation principle is fully compatible with the current principle (Win4DX or Mac4DX folder located at the same level as the structure file). You can choose to install:

- in the Win4DX or Mac4DX folder of the ACI subfolder, the plug-ins that you want to be available in all your databases,
- in the Win4DX or Mac4DX folder located at the same level as the structure file of a database, the plug-ins that you want to be available in this database only.

---

1. These folders contain plug-ins files. You create a Win4DX or a Mac4DX folder depending on the platform on which the database is executed. For more information, please refer to the *4D Plug-ins and Tools Installation Guide*.
2. *On Windows*, the access path of this folder is C:\Windows\ACI (C:\Windows is the Windows System folder) ; *On MacOS*, the access path of this folder is HardDisk:System:Peferences:ACI (HardDisk:System is the System folder).

Example of installation on Windows NT



The Mac4DX folders are useful for a cross-platform exploitation of the database with 4D Server only

Folders containing plug-ins available in the Music database

Windows NT System folder

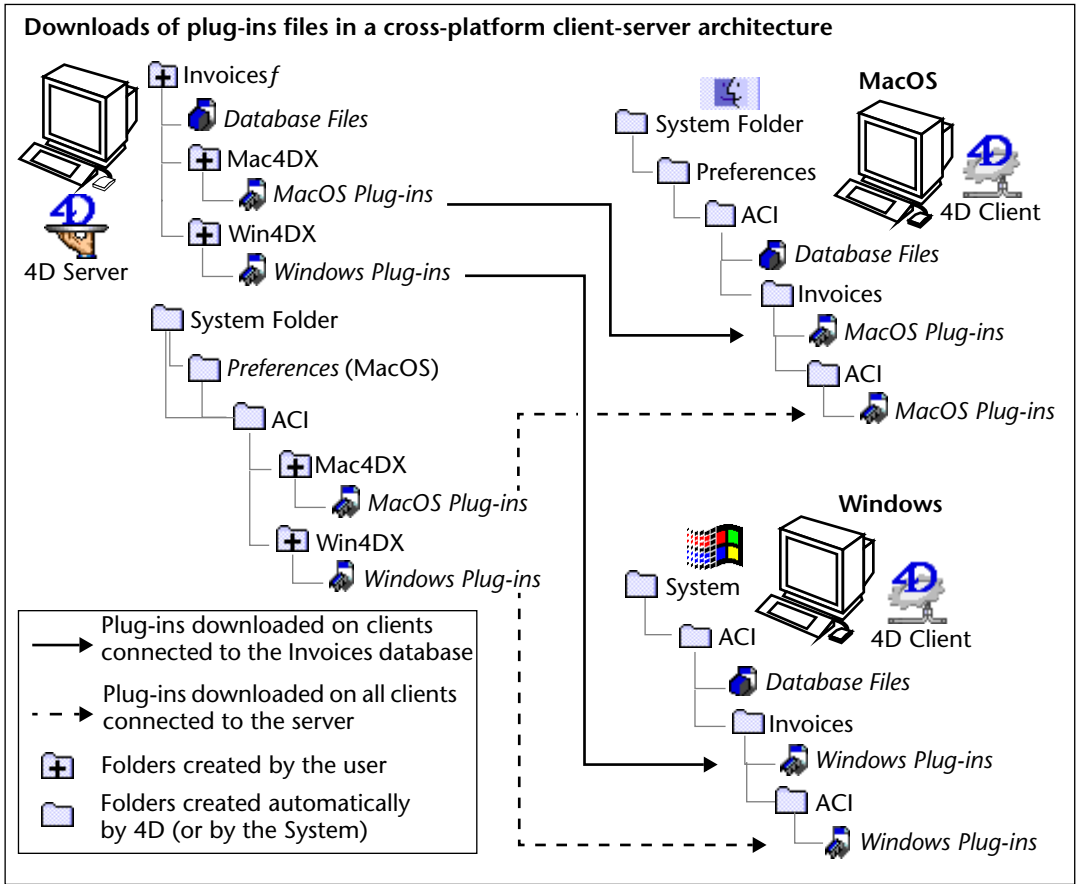
Folders containing plug-ins available in all the databases executed on this machine

Plug-ins located at the same level as the structure file have prority over those located in the ACI subfolder. This avoids any conflict between two copies of the same that would be placed in both locations. If this case occurs, only the plug-in located at the same level as the structure file is loaded.

Note This control works for ACI plug-ins only (internal ID <15000). It is impossible to detect conflicts with the plug-ins developped by Third parties. Consequently, be careful when you install third-party plug-ins.

Sharing Plug-ins with 4D Server

The contents of the Win4DX/Mac4DX folders located in the ACI subfolder of the server is automatically downloaded on every client, no matter which database it connects to. On each client machine, the plug-ins are stored in an ACI folder located in the DatabaseName folder, in the ACI subfolder of the System (cf. the following diagram).



Structure Window's Interface and Navigation

The interface of 4th Dimension's Structure window has been modified and therefore how you navigate in it has also been improved thanks to the following elements:

- contextual menus,
- keyboard shortcuts to navigate in the Structure window,
- a dialog box confirming the creation of a table,
- displaying the lengths of Alpha fields.

Contextual Menus

In numerous Design mode editors, you can now use contextual menus to execute specific actions on objects or to open dialog boxes.

► To use a contextual menu:

1 **On Windows, click on an object or area with the right mouse button. On MacOS, Control+click on an object or area.**

The contextual menu associated to the object or area appears.

Contextual menu associated to the fields in the Structure window

Note On MacOS, when you press the **Control** key, the mouse pointer appears as such indicating that a contextual menu has been activated.

2 **Select a command in the contextual menu, as you would for any other menu.**

Commands in contextual menus vary according to the current editor as well as the object on which you clicked. As its name indicates, a contextual menu only proposes actions related to a specific context. The actions that you can access are identical to those in the standard menus, buttons, and keyboard shortcuts.

Using the Keyboard to Navigate in the Structure Editor

You can navigate the tables and fields in the Structure window by using keyboard shortcuts.
Navigating between tables and fields when their properties palettes are displayed is also much easier now.

The following shortcuts are available:

Keys	Actions in the Structure Editor
	Tables and Subtables
[Tab]	• Selects the next table (or subtable) in the Structure window. • The selected table (or subtable) is centered in the Structure window and is brought to the front (see below)
	Fields and Subfields
[↓] *and* [↑]	The table's (or subtable's) field toward the top or bottom is selected
[Home] *and* [End]	The first or last field in the table (or subtable) is selected
[PgUp] *and* [PgDn]	The next or previous field "page" is displayed

Managing the Table Order

You can bring a table to the front, regardless of its creation order. You just need to select it either by clicking on it or by using the **Tab** key. The relative table order derives from the user's actions. This order is saved from one session to another and is specific to each user.

The table order determines how the tables are displayed in the Structure window (when they are superimposed), as well as how they are selected when you use the **Tab** key.

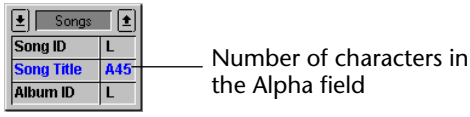
Creation of a Table

In the Structure Editor, when you create a new table (by using the **New table** item in the contextual menu or in the **Structure** menu, or by clicking on the corresponding button in the tool bar) you must now first accept the following confirmation dialog box:

Clicking the **Cancel** button allows you to leave the database structure untouched.

Displaying the Length of Alpha Fields

In the Structure window, the number of characters defined for Alpha fields is now displayed.



Note This information is not displayed if the option **Use Icons for Field Types** is selected, in the Design Environment page of the Database Properties window.

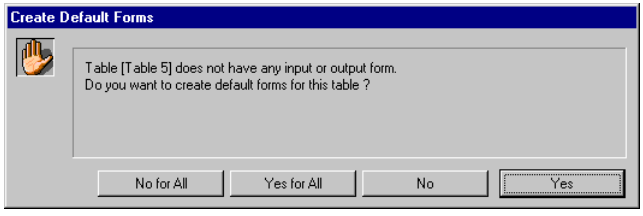
Forms

4D 6.5 offers new features and improvements when you create and modify forms. This section discusses the following subjects:

- **Configuring the Automatic Generation of Forms:** you can now configure the automatic creation of forms.
- **Modifications in the Form Editor:** the Form Editor's interface and tools have been modified in different areas (menus, palettes, display filters, markers...).
- **Object Properties List:** this new palette allows you to view and modify the object properties in a form as a dynamic list and is an alternative to the Object Properties palette.
- **Form Objects:** a new type of form object, called the splitter, is now available. Table and field names placed in a form can be dynamic. New possibilities are now available to you for existing objects: a transparent background for dynamic objects, a new mode for picture buttons, variables can be included in tips, etc.
- **List Form Headers:** active objects (buttons, check boxes...) can now be placed in the header area of list forms (output forms).
- **Contextual On-line Help:** you can now associate on-line help to a database as well as associate a "topic" to each form.

Creating Default Forms

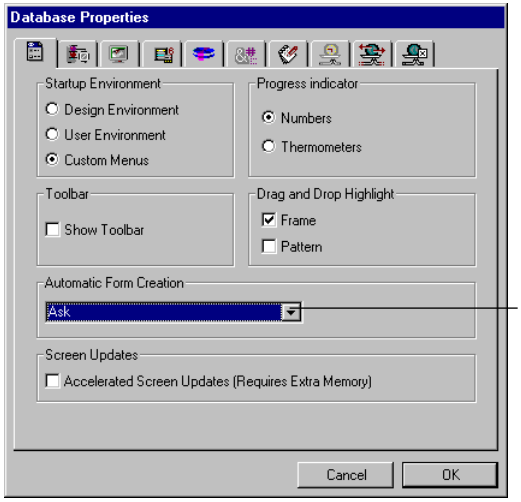
When you create a table in Design mode and go to the User mode, 4D tells you that no form has been created for the new table and offers to create a default input and output form :



You can modify this feature so that 4D doesn't display this alert dialog box.

► To configure the automatic creation of default forms:

1 In the Database Properties dialog box, click on the "General" tab :



The "Automatic Form Creation" menu offers three options:

- **Never**: the alert dialog box doesn't appear and no default form is created.
- **Ask**: the alert dialog box systematically appears (default option) when no form for the table has been created.
- **Always Yes for All**: the alert dialog box doesn't appear, default forms are automatically created for all the tables.

2 Select an option and accept the dialog box.

This setting will be applied to all the databases that you open with the same version of 4D.

Modifications to the Form Editor

The interface of 4D's Form Editor has been modified so that creating and modifying forms is easier and optimized:

- Contextual menus.
- The global selection shortcut has been modified.
- Associating a menu bar to a form has been modified.
- New options that allow you to show or hide the elements in the Form Editor.
- Marker labels.
- Displaying the form's limits.
- Accessing the form's method directly.
- Managing the palette's focus automatically.
- New tools palette.
- New commands to manage the form's pages.
- Choosing the form's destination.
- Alignment assistant.
- Matrix of objects duplication dialog box.

Contextual Menus

You can use contextual menus in 4D 6.5's Form Editor window. To activate them, click with the **right mouse button** (Windows) or **Control+click** (MacOS).

Note Refer to the paragraph "Contextual Menus", page 17.

Apart from the standard editing commands (copy, paste...) that are always available, the contents of the contextual menus in the Form Editor vary according to where you click:

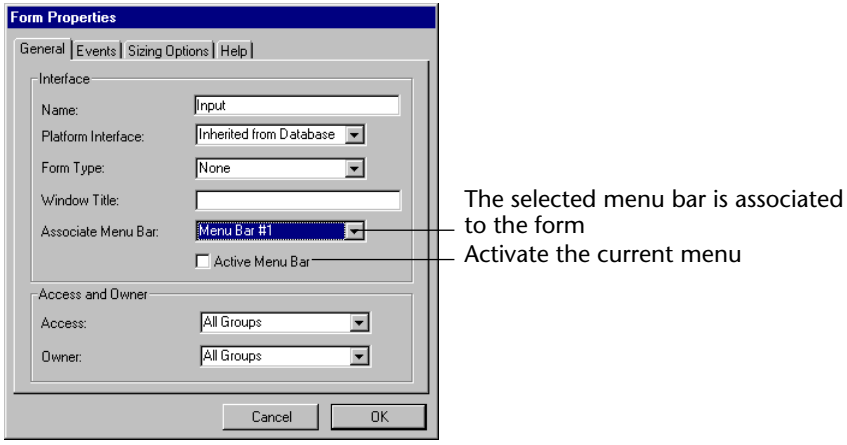
- If you click on a form object, the menu offers options and commands associated to the object: **Object properties**, **Object method**, **Color**, **Align**, **Borderline style**, **Automatic size**, **Automatic action** (according to the object) and **Level**. Additional commands can also be proposed depending on the object. For example, the **Attached Field** command appears if the object is a field, the **List** command appears if the object is a hierarchical list, etc.
- If you click in the Form editor's window without clicking on an object, the menu proposes options and commands associated to the form: **Form Properties**, **Form Method**, **Turn Grid On**, **Display** (see the paragraph "Showing/Hiding Elements in the Form Editor", page 22), and **Goto Page**.

Global Selection Shortcut (MacOS)

On MacOS, you activate the contextual menu by using the **Control+click** shortcut. This shortcut was previously used in the Form Editor to only select objects that were fully included in the selection rectangle.
This function is now accessible by using the **Option+click** shortcut (on MacOS) or the **Alt+click** shortcut (on Windows).

Associating a Menu Bar to a Form

The **Associate Menu Bar** menu item in the **Form** menu has been removed. This function is now a form property and is accessible from the new Property List or the Form Properties dialog box:



This menu contains all the menu bars that have been created in the database. You can then choose the menu bar that you want to associate to the form.

When the **Active Menu Bar** option is selected, 4D displays the associated menu bar along with the current menu bar in Custom Menus mode. In previous versions of 4D, this option was enabled by adding the – (minus) sign before the number of the associated menu bar. For more information, please refer to 4D's *Design* manual.

Showing/Hiding Elements in the Form Editor

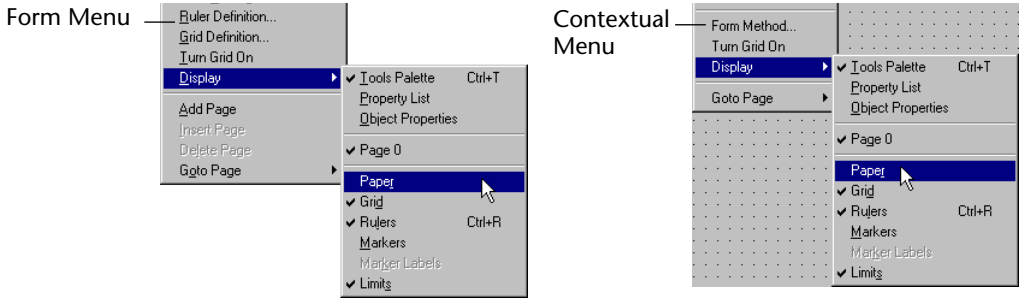
You can now hide or show most interface elements in the Form Editor. This feature allows you to show only the elements that you need to create or view in a form or only the tools that you want to use. This option is always applied to the Form Editor's current window.

► To show or hide an element in the Form Editor:

1 **Select Display from the Form menu.**
OR
Use the Display command in the contextual menu that appears in the Form Editor's window:

- On Windows, click with the **right mouse button** (without clicking on an object).
- On MacOS, **Control+click** (without clicking on an object).

A hierarchical submenu appears listing all the elements that you can show or hide:



A check mark placed next to the element indicates that it will be shown. To hide an element, select the element so that the check mark disappears.

2 **Select an element that you want to hide or show.**

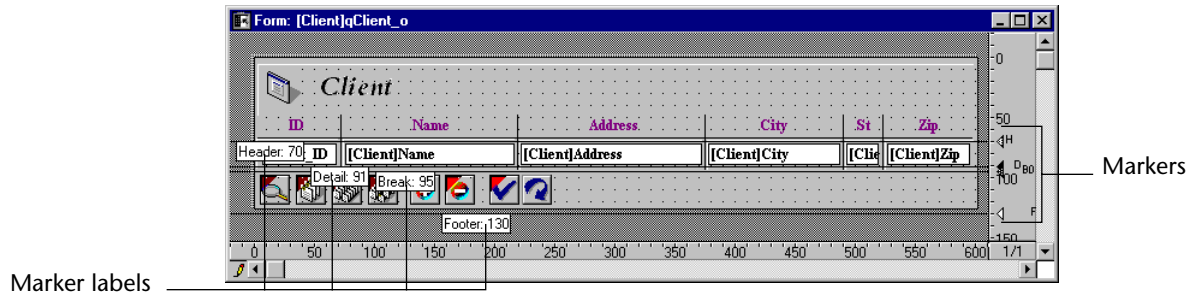
Here is a description of the commands in this menu:

- **Tools Palette**: shows or hides the Form Editor's tools palette. This palette has been modified in version 6.5, it regroups the tools and object palettes in previous versions of 4D (see paragraph "New Tools Palette", page 27).
- **Property List**: shows or hides the Property List. This list is a new feature in 4D version 6.5 and is described in the paragraph "Property List", page 33. The Property List is an alternative to the Object Properties palette. In previous versions, these two elements couldn't be displayed simultaneously. You can now toggle between the Property List and **Object Properties**; when one is selected, the other one is deselected.
- **Object Properties**: shows or hides the Object Properties palette. This palette can be replaced by the Property List (see above).

- **Page 0**: shows or hides the objects from page 0 in the form's current page. This option allows you to distinguish between the objects on the form's current page and those on page 0.
- **Paper**: shows or hides the borders of the printing page, which are shown as gray lines. This option can have no apparent effect when the **Limits** (see below) option is selected. If the size of the form is smaller than the printing page, the page's borders are shown outside of the form's viewing area and therefore do not appear.
- **Grid**: shows or hides the grid that is used as a guide when aligning objects. In previous versions of 4D, this grid was "invisible" and its appearance was related to the ruler unit that was chosen. In version 6.5, the grid that appears in the form's background is completely independent from the ruler; however, you can still use it to align objects on the form. You can define the grid's unit in the grid definition dialog box, which you can access by selecting **Grid Definition** in the **Form** menu.
- **Rulers**: shows or hides the rulers in the Form Editor's window as well as the pop-up menu that allows you to navigate the form's pages.
- **Markers**: shows or hides the markers that show the limits of the form's different areas.
- **Marker Labels**: shows or hides the marker labels, when they are displayed. For more information, please refer to the paragraph "Marker Labels", page 24.
- **Limits**: shows or hides the form's limits. When this option is selected, the form is displayed in the Form Editor as it appears in User mode. For more information, please refer to the paragraph "Displaying the Form's Limits", page 25.

Marker Labels

4D 6.5 allows you to display marker labels that delimit the form's areas. These labels contain the name and position of the marker.



This new feature offers the following advantages:

- markers as well as their positions can be clearly seen,
- a marker can now be easily moved by clicking on its label and dragging it.

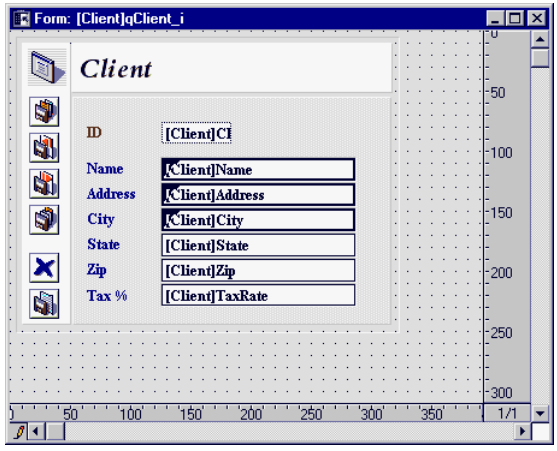
Note For marker labels to be displayed, markers themselves must also be displayed (see the paragraph “Showing/Hiding Elements in the Form Editor”, page 22). Marker labels automatically appear when you click on a marker to move it.

If you **Shift+click** on a marker label to move it, all the other markers below it will also be moved.

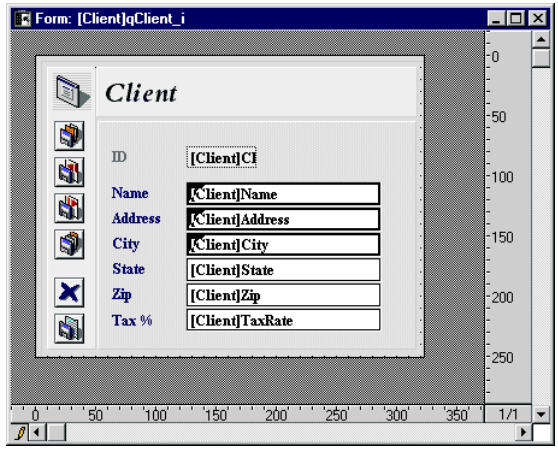
Displaying the Form’s Limits

This new option allows you to view the form’s limits, which means that you can see the form as it is displayed in User mode. This feature is especially interesting because you can modify a form without having to go to the User mode to see the results.
To view a form’s limits, you just need to select **Limits** from the **Form** menu (see the paragraph “Showing/Hiding Elements in the Form Editor”, page 22).

Limits not displayed (4D 6.0.x mode)

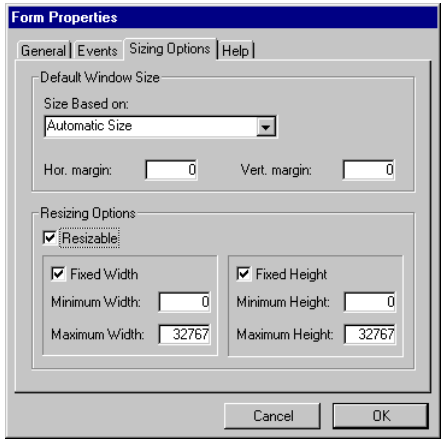


Limits displayed



The size of the form’s default window is calculated from the values defined on the “Sizing Options” page in Form Properties.

The **Size based on**, **Hor. Margin** and **Vert. Margin** options modify the form's limits.



Note You can display the Form Properties dialog box by using the contextual menu (**right mouse button** on Windows or **Control+click** on MacOS in the Form Editor's window without clicking on an object).

By default, the **Automatic size** option is selected, which means that the form's limits are calculated according to the objects on the form. When you move or resize an object placed near the form's limit, the size of the window is modified to accommodate the object.

Accessing a Form's Method Directly

You can now view or create a form's method directly from the Form Editor by using one of the three solutions below:

- Select **Form Method** from the **Form** menu.
- Display the Form Editor's contextual menu (**right mouse button** on Windows or **Control+click** on MacOS in the Form Editor's window without clicking on an object) and select **Form Method**.
- While the new **Property List** (see the paragraph "Property List", page 33) is open, click in the Form Editor's window without clicking on an object or select **Form: FormName** in the drop-down menu located at the top of the list. Click on the **Edit...** button, located next to the **Form Method** line, to open the form method.

Automatically Managing the Focus of the Palettes

Managing the focus between objects and floating windows is now automatic. The element that has the “focus” is the one on which the standard copy-paste commands and keyboard shortcuts are going to be executed.
In previous versions, and in particular in the Object Properties palette, you had to manage the focus manually (by using a switch). In 4D version 6.5, the switch has been taken away.

In version 6.5, the element (form editor object or floating palette) that has the focus is automatically the one in which the user has clicked last.

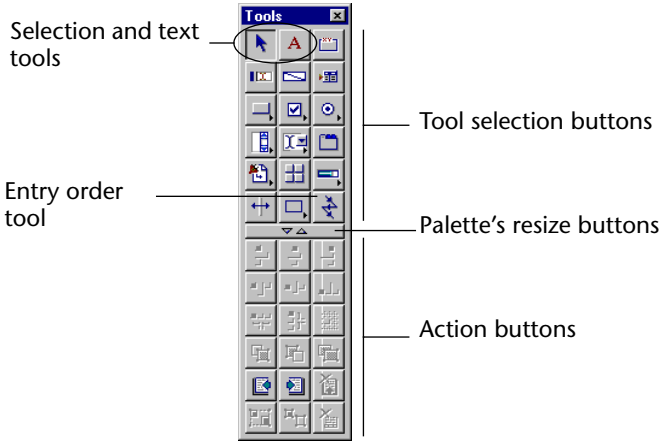
This principle applies to all of 4D’s editors that have floating palettes. However, if a palette does not contain a focusable object, it will never have the focus.

New Tools Palette

The object palette and 4D 6.0.x’s tools palette have been merged together to form a new Tools palette.

The new Tools palette’s functions are the same:

- The upper area of the palette regroups all the **tools**. You can create an object either by selecting a tool and drawing the object in form edition area, or by dragging and then dropping a Tool button in the form edition area. The object is then inserted into the form and is given the object’s default values and is automatically selected. The Object Properties palette (or the Property List) appears, if it wasn’t already being displayed.



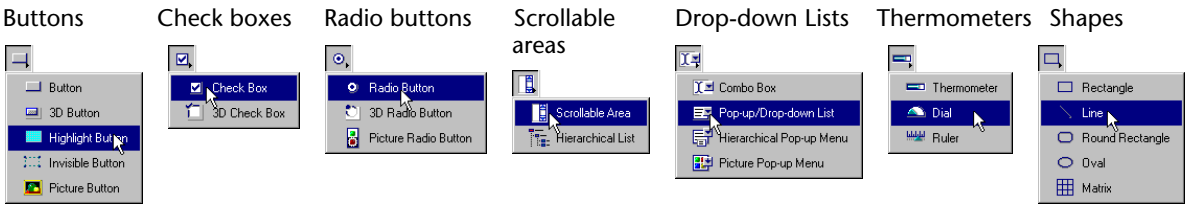
Some objects (buttons, radio buttons, menus...) are regrouped by type in the Tools palette. Each object type contains different variants that you can choose. The current variant displayed in the Tools palette is the object that will be inserted in the form, if it is dragged and dropped onto the form.
The objects that have variants have a small triangle in the lower right corner of the button.

▶ To select a variant for a certain type of object:

1 **Click on the object button and hold down the mouse button.**
OR
On Windows, click on the object button with the right mouse button.
On MacOS, Control+click on the object button.

You can select another object type from the menu that appears. The selected object type becomes the current type.

Here are the different variants proposed in the Tools palette:



Notes

- The **Plug-in Area** button can also offer alternatives according to the plug-ins that are installed in the database.
- The "Not enterable" variable type no longer exists as such. The "Enterable" option is now a property (accessible from the Object Properties Palettes or the Property List) for variable type objects.

■ The lower area of the palette regroups all the **actions**. You can hide this area by clicking the  button, which is located in the middle of the palette.

If the lower area is hidden, only the “tools” part of the palette is displayed:

To see the entire Tools palette, click on the  button again. The actions in the Tools palette are identical to those in previous versions of 4D. For more information, please refer to the *Design Reference*. However, six new actions have been added to the Tools palette:

New Page Management Commands

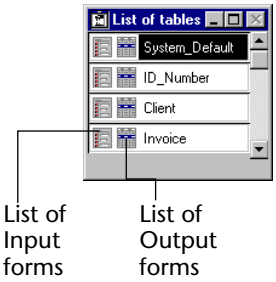
You can now access the form page management functions in the **Form** menu:

- **Add Page**: adds a page after the last page in the form.
- **Insert Page**: inserts a page before the current page.
- **Delete Page**: deletes the current page.
- **Goto Page**: goes to the page whose number is selected.

Note The **Goto Page** command can be accessed in two other ways:
• by using the pop-up menu located at the bottom right corner of the Form Editor’s window when the rulers are displayed.
• by using the Form Editor’s contextual menu (**right mouse button** on Windows or **Control+click** on MacOS, without clicking on an object).

Choosing a Form Type

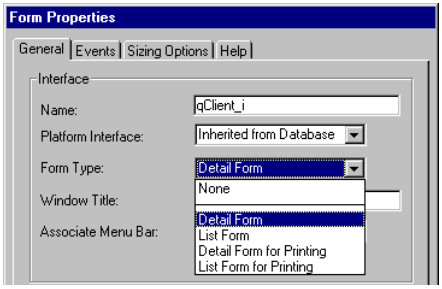
In 4D version 6.5, the form type (detail form, list form, detail form for printing, or list form for printing) can now be modified in the Form Properties dialog box.
In previous versions, this property could only be defined when you first created the form.



In 4D 6.5, this property allows you to restrict the number of forms displayed in the current Input and Output form selection lists (the **List of tables** window in the User mode). Only forms whose type corresponds to a list are displayed.

► To modify the form type:

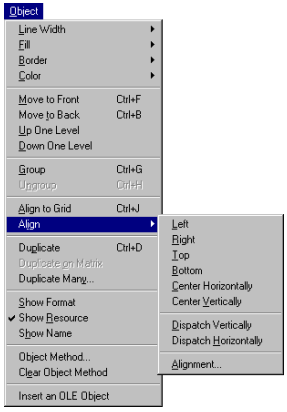
1 In the Form Properties dialog box, choose a value from the Form Type menu.



2 Accept the dialog box.

When the form type is “None”, it is displayed in both the Output and Input form menus in the List of Tables window. By default, the form type assigned to forms that come from databases created with an earlier version of 4D that are then opened with 4D 6.5 is “None”.

Alignment Assistant

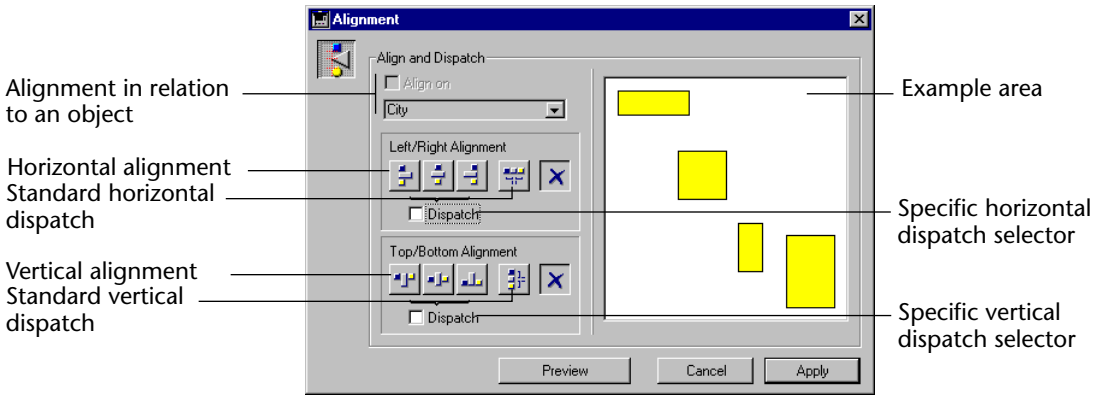


As in previous versions of 4D, you can still align and dispatch form objects by using the buttons in the tool palette. However, in 4D version 6.5, you can now use the **Align** submenu in the **Objects** menu and in the Form Editor’s contextual menu.

You can also use an assistant that allows you to align and dispatch the selected objects in different ways. This assistant allows you to align objects in relation to each other as well as to preview the result of your settings before applying them.

To display the alignment assistant, select the objects to align or to dispatch and choose **Alignment...** in the **Align** submenu.

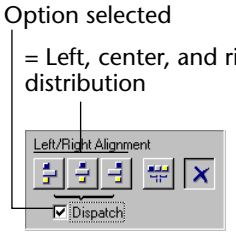
The following window appears:



- **Align**
  - To align the selected objects, click on the icon of your choice.
  - You can align a group of objects in relation to a particular object (in this case, the position of the reference object doesn't change). To do so, check the **Align on** option and select its name from the list.

- **Dispatch**
  - To dispatch the selected objects, click on the horizontal or vertical dispatch icon. The objects are then dispatched so that their sides are at an equal distance apart (standard dispatch).
  - You can also define specific dispatches. For example, you can dispatch objects so that their right sides are at an equal distance apart (and not their intervals). To do so, check the **Dispatch** option. Please note that this option acts as a selector. Once this option is checked, the icons above it are then applied to the dispatch.

    Horizontally, the icons correspond to the following dispatches: the **lefts**, **centers (vertical)** and **rights** of the selected objects are at an equal distance.
    Vertically, the icons correspond to the following dispatches: the **tops**, **centers (horizontal)** and **bottoms** of the selected objects are at an equal distance.



- The **example area** illustrates an example of the selected operation. You can also preview the result of the parameters you have chosen by clicking on the **Preview** button. The operation is executed in the Form Editor, but the dialog box remains in the front so that you can either **Apply** or **Cancel** the modifications.

Duplicating Objects on a Matrix

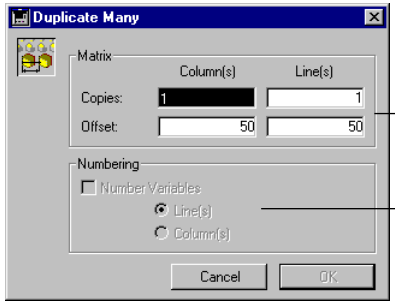
4D 6.5's Form Editor has a new dialog box that allows you to duplicate objects.
You can create as many copies of one object, or multiple objects, as you wish in one single operation. You can also define the offset between each duplicated object.
With this dialog box, you can also create a matrix of automatically numbered variables.

► To duplicate multiple objects:

1 **Select the object(s) to duplicate.**

2 **Select Duplicate Many... from Object menu.**

The "Duplicate Many" dialog box appears:



Defining the duplication matrix

Options to create a matrix of numbered variables (area is active only if a variable has been selected)

3 **In the upper area, enter the number of columns and lines of objects you want to get.**

- For example, if you want three columns and two lines of objects, enter 3 in the Column(s) area and 2 in the Line(s) area.
- If you want three horizontal new copies of an object, enter 4 in the Column(s) area and leave the default value, 1, in the Line(s) area.

4 **For lines and columns, define the offset that you wish to leave between each copy.**

The value must be expressed in points. It will be applied to r each copy, or copies, in relation to the original object.
For example, if you want to leave a vertical offset of 20 points between each object and the height of the source object is 50 points, enter 70 in the column's "Offset" area.

5 **If you wish to create a matrix of variables, select the "Number Variables" option (otherwise go to step 7).**

This option is only active if the selected object is a variable.

6 **Select the direction in which the variables are to be numbered, either by line(s) or by column(s).**

7 Click on the OK button.

The number of columns and lines of the specified object(s) is created.

Note The **Duplicate on Matrix** command also allows you to generate an array of automatically numbered variables integrated into a graphical matrix. This function existed in previous versions of 4D and was called **Objects on Grid**. For more information, refer to 4D's *Design* manual.

Property List

4D 6.5 offers an alternative to the Object Properties Palette: the **Property List**.

Property List and Object Properties Palette

These two tools globally offer the same features to manage form objects. However, the Property List differs in the following areas:

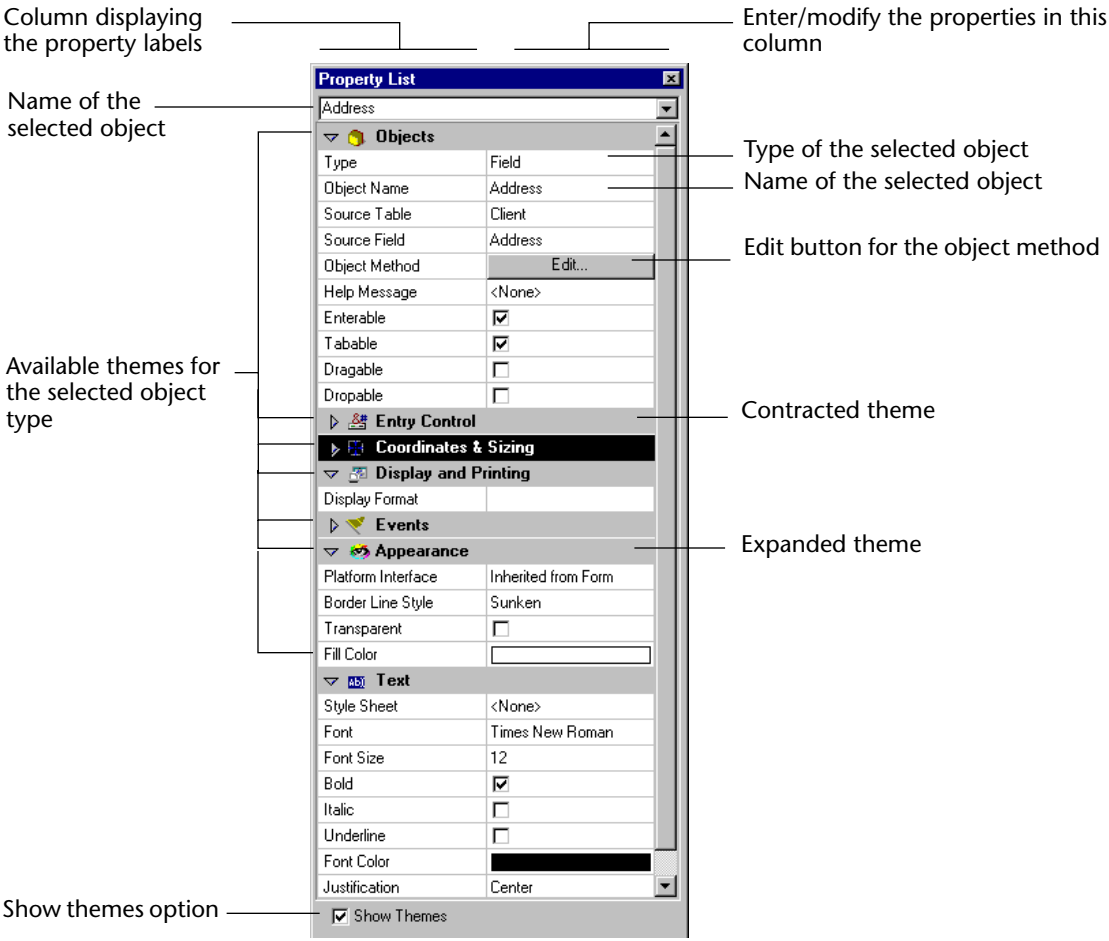
- The contents of the Property List are contextual: the options proposed are linked to the selected object. According to the object and its type, only the available themes and properties are displayed. Moreover, the contents of the list are modified as you change the settings. If, for example, you assign the Focusable property to a field, the Tabable property appears. If you define a variable as "Enterable", the Entry Control theme appears.
- The Property List allows you to display the current form's properties (see the paragraph "Showing the Form Properties in the Property List", page 36), which is not possible from the Object Properties Palette.
- The Property List allows you to modify the object type as long as it is in the same "family" (see the paragraph "Changing Object Types", page 36).
- The Property List allows you to give a type (alpha, numeric...) to the form's variables (see the paragraph "Typing Variables in a Form", page 37).
- The Property List doesn't allow you to modify style sheets or tips.

Note On the whole, the properties and their values are identical to those in previous versions of 4D. For more information, please refer to 4D's *Design* manual. In this manual, we only describe the new features in the Property List.

Displaying the Property List

You cannot display the two palettes simultaneously. However, you can choose which one to use depending on your needs and/or work habits. To display the Property List, select its name in the **Display** submenu in the **Form** menu, or in the **Display** submenu of the Form Editor's contextual menu (**right mouse button** on Windows, **Control+click** on MacOS in the Form Editor's window, without clicking on an object).

The Property List window is as follows:

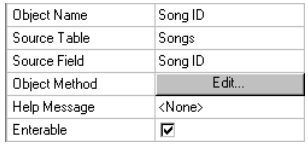


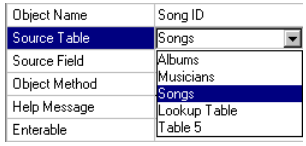
Note The logic of the two palettes is different: the tabs in the Object Properties Palette do not exactly correspond to the themes in the Property List.

Using the Property List

Here are the main Property List's features:

- You can resize the window by clicking on the lower right corner of the window.
- You can expand or contract each theme by clicking on the arrow to the left of its name. This feature allows you to display only the properties that you want to work with. You can also hide the theme names by checking them or by deselecting the **Show Themes** option. In this case, the properties appear as a continuous list in alphabetical order.
- If you select multiple objects, of the same type or of different types, only the common themes and/or properties will be displayed.
- The left column provides a list of the available properties and themes. The right column shows the current value for each property.
  To modify the current value of a property, you have to select it and define its new value.
  According to the property type that you want to modify, different entry modes are provided: combo boxes, check boxes, picture menus, buttons, drop-down menus, etc.

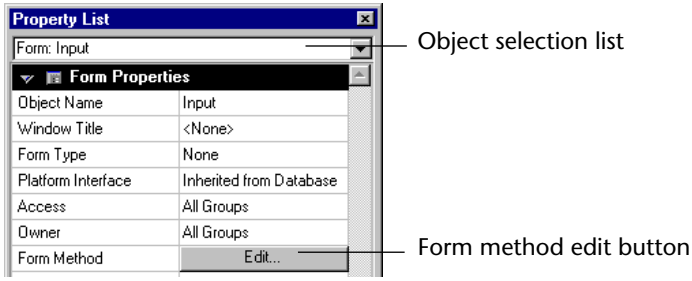




- You can use the following shortcuts in the Property List:
  - **Arrow keys**: To move from cell to cell.
  - **PgUp** and **PgDn**: To select the first and last cell visible in the displayed list.
  - **Home** and **End**: To select the first or last cell in the list.
  - **Ctrl+click** (Windows) or **Command+click** (MacOS) on an event: To select/deselect all the events according to the initial state of the event on which you clicked.
  - **Ctrl+click** (Windows) or **Command+click** (MacOS) on a theme name: To expand/contract all the themes.

Showing the Form Properties in the Property List

When you click in an empty area in the Form Editor window, the Property List displays the Form Properties. You can also display this list by selecting **Form: FormName** from the object selection list at the top of the Property List window:



The Property List window displays the form's properties as well as gives you access to the form's method.

Changing Object Types

The Property List allows you to transform every object type (active or not) into another object type. You can also transform a field into a variable and vice-versa.

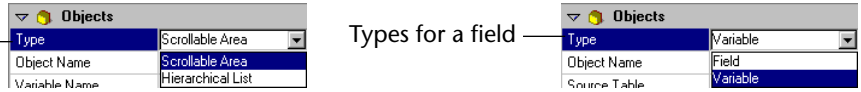
The Property List only allows you to make "coherent" type changes. Each object type belongs to a "family" and an object can only be changed into another object in the same family.

▶ To change an object type:

1 **Select the object in the form that you want to change.**

2 **In the Property List, pull down the Type menu.**

Only the object types of the same family are displayed in the list:



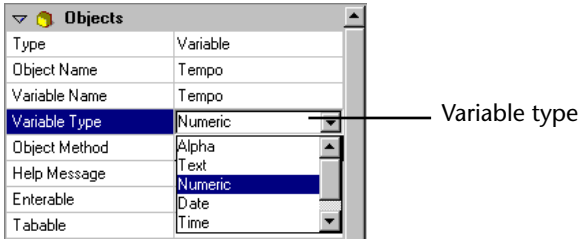
3 **Select the new object type.**

The list is then reorganized to display the properties of the new object type. The name of the object, its properties (enterable, size, color, etc.) as well as its method (if one exists) are retained.

Note When you change a variable into a field, 4D assigns the first field in the first table to the object by default. You can manually define the table and field in Source table and Source field.

Typing Variables in a Form

The Property List also introduces the notion of “typing” variables in a form.
To define the a variable’s type in a form, you must select a type in Variable Type:



By default, variables are of type “Alpha”. The Property List is then updated. For example, only the appropriate formats for the defined type are proposed in the list of “Display and Printing” formats, which makes it easier for you to find the proper format.

“Typing” is only used to contextual manage display formats. This typing is independent from 4D’s language and is not a substitute for declaring variables in methods.

Form Objects

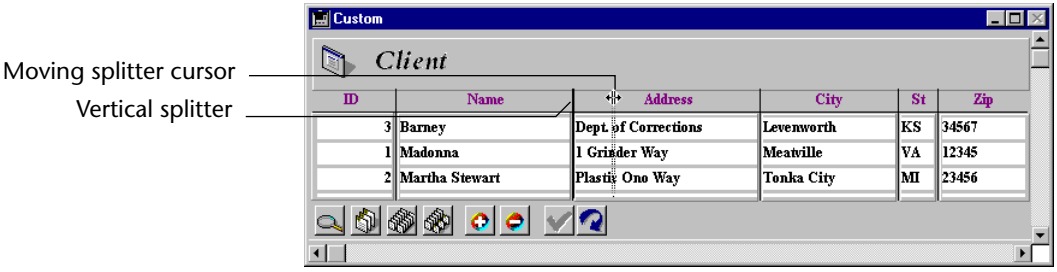
4D version 6.5 introduces numerous new features regarding form objects and how they are managed:

- New object: the **Splitter**.
- Inserting dynamic table and field names.
- Inserting dynamic references in Form windows name.
- Inserting dynamic references in tips.
- Creating lists/hierarchical menus by dragging and dropping.
- Managing the backgrounds of transparent objects.
- Color labels.
- Modifying the platform’s interface.
- Modifying the relationship between focusable and enterable.
- “Switch every N ticks” and “Roll over” modes for picture buttons.

Splitter

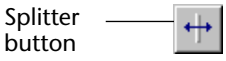
The **Splitter** is a new form object type in 4D 6.5. A splitter divides a form into two areas, allowing the user to enlarge and reduce the areas by moving the splitter one way or the other. A splitter can be either horizontal or vertical.
The splitter takes into account each object's resizing properties, which means that you can completely customize your database's interface.

The splitter is generally used in output forms (in Custom Menus mode) so that columns can be resized:



Here are some of the splitter's general characteristics:

- You can place as many splitters as you want in a form and use a mixture of horizontal and vertical splitters in the same form.
- A splitter can cut an object. This object will be resized when the splitter is moved.
- Splitters cannot be used in output forms in User mode and in sub-forms.
- If you resize a form by using a splitter, the new dimensions of the form are only saved while the form is being displayed. Once a form is closed, the initial dimensions are restored.



To insert a splitter in a form, use the Splitter button in the Object palette.
Once it is inserted, the splitter appears as a line. You can modify its border style to obtain a thinner line or change its color.

4 **Place the splitter on your form according to your needs.**

- **Interaction with the properties of neighboring objects**
  In a form, splitters interact with the objects that are around it according to these objects's resizing options:

Resizing options for the object(s)	Object(s) above the horizontal splitter or to the left of the vertical splitter[1]	Object(s) below the horizontal splitter or to the right of the vertical splitter[2]
None	Remain as is	Are moved with the splitter; their position relative to the splitter is not modified
Resize	Keep their original position, but they are resized according to the splitter's new position	
Move	Are moved with the splitter	

1. An object located in this position is used as a thrust when moving a horizontal splitter toward the top or a vertical splitter to the left, if no resizing option has been defined.
2. The buffer, when moving a horizontal splitter toward the bottom or a vertical splitter to the right is either the window's border or another splitter. This buffer is calculated in such a way that the moved objects remain entirely visible in the form or do not pass under/next to another splitter.

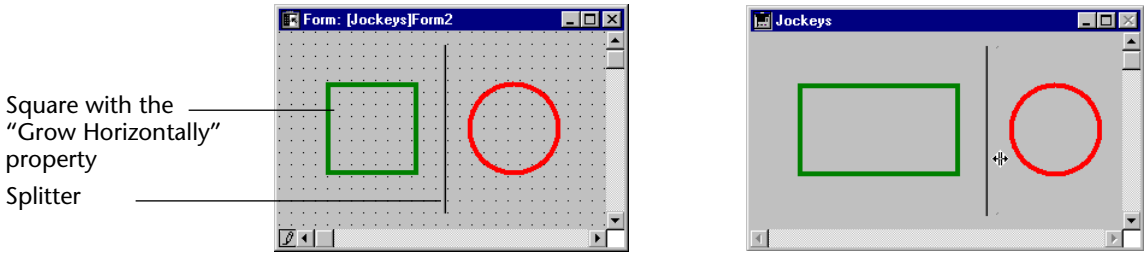
Note An object completely contained in the rectangle that defines the splitter is moved at the same time as the splitter.

- **Managing splitters procedurally**
  You can associate an object method to a splitter and it will be executed when the user releases the splitter.
  A variable of type Longint is associated to each splitter. This variable can be used in your object and/or form methods. Its value is equal to the splitter's current position, in pixels.
  - if the value is negative: the splitter was moved toward the top or toward the left,
  - if the value is positive: the splitter was moved toward the bottom or toward the right,
  - if the value is 0: the splitter was moved to its original position.

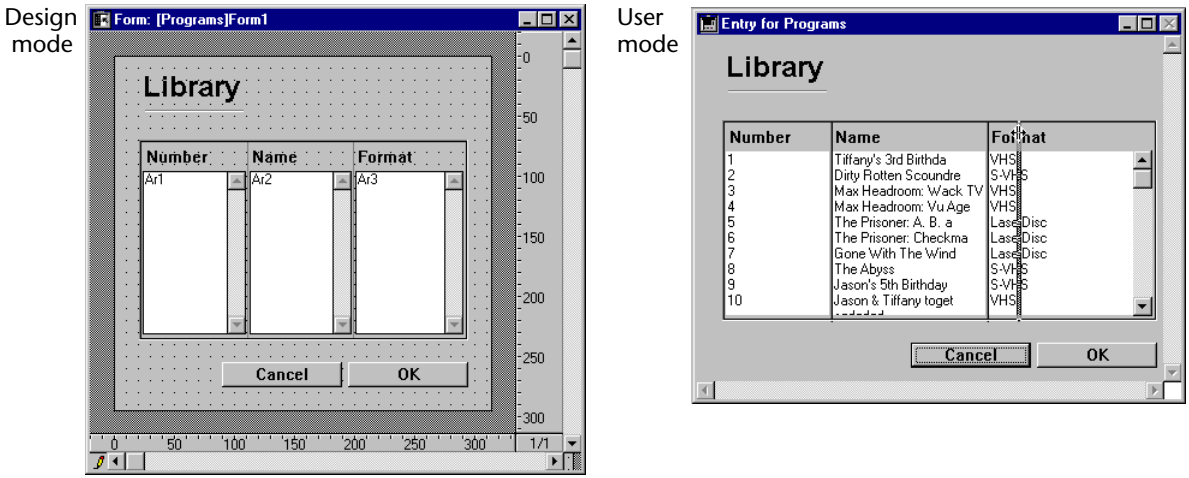
  You can also move the splitter programmatically : you just have to set the value of the associated variable. For example, if a vertical splitter is associated to a variable named *split1*, and if you execute the following statement : split1:=-10, the splitter will be moved from 10 pixels to the left — as if the user did it manually.
  The move is actually performed at the end of the execution of the form or object method containing the statement.

- **Automatic Splitter**
  This new automatic action allows you to create custom splitters in your forms. You can assign this action to an object of type **invisible button**. When an invisible button is assigned this automatic action, it acts exactly as a splitter. If, for example, you paste a picture on the invisible button, you can create a custom interface for your splitters.

- **Examples**
  - *Example 1*: An input form contains a square, a vertical splitter and a circle. The "Grow Horizontally" property is applied to the rectangle. When the splitter is moved to the right or to the left, the rectangle becomes bigger or smaller, the circle is moved and its size is not modified. If the window is resized, the objects do not change position or size.



  - *Example 2*: An input form containing three grouped arrays. The "Horizontal sizing" property is applied to all three arrays as well as their titles and a vertical splitter is placed between each column. In this way, you can modify each columns's relative size:



Inserting Dynamic Table and Field Names

You can now insert dynamic table and field names into your forms (as static text).

When you place dynamic table or field names in your forms, they are automatically updated throughout your database:

- either when you modify the table or field name in the Structure window
- or when the 4D commands Table name or Field name (in the "Structure Access" theme) are called.

This feature is particularly useful when you want to translate the table or field names in the forms on-the-fly.

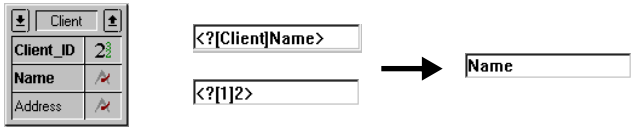
► To insert a dynamic table or field name in a form:

1 **In a static text area, enter the following reference:**

- to insert a dynamic table name: <**?[TableName]**> or <**?[2]**> (the table's creation order number, meaning the second table created).
- to insert a dynamic field name: <**?[TableName]FieldName**> or <**?[2]3**> (the table's and field's creation order number), or even <**?3**> (the field's creation order number) for the current table's field.

Please note that table and field numbers correspond to their creation order. You can therefore add or rename tables and fields without modifying the dynamic reference system.

2 **Click outside of the text area.**
The current field's or table's name appears as it has been defined in the Structure window.



Note You can view the "actual" contents of a static area in the Form Editor by clicking on the area or by selecting **Show Format** or **Show Name** from the **Objects** menu.

In the User and Custom Menus modes, a table's or field's name can be modified on-the-fly by using the Table name and Field name commands. In this case, the table and field name references will display the values defined by these commands.

Using dynamic field names is proposed on the **Options** page of the Form Wizard:

Dynamic References in Form windows name

You can now insert dynamic references in the form window's name. The current value of such a reference is set when the command INPUT FORM (with the * parameter) or Open form window is executed.

You define the name of a form window in the Property List (see paragraph "Showing the Form Properties in the Property List", page 36), or in the Form's Properties dialog box (you can access this dialog box through the **Form** menu or the Form Editor's contextual menu).

Note The name of a form window can contain up to 31 characters.

You can insert, in a form window name, the following references:

- a **STR# Resource Reference**: using the syntax ":16000,2" where 16000 is the resource ID and 2 the resource element.
- a **Dynamic Table or Field Name**: using the syntax <?[TableID]-FieldID or <?[TableName]FieldName>. For more information, please refer to the paragraph "Inserting Dynamic Table and Field Names", page 41.
- a **variable or a field**: using the syntax <VariableName> or <[Table]Field>. The current value of the variable or the field will be displayed in the window title.

Dynamic References in Tips

As of version 6.5, you can insert references to variables, fields, and field or table names in tips.
For example, if you enter the following text: "Enter <[Family]First>'s age in this area". In User mode, the current value of the [Family]First field is displayed in the reference's position.

Dragging and Dropping Lists

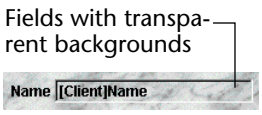
You can now create a hierarchical list or a hierarchical pop-up menu in a form by selecting a list from the List Editor or Explorer and dragging and dropping it on the form.

Note Lists can now contain up to 255 characters (instead of 31) in 4D 6.5.

- To insert a **Hierarchical List** object, drag the list from the List Editor or from the "Lists" page of the Explorer.
- To insert a **Hierarchical Pop-up Menu** object, drag the list from the List Editor or from the "Lists" page of the Explorer while holding down the **Shift** key.

Note Following the same principle, you can now create a button or a picture menu by dragging or dropping a picture from the Picture Library. For more information, please refer to the paragraph "Picture Library", page 77.

Managing the Backgrounds of Transparent Objects



Managing the transparent backgrounds of variables and fields has been improved.
In previous versions of 4D, selecting "automatic" as the background color allowed you to obtain a transparent effect. However, this color was actually the background color of the form.

You can now apply a real transparent background to variables and fields. To do so, you have to select the **Transparent** option in the Property List (using the Object Properties palette, select the **No fill pattern** (icon with the letter "N")). The variable or field is then displayed with a transparent background.

During data entry, a full background is automatically applied. This background color is defined in Object Properties. If this color is "automatic", the background color defined in the system is used.

Note You can now also assign foreground and background colors to **Pop-up/Drop down list** and **Combo box** objects.

Color Labels

In version 6.5, you can modify the color of a form object's label. All objects that have a label are concerned: buttons, check boxes, drop-down menus, scrollable areas, thermometers, etc.

To modify the color of an object's label, select the object then apply a color in one of the following ways:

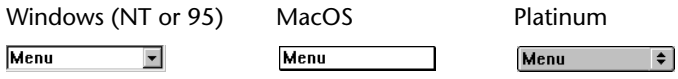
- in the **Objects** menu or the Form Editor's contextual menu, choose **Color**, then a color for the **Foreground**.
- in the Object Properties Palette, choose a color for **Foreground** in the **Colors** page.
- in the Property List, choose a color in the **Font Color** ("Text" theme) line.

Platform Interface

Managing the platform interface properties has been modified:

- The term "Copland" has now been replaced by "Platinum" (the MacOS 8 look) in the platform interface lists (Database Properties, Form Properties, etc.). On MacOS, when a dynamic text object (fields and variables) has the focus, a shadow frame is now displayed around the area.
- The appearance of hierarchical lists is now adapted to the interface defined for the database, form, or object. It also takes into consideration the parameters defined by the SET PLATFORM INTERFACE command, as long as the SET LIST PROPERTIES command is not called.
- **Pop-up menu**, **Drop down list** and **Menu/Drop down list** objects become a single object in 4D 6.5: **Pop-up/Drop down list**.
  Actually, these three names actually corresponded to the same object, even if its appearance may be different depending on the platform. Now simpler and more optimized, this new object is combined with the platform's interface property (for the database, form, or object).

  Therefore, when you insert a **Pop-up/Drop down list** in a form, its appearance will be different according to the platform:



  If the object's platform interface is **Automatic**, its appearance will be adjusted according to the platform interface defined for the form or database.

  The compatibility with previous versions of 4D is automatically ensured when you open a 6.0.x database with 4D version 6.5, objects of the three previous platform interface types are converted and their properties are adjusted so that their appearance is not modified.

Modifying Focusable/Enterable

The selection frame of enterable objects can be disabled by using the "Focusable" option. An enterable field or variable that is not "Focusable" remains non-enterable.

Picture Buttons: Two New Modes

Two new modes are available for picture buttons: "**Switch every N ticks**" and "**Switch when Cursor is over**".

- **Switch every N ticks**: this mode allows you to loop the contents of the picture button at a specified speed.
  - In the Object Properties Palette, you specify this mode by passing a fifth parameter to the picture button's syntax. This parameter activates the "Switch every N ticks" mode and supplies the value (in ticks) of the time interval that separates each picture.
    For example, if you pass: 2;3;?16807;0;10, the picture in the button will be looped every 10 ticks.
  - In the Property List, you have to enter a value in the "Switch every n ticks" line (**Display and Printing** theme).

  In this mode, all the other options are ignored— with the exception of the "Transparent Picture" option (mode 64 in the Object Properties Palette).
- **Switch when Cursor is over**: in this mode, the contents of the picture button are modified when the mouse cursor passes over it without the user clicking ("Roll over"). The initial picture is re-established when the cursor leaves the button's area. This mode is frequently used in Web navigators and in multimedia applications.
  The displayed picture is the last one in the set of frames, unless the "Use Last Frame as Disabled" option (mode 128 in the Object Properties palette) is also selected — in this case, it is the second to the last frame that is used for the roll over.
  - In the Object Properties Palette, you can specify this mode by passing **16** to the picture button's syntax.
    For example, if you want to define a complete picture button with the "action", "roll over" and "disabled" modes, you can define a set of frames containing one line and 4 columns. Each frame corresponds to the following states: action/released, action/clicked, roll over and disabled. You then pass 4;1;?15000;176 to the picture button's syntax.
  - In the Property List, you have to select the "Roll over" option (**Display and Printing** theme).

Headers of Output Forms

You can now place and use active objects in the headers of output forms (lists) displayed by using the DISPLAY SELECTION and MODIFY SELECTION commands.

Note This feature doesn't apply to lists displayed in User mode or to subforms.

Only active "clickable" and non-enterable objects are accepted:

- Button, Default Button, 3D Button, Highlight Button, Invisible Button, Picture Button,
- Pop-up/Drop-down list, Hierarchical Pop-up Menu, Picture Pop-up menu,
- Scrollable Area, Hierarchical list,
- Radio Button, 3D Radio Button, Picture Radio Button,
- Check Box, 3D Check Box,
- Thermometer, Ruler, Dial.

Note The Combo Box object is enterable and cannot be used.

Standard automatic actions like **Accept**, **Cancel** or **Automatic Splitter** can be associated to buttons.

All these objects react to the following form events:

- On Load (1),
- On Clicked (4),
- On Header (5),
- On Printing Footer (7),
- On Double Clicked (13),
- On Drop (16),
- On Drag Over (21),
- On Unload (24).

Note 4D 6.5 offers new form events that are described in the paragraph "Form Events", page 149.

Contextual On-line Help

4D version 6.5 allows you to associate a custom on-line help file to each 4D database. Moreover, you can associate a precise section of this help file to each of the database's forms, which allows you to provide contextual on-line help.

Creating a Custom On-line Help File

4D 6.5's custom on-line help system works as follows:

- The name of the help file must be identical to the database's structure file name. It must also have the ".HLP" or ".HTM" extension, depending on its format (see below) and platform.
- The help file must be placed in the database's folder or in the Win4DX and/or Mac4DX folder.
  With 4D Server, if you want the help file to be accessible to all the client workstations, place the file in the Win4DX and/or Mac4DX folder. It will then be automatically transferred to the client workstations.
- The help file can have one of the following types:
  - a ".HLP" file, which is the standard Microsoft Help[1] format for Windows. This type of file can be used on MacOS[2] provided the Microsoft Help for Mac application has been installed.
  - a file ".HTM", which is the HTML format. With this format you can create an identical on-line help on all platforms. The on-line help is then displayed in a Web browser.
  - an Apple Guide file (on MacOS only). This file can use *coachmarks* (allowing you to coach an object in the application window). If Apple Guide has not been installed, 4D will try to open the file in Microsoft Help format.

Note The Plug-ins can also have a help file, which must be placed in the Win4DX and/or Mac4DX folder for both single-user and client-server applications. The Plug-in's help file must be the name of the Plug-in with the .HLP or .HTM extension.

1. To know how to generate a .HLP file, refer to Microsoft's documentation.
2. You have to change the document type to HELP and the creator to MSHE. Successfully converting the document depends on the version of the on-line help compiler as well as the version of Microsoft Help being used. Otherwise, this solution doesn't allow you to use contextual on-line help.

Creating a Contextual On-line Help

Creating a contextual on-line help for 4D databases consists of associating a section number to each of your forms. When you call the on-line help from a form, the corresponding help page is displayed.

Contextual on-line help is available:

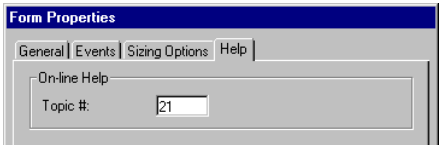
- on Windows, if the on-line help is in HLP format.
- on MacOS, if the on-line help is in Apple Guide format.
- on all platforms, if the on-line help is in HTML format and if the viewing browser is compatible with Internet Config 1.2.

► To associate a section number to a 4D form:

1 **In the Form Editor, select "Form Properties" from the "Form" menu.**
The Form Properties dialog box appears.

2 **Click on the *Help* tab.**
This property is also accessible in the Property List.

3 **Enter the section number you want to associate to the form.**



4 **Accept the dialog box.**

5 **In the help file, declare the same number for the section that concerns this form.**
This operation varies according to the format you are using:

- for HLP files, refer to the Microsoft on-line help compiler's documentation (Help Compiler).
- for Apple Guide files, refer to the Apple documentation.
- for HTML files, you must declare each section and assign a number to it.
  A section is declared by using a marker of type <A NAME="Section">. For example <A NAME="21">.
  The URL of the section has the following form <A HREF="#Section">...</A>. For example <A HREF="#21">...</A>

If the section number passed in the form is 0, 4D displays the first page of the help file.

Calling the On-line Help from a 4D Database

In 4D, you can call a database's custom on-line help in two ways:

- by selecting **DatabaseName Help** in the **Help** menu (in version 7 of MacOS, this menu presents itself as a question mark). In this case, the first page of the help file is displayed.
- by hitting the **F1** key (MacOS and Windows) or the **Help key** (MacOS only) when a form is displayed on screen. In this case, if a help field number has been associated to the form, the corresponding page is displayed (contextual help), otherwise the first page of the help file is displayed.

Note 4th Dimension has its own on-line help system (accessible through the **Help** menu — except for 4D applications integrated with a 4D Engine). On Windows, a contextual on-line help is available in the Methods Editor. When you select a command in the code and hit **F1**, the command's description appears.

Runtime Explorer

This new window allows you to view the behavior of the different structural elements in your database and to verify that the available resources are being correctly exploited. The Runtime Explorer is particularly useful in your database's development and analysis phase.

Displaying the Window

The Runtime Explorer window is accessible in all of 4th Dimension's modes: Design, User and Custom Menus (compiled or not).

Note In Custom Menus mode, only the Designer and Administrator have access to the Runtime Explorer window.

The Runtime Explorer can be displayed in two types of windows: in a "classical" window (only in Design mode) or in a floating palette (in all the modes). In this last case, the floating palette always remains displayed in the front.

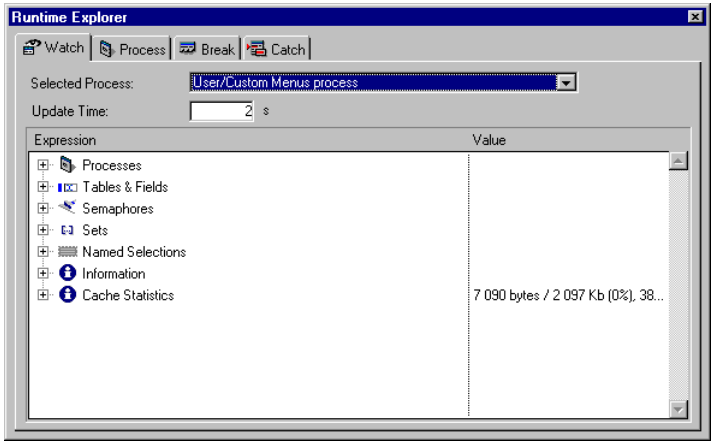
► To display the Runtime Explorer in a window (in Design mode):

1 Select Runtime Explorer from the Tools menu.

► To display the Runtime Explorer in a floating palette (from the Design, User or Custom Menus modes):

1 **On Windows, press Ctrl+Shift+F9.**
On MacOS, press Command+Shift+F9.
or
Hold down the Shift key and select Runtime Explorer from the Tools menu (Design mode only).

The Runtime Explorer window has four pages, that you can access by clicking on the following tabs: **Watch**, **Process**, **Break** and **Catch**.



Watch Page

The **Watch** page displays information about the execution of the code.

Note The information displayed in this page is identical to the information in 4D debugger's Watch pane.

- **Selected Process**: this drop-down list contains all the processes that are being executed in the database. It allows you to select the process(es) that you want to observe.
- **Update Time**: in this area, you can define a value (in seconds) that indicates how often the information in the page will be updated.

The "Expression" column displays the names of the objects and expressions. The "Value" column displays the current value corresponding to the objects and expressions. These columns can be resized, one in relation to another. To do so, click on the separation line and drag it laterally in the direction you want.

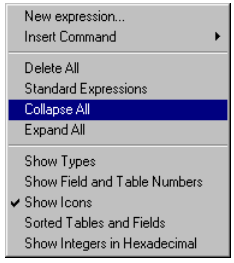
By clicking on a value in the right column, you can modify the object's value, if the object allows you to.

The multi-level hierarchical list is organized by multi-level themes. The themes are as follows:

- **Process**: allows you to view the list and the current status of the database's running processes.
- **Variables**: allows you to view the list of the database's interprocess variables as well as the list of the selected process's process variables.
- **Tables and fields**, **Semaphores**, **Sets**, **Named Selections**, **Information:** the information provided in these themes is identical to the information provided by 4D's **debugger**. For more information, refer to the "Debugger" chapter in 4D's Language manual. The **Tables & fields** theme contains additional information described in the paragraph "Debugger", page 151.
- **Cache statistics:** allows you to obtain information about 4D's cache. This new list is also available in 4D 6.5's debugger.

To delete an expression or a theme, select the corresponding line and press the **Delete** key.

You can also add a **New expression** or a 4D **Command**, or perform global actions: **Delete All**, display all the **Standard Expressions**, **Collapse All** or **Expand All**.
To do this, select the corresponding command in the contextual menu which appears when you click the **Right mouse button** (under Windows) or **Command+click** (under MacOS) in the window :



In addition, several displaying options are proposed in the lower part of the contextual menu. For more information, please refer to chapter "Debugger" in the *4D Language Reference* manual.

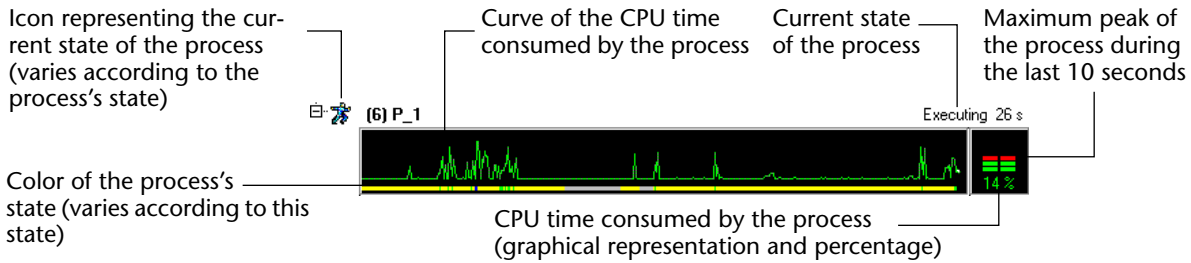
Process Page

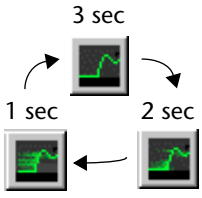
The **Process** page allows you to graphically view the CPU time consumed by each process as well as the state of each process.

Note This page replaces the **Process List** in previous versions of 4D. It has the advantage of being called from the User and Custom Menus modes as well as in compiled databases. Moreover, this page proposes new features.

You can show or hide the graphical display of each process by clicking on the expansion icon to the left of the process's name. You can also show or hide all the icons by clicking the **Show** and **Hide** buttons in the window's tool bar.

Here is a description of the information displayed:





You can modify the frequency at which data should be updated, which can be every one, two, or three seconds. To do so, click successively on the icon in the top right portion of the window. The greater the update time is, the more CPU time is consumed by the Runtime Explorer process. The number of processes to graphically represent on screen also influences the CPU time consumed by the process.

Note No CPU time is consumed for a process when its graphical representation is closed.

When you click in the graphical area, a vertical line appears where you clicked and a tip indicates the state of the process at that instant. By holding down the mouse button and moving it laterally, you can view the evolution of the process's state.

The process management commands are now accessible by using the tool bar's buttons in the window: **Resume** , **Pause** , **Abort** , **Trace** , **Hide** , **Show** , **Bring to Front** .

Break and Catch Pages

The **Break** page allows you to view and manage break points that you have placed in your database.
The **Catch** page displays the break points defined in the database in relation to commands (or expressions).

Note These pages replace the "Break points" and "Caught commands" parts of the **Break List** in previous versions of 4D. These pages have the advantage of being called from the User and Custom Menus modes, as well as in compiled databases.

The general functioning of these pages is identical to 4D's **Break List**. For more information, refer to the "Debugger" chapter in 4D's *Language* manual. The commands in the **Break List** menu are replaced by the buttons in the window's tool bar.

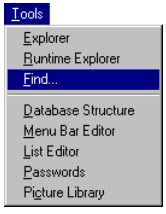
Note In 4D version 6.5, break points can now be defined directly in the 4D Methods Editor (refer to the paragraph "Inserting Break Points", page 68).

Find Editor

The Find Editor is a new feature in 4D 6.5, which came from 4D Insider. This editor allows you to search for a string in the entire database structure.

Searching in the Database

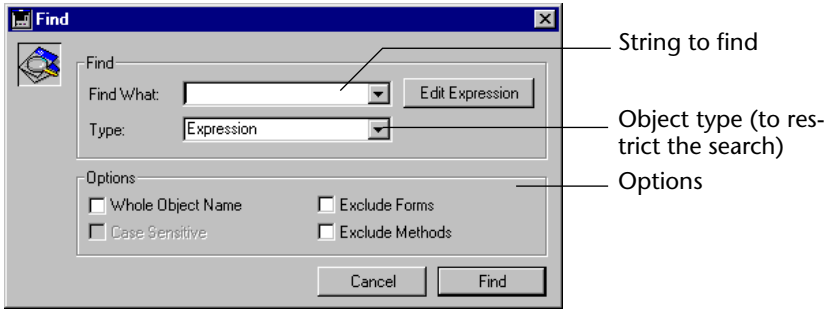
You can access 4th Dimension 6.5's Find Editor in the Design mode.



▶ To find a string:

1 In Design mode, select "Find..." from the Tools menu.

The Find Editor appears:



2 In the "Find What" area, enter a string to find.

Note The at-sign (@) is considered a character. It is not possible to use it as a "joker" when searching for a string in the database's structure.

3 Define (if necessary) the object type to which you want to restrict the search.

- Restricting the search to a certain object type accelerates the search. If you don't want the search to be limited to an object type, choose **All** in the **Type** menu.
- If you select **Expression** in the **Type** menu, the **Edit Expression** button becomes enabled. It allows you to define the expression to search directly in 4D's Formula Editor.

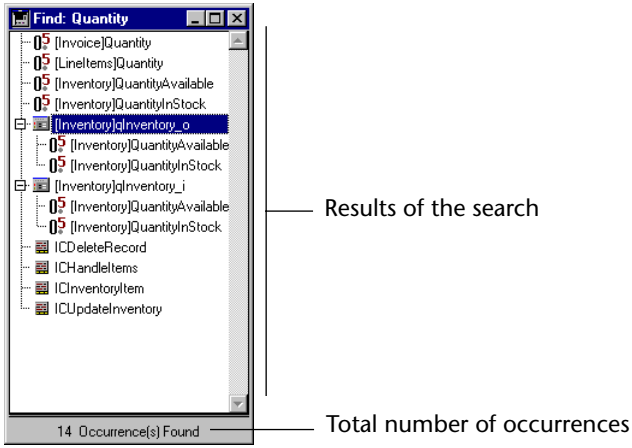
The object types are described in the paragraph "Object Types and Scope of the Search", page 55.

4 Define (if necessary) the searching options.

These options are described in the paragraph "Searching Options", page 56.

5 Click on the Find button or hit the Enter key.

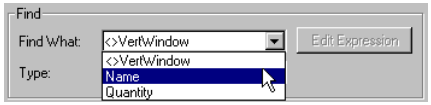
Once the search has finished, the occurrences found appear in a new window and are presented in a resizable hierarchical list.



You can double-click on a line in this window to view the object in its appropriate editor.

If you launch many searches successively, each search opens its own result window.

Once you have executed a search, the value entered in the Find What area is saved in memory. This value, as well as all the other values entered during the same session, can be selected from the combo box:



You can also quickly launch the same search many times.

Object Types and Scope of the Search

The Find Editor allows you to designate the object type of to search:

- an expression, for example “total:=Sum([Accounts]Total)”
- a variable, for example “$vpPicture1”
- a table or field name, for example “[Clients]Name”
- a form object name, for example “Background”
- a comment, for example “Modified on”
- you can also execute a global search among all object types.

By default, the string will be searched throughout the entire structure of the database. You can, however, exclude methods and/or forms from the search.

Depending on the designated object type, the search will be done among the following elements:

- forms (which can also be excluded),
- methods (which can also be excluded),
- menus and menu commands,
- lists,
- tables and fields (as well as subtables and subfields),
- comments.

The following table presents the structure elements in which you can search, depending on the different object types:

		Places in which to search					
		Forms and form names	Methods and method names	Menus/menu commands	Lists	Tables and fields (Structure window)	Comments
Object Types	Expression		X				
	Variable	X	X				
	Table or field name	X	X			X	
	Form object name	X	X				
	Comment						X
	All	X	X	X	X	X	X

Searching Options

You can select different options presented to you as check boxes. Depending on the type of search you execute, some options may be disabled:

- **Whole Object Name**
  When this option is selected, the search is limited to the exact occurrences of the searched object name or expression. In this case, for example, if you are searching for “client”, 4D will not find “clients” or “myclient”.
  By default, the option is not selected, which means that searching for “var” will find “Myvar”, “variation”, etc.

- **Case Sensitive**
  This option is selectable only if the option **Whole Object Name** is selected. When **Case Sensitive** is selected, the search takes into account the case of the characters as they have been entered in the Find Editor. Therefore, if you search for “MyVar”, 4D won’t find “myVar”.

- **Exclude Forms**
  When this option is selected, the search is done throughout the database, except in forms and form names.

- **Exclude Methods**
  When this option is selected, the search is done throughout the database, except in methods and method names.

Comments

4D 6.5 allows you to associate comments to objects in your database. Using comments is particularly appropriate for databases being developed by multiple programmers as well as, more generally, the maintenance of your database code.

A 4D comment is presented as text (the characters can have different font styles or colors, etc.) that can be modified and viewed at any time in Design mode. It can contain a description of an object to which it is associated as well as any information necessary to understand how the object functions in the database. The comments you create are stored in the database's structure.

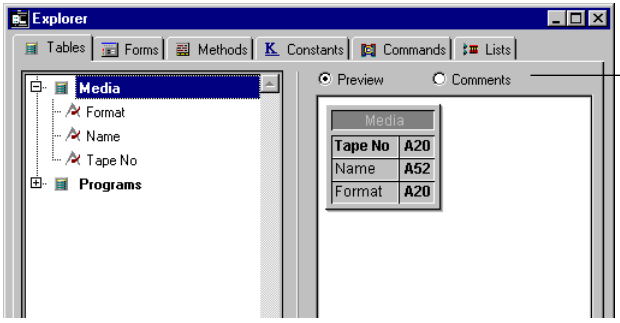
Moreover, 4D 6.5 allows you to generate automatic comments, which means that 4D automatically enters comments when an object is created or modified.

Note Comments generated by 4D 6.5 are compatible with 4D Insider's comments.

Methods (database methods, project methods, triggers, and form methods), forms, tables, subtables, and fields can be commented.

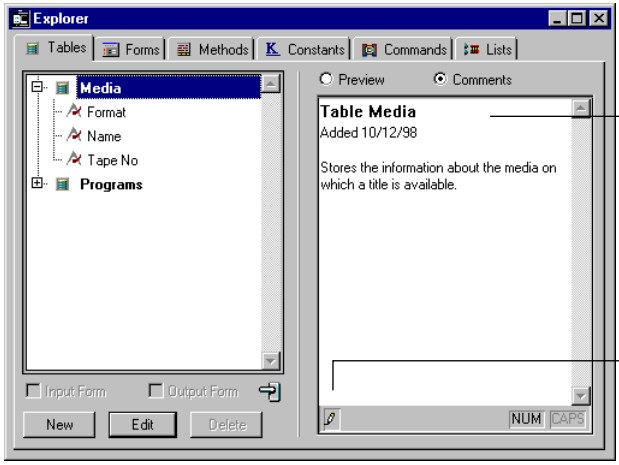
Associating a Comment to an Object

You can create, view, and modify comments from the Explorer. The Explorer's preview area now contains two radio buttons: **Preview** and **Comments**.



The **Preview** option (selected by default) corresponds to the Explorer's standard function, as in previous versions of 4D.

When the **Comments** option is selected, the preview area is replaced by the comments area.



The comments area has a vertical scrollbar that allows you to scroll through the text.

4D Server The locking icon, located in the bottom left corner of the area, indicates if the comment is already being edited by another user. If this is the case, the pencil has a slash through it and the comment can only be viewed.

► To view, create, or modify a comment:

1 **Select the object (method, table, or field) to document in the left portion of the Explorer.**

2 **Click on the "Comments" radio button, if it has not already been selected.**

The comments for the selected object appear.

3 **Enter or modify the text in the comments area.**

The text is saved as soon as you click outside of the entry area. You can enter up to 32 KB (32,000 characters) of text for each object.
You can use the standard text edition commands (**Copy**, **Paste**, **Select All**, etc.) available in the **Edit** menu or by using keyboard shortcuts in the comments area. You can also navigate the text in the comments area by using keyboard shortcuts as you would for any other text area.

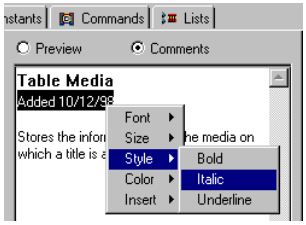
Modifying the Font Attributes for the Comments

You can enhance the font style of the comments (bold or italic), select another font, and even modify its color.

► To modify the comments's font style:

1 **In the Comments area, select the text that you want to modify.**

2 **On Windows: Click in the area with the right mouse button.**
or
On MacOS: Control+click in the area.
A hierarchical pop-up menu appears:



3 **Select the font attributes that you want to apply to the text.**

Inserting the Date, Time, or User

4D 6.5 offers you shortcuts to insert standard information, such as the date, time and user name (as defined in 4D's password table) into your comments.

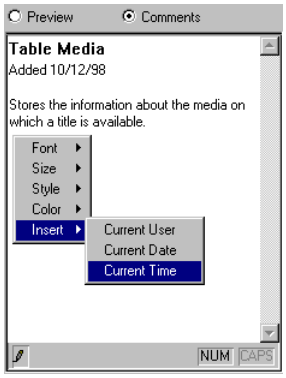
Note These shortcuts are particularly appropriate for the automatic generation of comments (see below), but can also be used to accelerate comments being entered manually.

► To insert the date, time or user name in a comment:

1 **In the Comments area, click on the location where you want to insert the information.**

2 **On Windows: click in the area with the right mouse button.**
Or
On MacOS: Control+click in the area.

A hierarchical pop-up menu appears:



3 Select the information you want to insert.

The selected information is inserted immediately into the Comments area with its current value.
If the database doesn't have a password system, the user name will not appear.

Note This shortcut helps you enter data, but does not allow you to insert variables. Information inserted in this way can only be updated manually. To insert automatic comments, please refer to the following section.

Inserting Automatic Comments

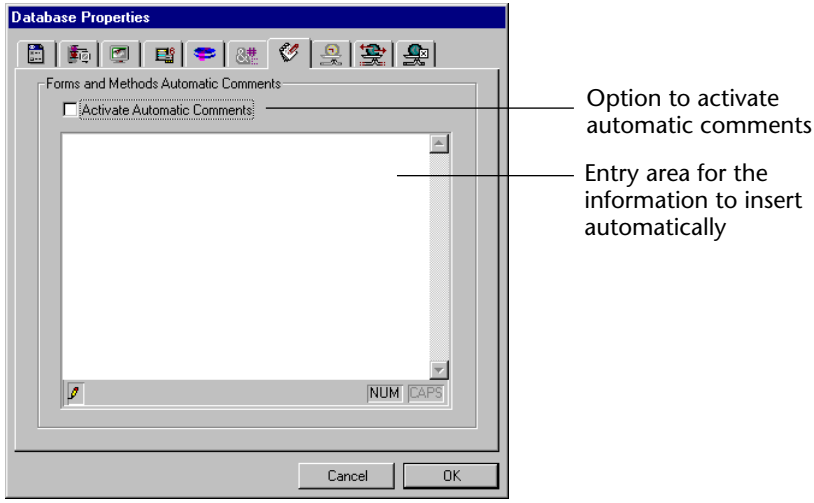
You can activate an automatic commenting system that can only be used for methods and forms in the database.

When this system is activated, a comment is automatically associated to every method or form created or modified in the database.

► To activate the automatic comments system:

1 In the Database Properties dialog box, click on the "Comments" tab.

The following page appears:



2 **Select the "Activate Automatic Comments" option.**

3 **Enter the information that you want to be inserted automatically in the entry area below.**

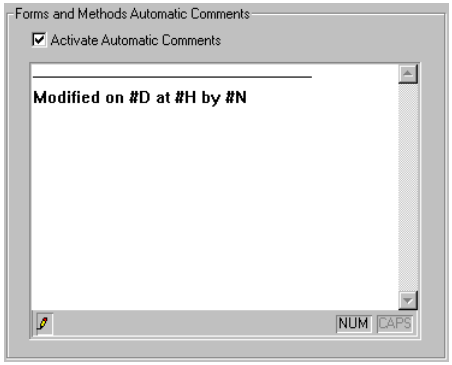
You can use variables that will be updated before being inserted in the comments:

- **#D** for the date
- **#H** for the time
- **#N** for the current user

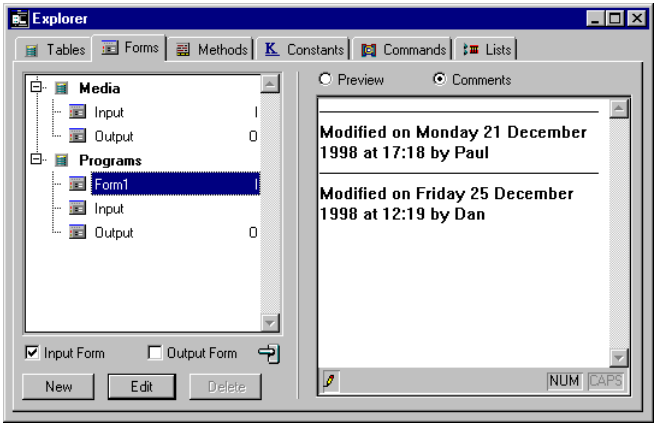
Note If your database doesn't have a password system, #N returns an empty string.

You can also insert this information by using a pop-up menu (refer to step 2 in the paragraph "Inserting the Date, Time, or User", page 59).

For example, if you enter the following values:



The Comments area for every method and form created or modified will be updated:



4D Server Automatic comments can be modified by any client workstation that has access to Database Properties. You can also modify these parameters from the server station.
Every modification made to the automatic Comments page is immediately taken into consideration by each client workstation as soon as an object is modified and its comments are accepted.

Editing Methods

4D 6.5 contains a new system to help you write methods: you can view a command's syntax in the Explorer and pre-enter it in the Methods Editor. The Methods editor contains new navigational shortcuts. You can define the font used as well as set break points directly in the Methods Editor.

Syntaxical Help

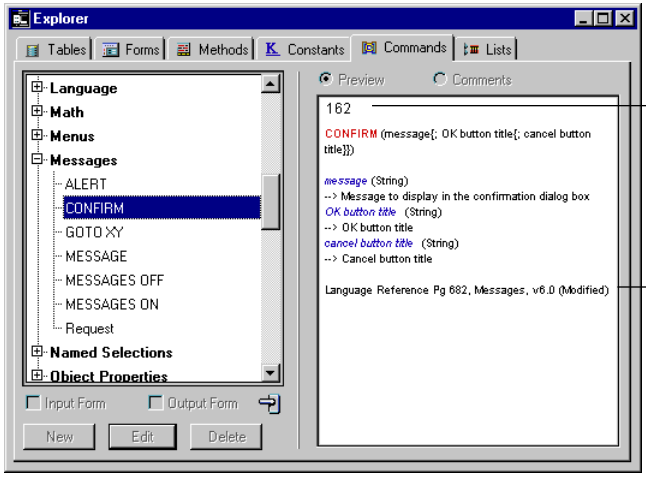
The new syntaxical help system consists of three elements:

- the possibility to view the syntax of 4D commands,
- the possibility to insert a 4D command with its syntax,
- viewing syntax errors directly in the Method Editor's window.

Viewing the Syntax

You can view the syntax of 4D commands in the Explorer or in the Methods Editor.

- **Viewing the syntax in the Explorer**
  To view the syntax of 4D commands in the Explorer, you have to click on the command name. The preview area (when it is visible) displays the command's syntax, a brief description of the parameter(s), as well as the page of the "Language" manual where the command is described:



The command's number

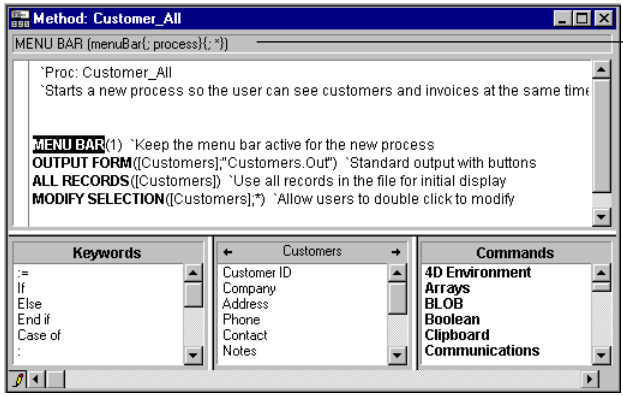
Command's syntax along with a description of the parameter(s)

The location of the command's description in 4D's pdf documentation

- **Viewing the syntax in the Methods Editor**
  You can view the syntax of a command directly in the Methods Editor. To do so, you have two possibilities:
  - select a routine in the Command list
  - enter or select a command in the edition area and hit the **Enter** key

Note Remember that you can also select a command in the Methods Editor and hit the **F1** key to display 4D's Windows on-line help for the selected command.

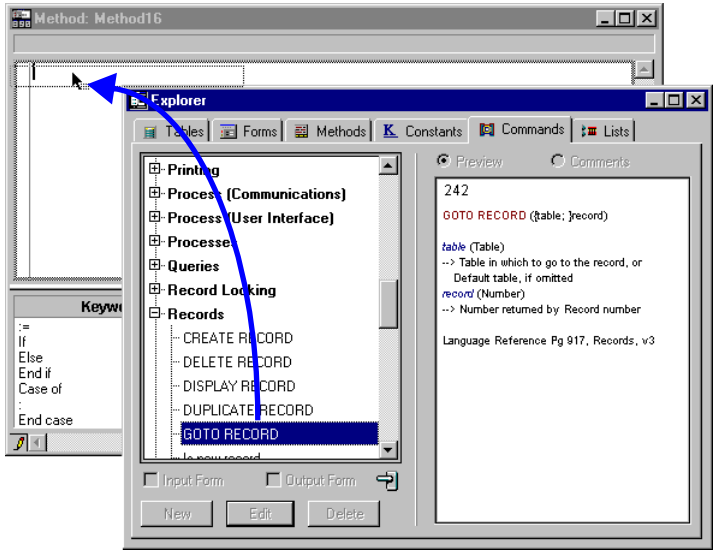
The syntax is displayed in a new area at the top of the Method Editor's window:



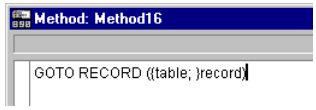
Pre-entering the Command's Syntax

You can now insert a command with its syntax (which consists of all of its parameters) in the Methods Editor. Of course, the inserted syntax appears more as an "input mask" that you need to adapt to your code. This feature allows you to not forget any parameters while writing a method.

To insert a command with its syntax, you need to drag it from the Explorer and drop it in the Methods Editor:



By default, the command is inserted with its syntax:



If you want to drag and drop a command without pre-entering the syntax (as in previous versions of 4D), hold down the **Alt** (Windows) or **Option** (MacOS) key.

Note Viewing and pre-entering a command's syntax are only possible if the 4D Help file is present in your disk. To benefit from these features, make sure that the help file has not been moved or renamed:
• on MacOS: the 4D Help file must be placed in your system's Preferences:ACI folder or in the 4D application's folder.
• on Windows: the 4D Help.RSR file must be placed in the Windows\ACI folder (where Windows represents the Windows system files folder) or in the 4D application's folder.

Viewing Syntax Errors

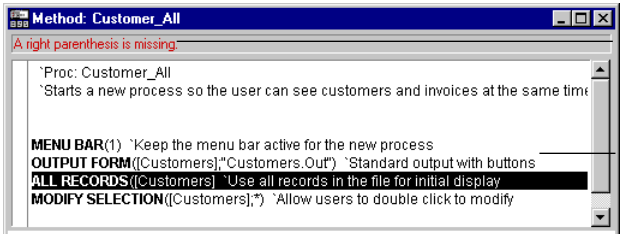
Syntax errors are now indicated directly in the Method Editor's window. 4D verifies the syntax when a line of code or the entire method is validated.

Validating the syntax of a line of code is done automatically when the cursor leaves the line (if you have clicked on another line or hit the **Return** key). You can also validate a line of code (without going to the following line) by hitting the **Enter** key.

Validating the entire method is done automatically when you save or close the window. You can also force the validation of the method by pressing **Ctrl+Enter** (Windows) or **Command+Enter** (MacOS).

At the moment of "validation", 4D checks each line's syntax (the syntax of the commands) and/or the structure of the instructions (**If... End if**, etc.).

When an error is detected, a message is written in the information area at the top of the method's window and 4D selects the line of code that contains the error:



Navigational Keyboard Shortcuts

Helpful keyboard shortcuts to navigate the code are now available in 4D's Method Editor. These shortcuts are also available in all of 4D's dialog boxes that contain data entry areas.

Windows	MacOS	Action
[Shift]+[→]		Create and enlarge the selection, character by character, to the right, or Reduce the selection, character by character, from the left
[Shift]+[←]		Reduce the selection, character by character, from the right or Create and enlarge the selection, character by character, to the left
[Shift]+[↓]		Create and enlarge a selection, line by line, from the top to the bottom
[Shift]+[↑]		Create and enlarge a selection, line by line, from the bottom to the top
[Ctrl]+[Shift]+[→]	[Command]+[Shift]+[→]	Create and enlarge the selection, word by word, from the right
[Ctrl]+[Shift]+[←]	[Command]+[Shift]+[←]	Reduce the selection, word for word, from the right, or create and enlarge the selection, word by word, from the left
[Ctrl]+[→]	[Command]+[→]	Move the insertion point, word by word, from left to right
[Ctrl]+[←]	[Command]+[←]	Move the insertion point, word by word, from right to left
[Home]		Place the insertion point at the beginning of the line
[End]		Place the insertion point at the end of the line
[Ctrl]+[Home]	[Command]+[Home]	Place the insertion point at the beginning of the method
[Ctrl]+End	[Command]+[End]	Place the insertion point at the end of the method
[Shift]+[Home]		Select all the characters in the line that are to the left of the cursor
[Shift]+[End]		Select all the characters in the line that are to the right of the cursor
[PgUp]		Scroll the contents of the method, page by page, from the bottom to the top (doesn't modify the insertion point)
[PgDn]		Scroll the contents of the method, page by page, from the top to the bottom (doesn't modify the insertion point)

Defining the Font in the Methods Editor

You can now define the font used in the Methods Editor (this feature was previously only available in Customizer Plus) in the Database Properties dialog box.

▶ To define the font used in the Methods Editor:

1 **In the Database Properties dialog box, click on the "User Interface" tab.**

2 **In the "Method Editor" area, define the font and font size (in pixels) that you want the editor to use.**

3 **Accept the dialog box.**

The modification is immediately applied to all the methods viewed and/or modified with your 4D application.

Inserting Break Points

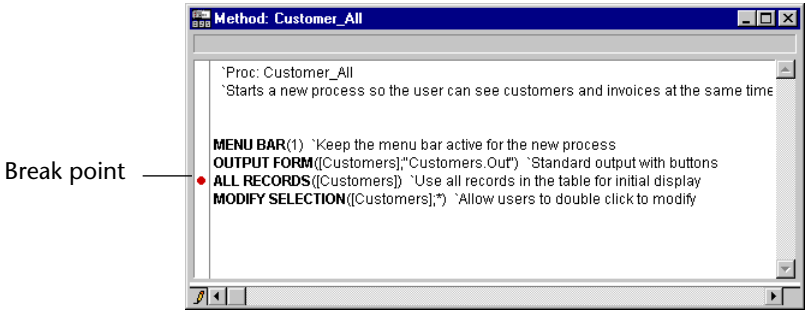
You can now insert, modify or delete break points directly in the Methods Editor. Of course, there is a dynamic interaction between the Methods Editor and the Runtime Explorer (as well as the debugger) in regards to break points. Actually, break points can be defined in these three editors.

Note For more information about the Runtime Explorer, refer to the paragraph "Runtime Explorer", page 49.

▶ To add a break point in the Methods Editor:

1 **Click in the left column where you want to insert a break point.**

When the mouse pointer is placed above this column, it is transformed into a dot. Once you click, a red bullet appears in the left column:



Note Break points always remain in the same place in the Methods Editor, even if you insert or delete lines of code.

► To delete a break point in the Methods Editor:

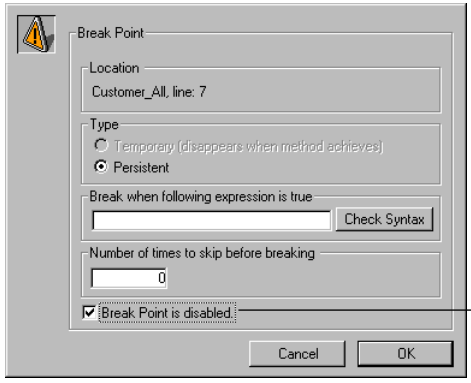
1 Click on the break point that you want to delete.

The break point disappears.

► To disable a break point in the Methods Editor:

1 Hit the Alt (Windows) or Option (MacOS) key and click on the break point.

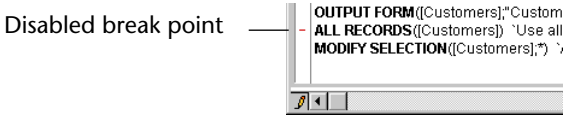
The Break Point Properties dialog box appears:



Note For more information about this dialog box — as well as about break points in general— refer to the "Debugger" chapter in 4D's *Language* manual.

2 Check the option to disable the break point and accept the dialog box.

The break point is then disabled:



New Database Properties

4D version 6.5 allows you to define new options in the Database Properties dialog box. Besides the properties related to passwords (described in the paragraph "Passwords", page 74) and to the Web server (described in the chapter "Web Server", page 155), you can also set the following properties:

- Managing the "@" character,
- Defining the location of the temporary folder,
- Automatically registering clients at startup (4D Server).

Note Generic parameters are also accessible when you use the new routine SET DATABASE PARAMETER, page 125.

Managing the @ Character

You can now define how 4D evaluates the @ character (the "at" sign) during searches or character string comparisons when it is found in a word. It can be considered either as a "wildcard" (for more information, please refer to 4D's *User mode* manual) or as a simple character. This possibility is particularly useful for databases that store e-Mail addresses (which are usually "name@provider.xx").

▶ To define how the @ character is to be considered:

1 **In the Database Properties dialog box, click on the second tab, "Data control and access".**

The corresponding page appears. By default, the **Consider @ as a character for Query and Order By** option is not checked. As in previous versions of 4D, the @ character is considered a "wildcard".

If you select this option, the @ character will be considered as a normal character when it is found in the middle of a string of (Alpha or Text) characters.

This option has an influence on searches, sorts, string comparisons as well as data stored in tables and data found in memory, like arrays. Fields and variables of type alpha (indexed or not) and text are concerned with how the @ character is managed by 4D.

Notes • For searches, it is important to note that if the search criteria **begins** or **ends** with @, the "@" character will be considered as a joker. Only if the "@" character is placed **in the middle of** a word (for example: bill@cgi.com) will 4D treat it differently.

• This option can also have an influence on the behavior of the commands in the "Object Properties" theme that accept the joker character ("@") in the object parameter. Please refer to 4D's *Language* manual.

2 Select or deselect the option and click on the OK button.

3 Close and reopen the database.

This step is necessary for the option to take effect.

Once the database is reopened, all of the database's indexes are automatically re-indexed.

Defining Where to Store the Temporary Folder

You can now designate the location on your disk where you want 4D to store the temporary files that are created while the database is executing. 4D mainly uses temporary files for transactions and named selections.

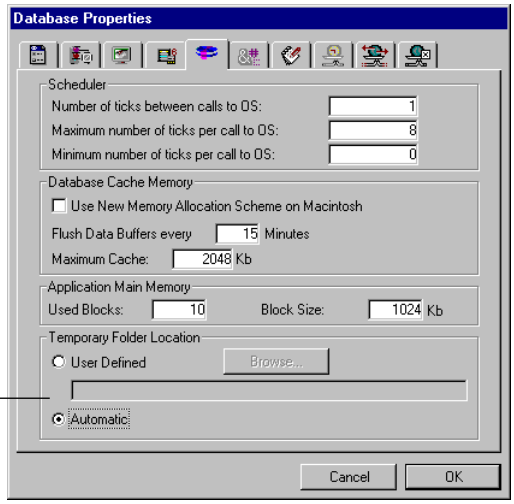
By default, temporary files are managed automatically, as in previous versions of 4D:

- on Windows, 4D places temporary files on volume C,
- on MacOS, 4D places temporary files on the volume that has the most available free space (without taking into consideration mounted distant volumes).

► To modify the location of the temporary folder:

1 In the Database Properties dialog box, click on the "System settings" tab.

The following page appears:



Temporary folder definition area

By default, the **Automatic** option is selected.

2 **Click on the "User Defined" radio button.**

The **Browse...** button becomes enabled.

3 **Click on the Browse... button.**

A standard open file dialog box appears.

4 **Search for and select the location where you want to place the temporary folder and accept the dialog box.**

Temporary files for named selections and transactions will then be written to the location indicated. The personalized access path is stored in 4D's preferences file.

Note If the access path is incorrect, the **Automatic** mode is reactivated (without the option being modified in the Database Properties dialog box). It is the database developer's responsibility to make sure that the access path (names of the volumes or folders) is not modified.

Registering Clients at Startup (4D Server only)

This new option allows you to automatically register 4D Clients as soon as they connect to a 4D Server database. Once "registered", a client can execute anything requested by another client or by the server itself (please also see the description of the routine REGISTER CLIENT, page 113).

You can define this option in the workstation's or server's properties. In both cases, this option is applied to each client workstation that connects to the database (this option is stored in the database's structure file).

Note This option is selected by default in databases created with 4D version 6.5. In databases created with a previous version of 4D, it is not selected by default.

► To automatically register clients:

1 **In the Database Properties dialog box, click on the "Connections" tab.**

2 **Select the "Register Clients at Startup" option and accept the dialog box.**

Now, all the clients that connect to the database will be automatically registered. However, it is necessary to quit and restart the clients already connected for this option to take effect.

This option is mainly to be used by the **On 4D Client** function in the Method execution dialog box in the User mode (cf. paragraph "Executing a Method", page 96). To create a sophisticated system to distribute tasks between clients, it is preferable that you use the commands that have been created for this purpose (see the paragraph "Processes", page 113).

Passwords

New password management functions are now available:

- You can now define a group so that it has access to the User mode.
- You can now create a Default User.
- You can now modify and create a user password with more security.

Note The way in which the 4D Web server manages passwords has also been modified. For more information, please refer to the paragraph "Connection Security", page 156.

Group's Access to the User Mode

4D 6.5 allows you to define a group's access to the User mode. This option allows you to control and protect the access to the User mode when a user is in Custom Menus mode.

A user that is not part of the group that has access to the User mode will not be able to access the User mode from the Custom Menus mode, neither by using a menu command nor by using the standard shortcut (**Option+f** on MacOS, **Alt+F4** or close box on Windows). If a user tries to access the User mode without having the appropriate privileges, 4D automatically quits.

- The Designer and Administrator always have access to the User mode even if they are not explicitly part of the group that has access to the User mode.
- A user that has access to the Design mode always has access to the User mode, even if the user is not explicitly part of the group that has access to the User mode.

► To give a group access to the User mode:

1 **In the Database Properties window, click on the second tab, "Data control and access".**

The corresponding dialog box appears.

2 **Choose a group in the "User Mode Access" drop-down list.**

You must have previously defined the access groups in 4D's Password editor.

3 **Accept the dialog box.**

Default User

You can now define a “Default User” to use your database. When this option is active, the Password dialog box is not presented to the users that open or connect to the database. Moreover, if you have not associated a password to the Default User, the Password dialog box no longer appears, the database opens directly.
Each user has the access privileges and restrictions defined for the Default User.
This option simplifies access to the database while maintaining a complete data control system.

► To define a Default User:

1 **In Design mode, create a user (the name you choose is not important) in the Password Editor.**

You can associate a password to the user, but it is not mandatory.

2 **In the different 4D editors, choose the access privileges and restrictions for this user.**

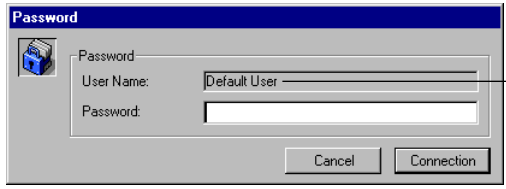
3 **In the Database Properties window, click on the “Data control and access” tab.**

4 **Choose your user in the “Default User” drop-down list (located at the bottom of the window).**

5 **Accept the dialog box.**

The access to the database is now no longer customized.

- If you have associated a password to the Default User, a dialog box appears when the database is opened and the Default User’s password must be entered:



- If you haven’t associated a password to the Default User, the above dialog box doesn’t appear.

Redisplaying the Password Dialog Box

You can "force" 4D to display the standard Password dialog box to, for example, connect to the database as the Designer or Administrator.

► To redisplay the Password dialog box when the "Default User" mode is active:

1 **Open (or connect yourself to) the database by holding down the Shift key.**

A dialog box appears allowing you to enter a name and password.

Creating and Modifying a Password

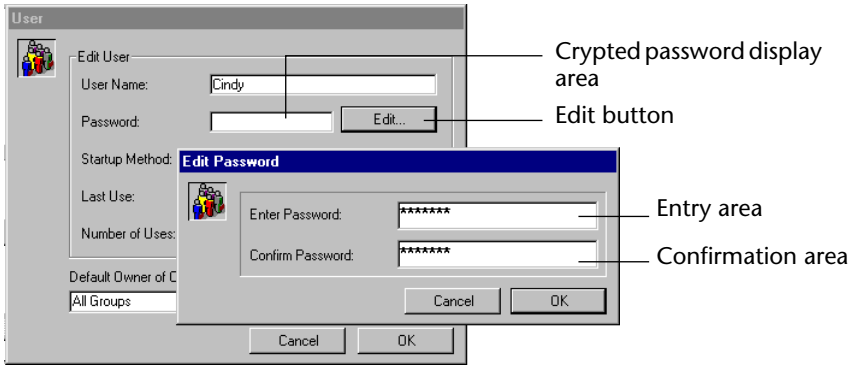
Creating and modifying a user password have now been better secured. You now have to enter the new password twice to reduce the risks of a typing error.

► To create or modify a password:

1 **In the User definition dialog box, click on the "Edit..." button.**

An entry dialog box appears.

2 **Enter the password in the entry area and then enter it again in the confirmation area.**



3 **Accept the dialog box.**

If the two entries are different, 4D beeps and the dialog box remains open.

Picture Library

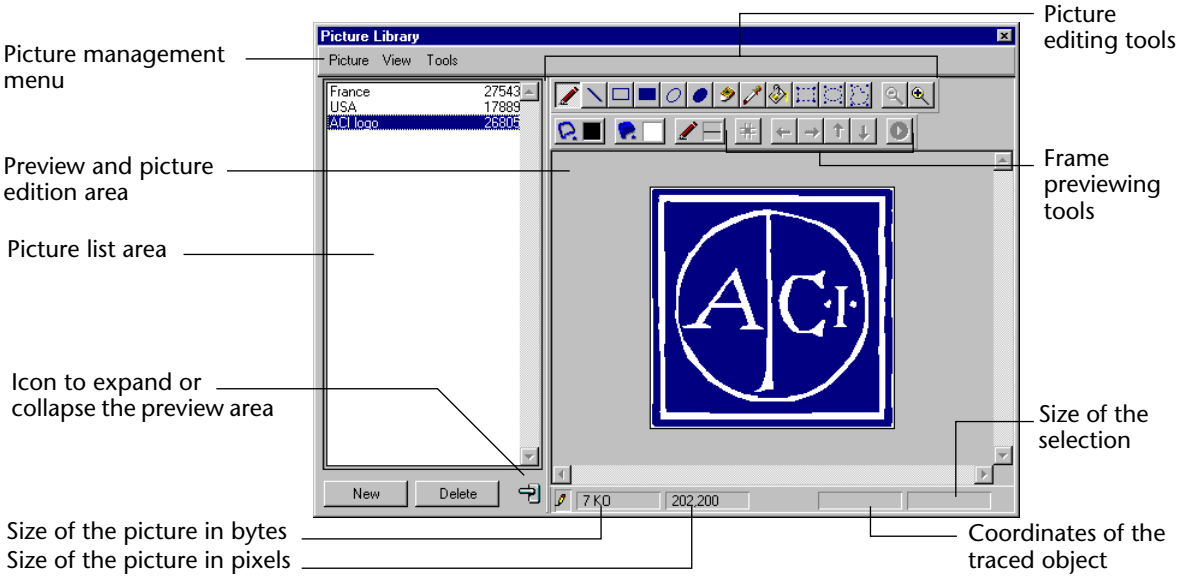
The possibilities offered by the picture library have been extended in 4D version 6.5. The modifications mainly affect the following areas:

- Managing and viewing pictures.
- Creating and modifying pictures directly in the library.
- Modifying and creating picture frames.

Managing and Viewing Pictures

The Picture Library now appears as a floating palette, which allows you to work simultaneously with different windows (for example, different forms) while keeping the library in the front.

The window contains menus that allow you to access picture management functions as well as a tool bar to edit the contents of the pictures. You can also hide or show the picture's viewing/editing area by clicking on the switch at the bottom of the picture list.



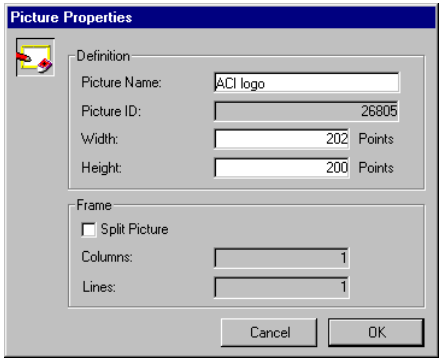
Adding Pictures

You can now add pictures to the Picture Library in three ways:

- **By importing a picture file**.
  To do so, select **Import picture...** from the library's **Picture** menu. In the standard open file dialog box that appears, select the picture to import.
- **By pasting a picture from the Clipboard**.
  This solution corresponds to the previous functioning of the picture library. To do so, copy a picture in the Clipboard, select the Picture list area and press **Ctrl+v** (on Windows) or **Command+v** (on MacOS), or choose **Paste** from the **Edit** menu.
- **By creating an empty picture in which you draw its contents.**
  The picture library contains an integrated graphical editor. To use it, click on the **Create** button in the palette or select **New Picture...** from the **Picture** menu. For more information about creating or modifying pictures, please refer to the paragraph "Creating and Modifying Pictures", page 79.

Picture's Properties

No matter how you add a picture, the following dialog box appears to define the picture' s properties:



- **Picture Name**: allows you to modify the default name of the picture.
- **Picture ID**: allows you to assign an ID number to the picture.

  WARNING: *You can define this number when you create the picture, but you cannot modify it afterwards.*
- **Width** and **Height**: allow you to define the size of the picture. These values are precalculated when you import a picture (from a file or from the Clipboard). When you split the picture (see below), the values correspond to the size of each frame.

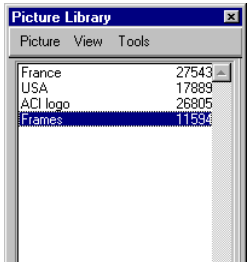
- **Frame area**: allows you to define the picture's splitting into frames so as to create buttons or picture menus. For more information, please refer to the paragraph "Frames Assistant", page 81.

To create the picture, accept the dialog box. Once the picture's properties have been defined, you can modify them at any time by selecting the picture and by selecting **Picture Properties...** from the library's **Picture** menu.

Viewing Pictures

You can now define how you want to view the images in the Picture Library, by using the **View** menu:

View as a **List** (default mode)



View as a **Picture**

Creating and Modifying Pictures

The Picture Library has a tool bar (like a Paint program) that allows you to draw and modify pictures.
To create or modify a picture, you just have to click on the view/edit area.

The Picture Library's editor works only in* bitmap *mode. Of course, you can import a picture — from a file or from the Clipboard — of a different type (for example, a vectorial picture) and its characteristics will be conserved when you use it in your database. On the other hand, if you modify this picture in the editor, it will be transformed into a bitmap. In this case, its particular characteristics will be lost when the modified picture is saved. When you modify an imported picture in the library, an alert dialog box appears to tell you that the picture will be converted into a bitmap.

The Picture Library's graphic tools are the following:

Icons	Tools	Function	Option keys
	Pen	Draws point by point	Alt (Option): allows you to take the color above which the cursor is located
	Line	Draws a line	Shift: the angles are in multiples of 45°
	Hollow Rectangle + Full Rectangle	Draws a hollow rectangle Draws a full rectangle	Shift: draws squares Ctrl (Command): the rectangle is drawn from its center
	Hollow Oval + Full Oval	Draws a hollow oval Draws a full oval	Shift: draws circles Ctrl (Command): the oval is drawn from its center
	Eraser	Erases by using the current background color	
	Color Picker	Modifies the line color by using a color from the picture	
	Flood Fill	Fills an enclosed area with the current background color	
	Selection tools	Create a selection	
	Zoom	Zoom the picture	
	Outline Color + Fill Color	Line color and background color menus	These options can be modified from the editor's contextual menu (Click with the right mouse button on Windows or Control+click on MacOS)
	Line width	Line width menu	

Note You also have other traditional editing commands (such as copy, paste, etc.) at your disposal from the **Edit** menu, 4D's tool bar or the standard keyboard shortcuts.

• If you use the standard Paste command while the picture edition area is selected, the Clipboard contents is inserted into the currently edited picture.

• If you use the standard Paste command while the picture list is selected, a new picture is added to the list.

Saving and Cancelling Modifications

Any modification made to a picture is automatically saved in the library as soon as you click outside of the edition area (which means as soon as the edition area "loses" focus).

WARNING: *Once the picture has been saved, you cannot revert any of the modifications that you have made to it.*

While modifying a picture, you can cancel the last modification made to it by selecting **Cancel** from the **Edit** menu (or the tool bar) in 4D, as well as the standard shortcut **Ctrl+z** (Windows) or **Command+z** (MacOS).
You can also cancel all the modifications made to a picture by selecting **Revert to saved** from the library's **Picture** menu.

Frames Assistant

The picture library contains integrated functions that allow you to create and modify "frames". Frames are used to create picture buttons or picture menus. The idea consists of splitting a picture in lines and columns, each cell is therefore a "frame". 4D takes care of displaying the correct frame in the picture button or picture menu according to the specified parameters (see the *Design Reference* manual).

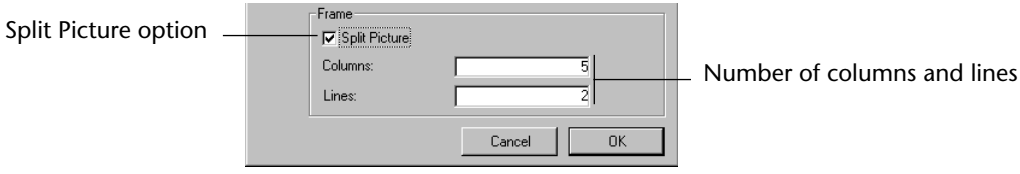
You can define a sequence of frames when creating a picture or even afterwards. You can also insert or delete lines, columns, or frames in an already defined sequence of frames.

Creating a Sequence of Frames

You can create a sequence of frames from a picture already placed in the Picture Library or when you add a picture to the Picture Library.

In both cases, you define how the picture is to be split in the Picture Properties dialog box. If you are creating a picture, the dialog box appears automatically. Otherwise, choose picture and select **Picture Properties...** from the **Picture** menu.

The Frame area allows you to define the number of lines and columns of your frame sequence. To create frames, you must first check the **Split Picture** option:

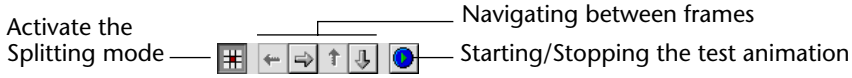


Size of the Frames

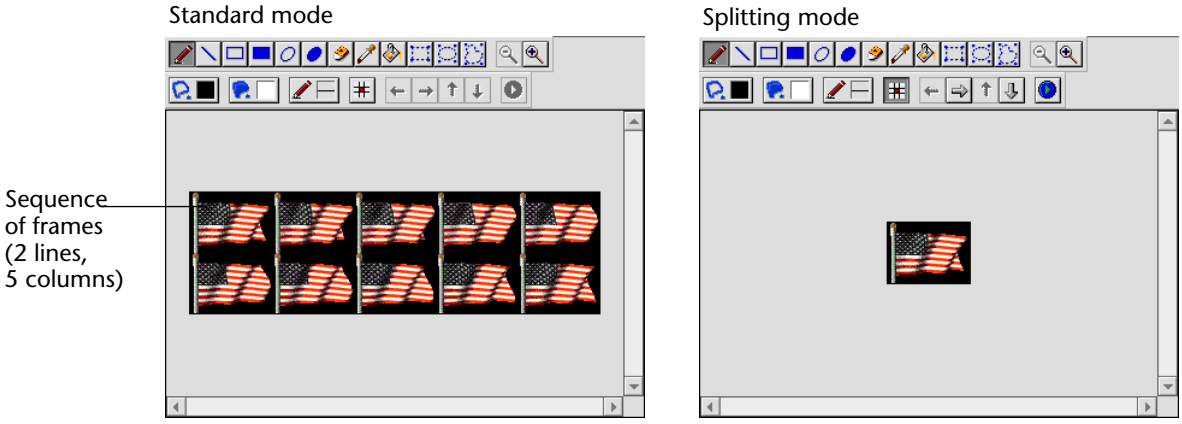
The size of the frames is automatically calculated by 4D. When you define a sequence of frames, the “Width” and “Height” areas are modified and the size of each frame is displayed.
If you want to modify the size of the frames later, you just have to enter new values into “Width” and “Height” without worrying about the global size of the picture. Each resulting frame will automatically be centered (without the picture being distorted) in the new size if it is bigger. If the new size is smaller, each frame will be truncated.

Viewing the Frames

You can preview each frame to check the sequence’s appearance by using specific tools in the Picture Library’s tool bar:



When you activate the splitting mode, 4D splits the picture and displays the first frame (located in the upper left corner). You can then navigate the frames by using the arrow buttons.



You can also display all the frames automatically. This function is particularly useful if you want to create picture buttons in a continuous sequence. To do so, click on the **Test animation** button. The frames appear in a continuous manner. To stop the test, click on the **Test animation** button one more time.

Inserting and Deleting Frames

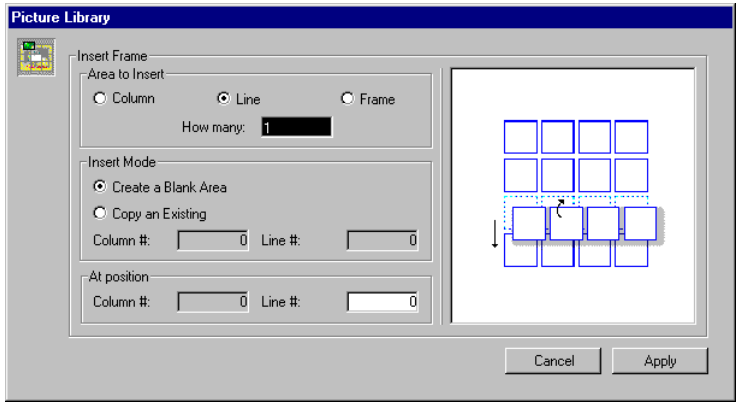
The Picture Library allows you to insert and delete frames in a previously created sequence of frames to suit your needs. You can insert frames as well as columns and lines.

► To insert frames:

1 **Select the frame sequence to modify.**

2 **Select Insert... from the Picture Library's Tools menu.**

The following dialog box appears:



3 **Define the column, line or frame that you want to insert and indicate how many in the upper portion of the dialog box.**

The preview area on the right side of the dialog box shows you an example of the selected operation. Note that inserting an element moves the others (no element is replaced).

4 **In the Insert Mode area, indicate if you want the inserted element to be blank or if it should contain the contents of an existing element.**

In this last case, you must designate the element to recopy.

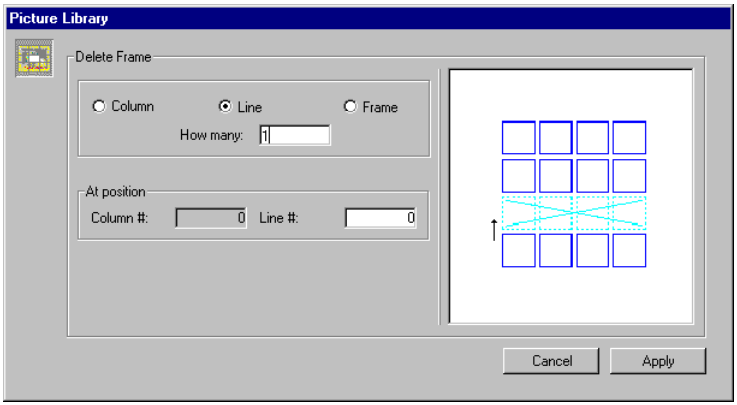
Note The number of the first column and first line is 0.

5 **Indicate the location in which you want the new element to be inserted and accept the dialog box.**

The element is then inserted in the picture.

► To delete frames:

1 **Select Delete... from the Picture Library's Tools menu.**
The following dialog box appears:



2 **Define the element (column, line, or frame) that you want to delete and indicate how many.**
The preview area on the right of the dialog box shows you an example of the selected operation.

3 **Indicate the position where you want the element to be deleted and accept the dialog box.**

Note The number of the first column and first line is 0.

Shortcut to Insert Picture Buttons/Pop-up Menus

When a picture is defined as a sequence of frames, you can use shortcuts to insert it as a picture button or a picture pop-up menu in your forms:

- To create a picture button, drag the picture from the library and drop it in the form.
- To create a picture pop-up menu, drag the picture from the library and drop it in the form while holding down the **Shift** key.

Note Dragging and dropping a picture that is not defined as a sequence of frames will be inserted as a standard picture.
The new Property List allows you to distinguish between two types of pictures: **Picture Library** (dynamically updated when the source picture in the library is modified, as in previous versions of 4D.) and ***Static Picture*** (not associated to the library's source picture). Refer to the paragraph "Property List", page 33.

Event Viewer

4D version 6.5 allows you to take advantage of Windows NT's "Event Viewer". This service receives and stores messages coming from the system and applications that are being executed so that you can control the sessions.

Configuration

To use this feature, the following conditions must be met:

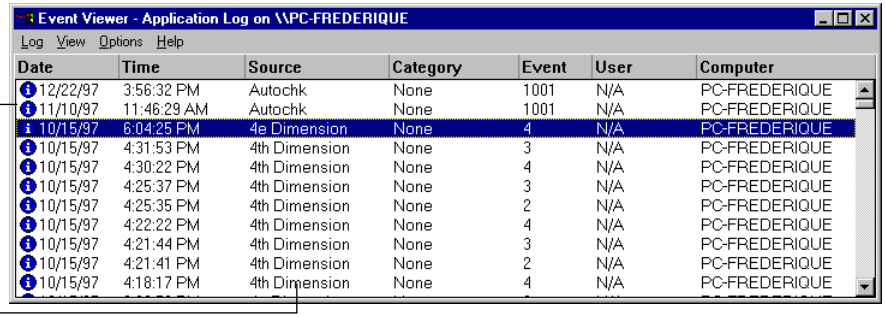
- The 4D application must be executed on Windows NT.
- The 4DMSG.DLL file must be in the System32 subfolder in the Windows System folder.
- The Windows NT Event Log must have already been launched.

Viewing the Event Viewer

In Windows NT's **Start** menu, choose **Programs**, **Administrative Tools**, and then **Event Viewer**.

In the **Log** menu, select **Applications**. The **Event Viewer** window shows the messages sent by applications.

Message type icon
Blue: information
Yellow: warning
Red: error



Application that provoked the event

To obtain a message's details, double-click on the row. 4th Dimension sends information messages about the starting up of databases, Web servers, etc.

You can generate your own messages in the Event Viewer from your 4D databases by using the LOG EVENT command. Refer to the paragraph "System Environment", page 130.

3 User Mode

Several functions and tools in the User mode have been modified in version 6.5 of 4D:

- Data import and export functions have been completely revamped.
- The Label editor has been modified.
- The method execution dialog box now contains new options.
- The default keyboard shortcut for adding records in subforms on MacOS has been modified.

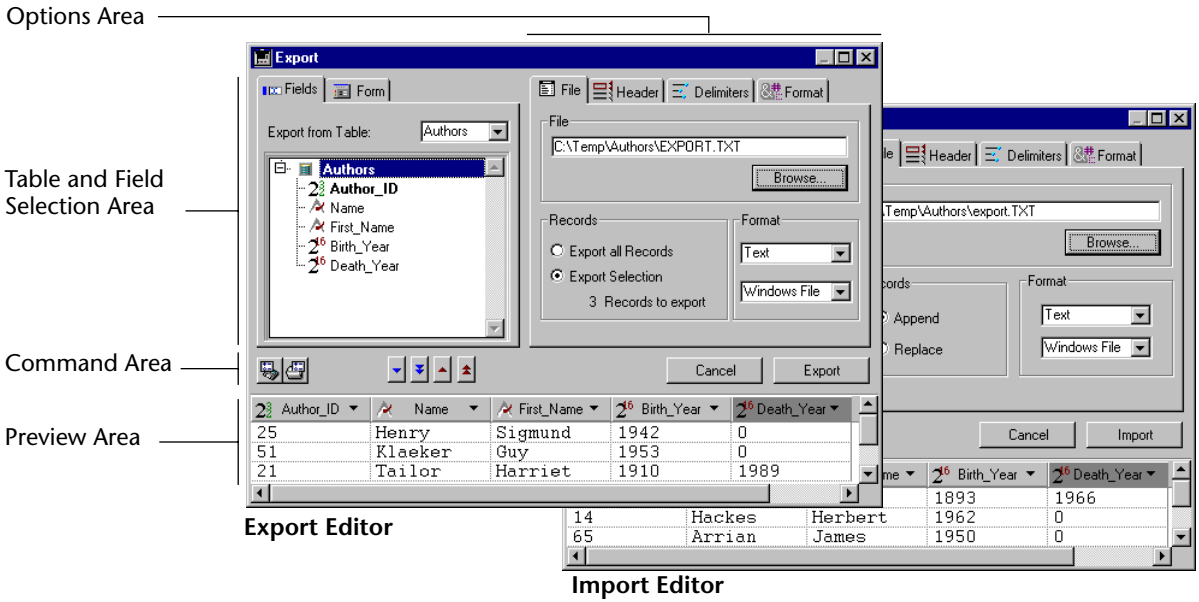
Import - Export

In 4D 6.5, the data import and export functions benefit from numerous new features, such as new editors, new options to set the import and export procedures, new file formats, and also the possibility to save and load import and export "preference" files that contain all your import and export settings.

Note New import-export commands have also been added to 4D 6.5. Please refer to the paragraph "Import-Export", page 102 in the "Language" chapter of this manual.

New editors

Importing and exporting records can now be executed by using their new editors. The two windows appear similar; however, their contents differ. The editors contain four main areas:



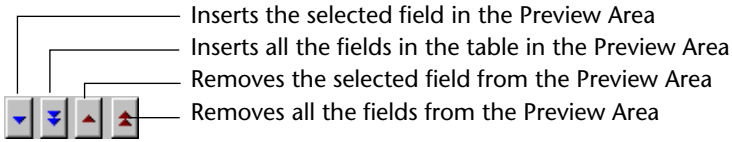
Export Editor

Import Editor

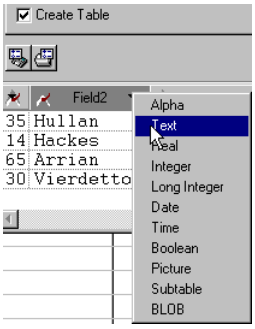
Table and Field Selection Area

This area allows you to define the fields you want to import or export. As in previous versions of 4D, you can manually select the fields to import or export or you can use a 4D form.
The menu of the area allows you to choose the entire database table. Therefore, you don't need to preselect the table to be used for the import or export in 4D.

- **Export**: to manually select a field to export, you just have to double-click on the field's name in the hierarchical list or select it by clicking on it and then clicking the insert button (which is located in the command area). The editor contains numerous buttons that allow you to insert and to remove fields in the preview area (only if you don't use a form for the export) :



- **Import**: The hierarchical list allows you to view the fields of the selected table (by default the first table is selected). The value of each entry is shown in the preview area (cf. paragraph "Preview Area", page 90).



The **Create Table** option allows you to import data into a table created directly by the Import editor. When you select this option, 4D creates a new database table (which you can rename later) to import the data into. 4D determines the number of fields necessary as well as the type of each field according to the data being imported.
You can modify the data type for the created fields by clicking on the field's title bar in the Preview Area. A menu displays the different types available.
The table is created only during the import. If you cancel the import or deselect the option, the database structure remains unmodified.

If you want to use a form for the import or export, click on the **Form** tab, located at the top of the area. You can choose a form from the list of forms for the selected table appears. The import or export will be done in the entry order of the fields defined in the form. You can use forms that contain enterable variables. Methods associated to variables, as well as to fields, will be executed when each record is imported or exported. Note, however, that these methods are not executed in the Preview Area.

Value to Field (Import Editor)

Value to Field button 

The Value to Field button allows you to "intelligently" assign values from the import file to the fields in the 4D table according to their type and not according to the order in which they were created (by default). The type of the import value is estimated, then it is assigned to a compatible 4D field. The interpretation is done in the following manner:

Estimated type of the value	4D Type
All numbers	Real (Number)
True/False	Boolean
Date in one of 4D formats	Date
Time	Time
Other	Alpha

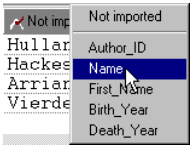
If no compatibility is found, the import value is declared "not imported". If you want to import it, you must assign a 4D field to it manually in the Preview Area.

Note This option is not available if you use a form for the import or if you select the **Create Table** checkbox.

Preview Area

This area allows you to preview the contents of the import or export as columns of data. You can resize each column by selecting and dragging the boundaries of the title bar with the mouse.
The title bar for each column indicates the type (by showing an icon) and the name of the imported or exported field. You can click in this bar and obtain the following information:

- **Export**: The list of the export table's fields as well as its related fields, if any, appear allowing you to modify the fields to export.
If you have clicked with the **right mouse button** (on Windows) or **Ctrl+click** (on MacOS) in the title bar, the list of display formats available for the selected field appears. You can define the format used for the export. If you don't modify this option, the **Default Format** is used (this option can also be defined in the **Format** options page, cf. paragraph "Format Page", page 92).
You can remove a column so as to exclude it from the export, and to not display it in the Preview Area. To do so, select the column and hit the **Delete** key.

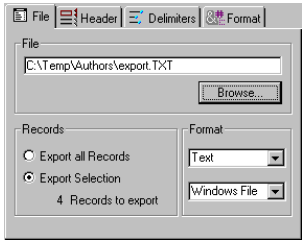


- **Import**: The list of the import table's fields appears, which allows you to modify the fields in which the data will be imported.
You can also select the **Not imported** option. In this case, the column of data is not imported.

New Options

The import and export editors now offer you new options that are accessible through tabs in the Options area. The number of tabs (between 1 and 5) varies according to the type of file used and the type of operation to be performed.

File Page



The "File" area indicates the file name, its access path and the type of import or export file selected. The **Browse** button allows you to choose or create another file.

The "Records" area allows you to indicate the manner in which you wish to use the current selection:

- **Export**: You indicate if you want to export all the records of the selected table (the **Export all Records** option) or only the current selection (the **Export Selection** option). In both cases, the number of records concerned is displayed in this area.

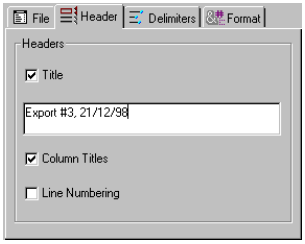
- **Import**: You indicate if the imported records replace the current selection (the **Replace** option) or if they are to be added to the existing data and create a new selection (the **Append** option).

The "Format" area allows you to indicate the format of the import or export file (for text files only). The option pages (tabs) will be displayed, or not, according to these settings. The new formats **Fixed Length Text** and **4th Dimension** are described in the paragraph "New File Formats", page 93.

You can also define the type of the file to be either **Macintosh** or **Windows**. This option has an influence on the definition of the delimiters[1]:

- **Macintosh file**: The delimiters will be the standard ones used on MacOS (End of field= **Tab**, End of record= **Return**, End of file= <**None**>).
- **Windows file**: The delimiters will be the standard ones used on Windows (End of field = **Tab**, End of record= **Return+Line feed**, End of file = <**None**>). Moreover, in this case 4D's MacOS/Windows conversion filter is used (remember that it is an internal filter to ensure multi-platform compatibility of 4D databases. 4D stores the data in ASCII Mac format).

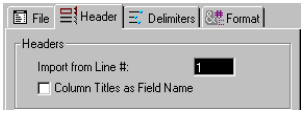
Header Page



For exports, this page allows you to specify the header of the file:

- **Title**: This option allows you to enter a title for the export document. Enter the title in the area below the check box.
- **Column Titles**: This option exports the titles of the columns, which are the field names.
- **Line numbering**: This option numbers each line, which is each exported record. The numbering starts at 1 and increments by 1.

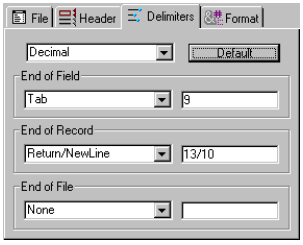
1. This option allows you to pre-enter values in the Delimiters Page. You can also modify them directly on this page.



For imports from a text file, two options are available:

- **Import from line #**: This option allows you to define the specific line from which 4D can begin importing the data. It is particularly useful when the import file starts with unformatted lines (title, date, etc.) because the import columns are calculated according to the format of the first line.
- **Column Titles as Field Name**: This option tells 4D to use the column titles as field names (if you have selected the **Create Table** option).

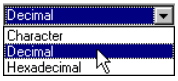
Delimiters Page



This page allows you to define the delimiters used in export or import files (for imports the file must be of type text).

Besides the standard delimiters, a new type of delimiter is available: **End of File**. This delimiter is necessary when exchanging files with certain applications.

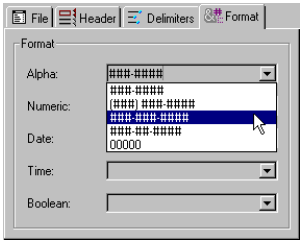
To modify the delimiters, you can use the scrolldown menus to the left of each area. These menus contain the values most frequently used for each delimiter type.
You can also directly enter a delimiter in the areas on the right.



The menu at the top of the page allows you to view the delimiters in different formats: **Character**, **Decimal** (default format) and **Hexadecimal**.

Note The delimiters are reinitialized to their default value if the type of the document is modified in the **File** page.

Format Page



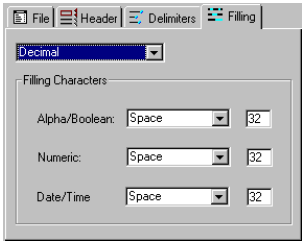
This page allows you to define the formats of the imported or exported values (only for text files). By default, 4D's standard formats are used.

- **Export**: You can define the formats of Alpha, numeric, Date, Time, and Boolean fields.
To do so, select the column to format in the Preview Area. According to the field type, the corresponding combo-box is enabled in the **Format** page. You can then select a 4D format or enter your own format.

Note You can also define a column's format by using the contextual menu (see the paragraph "Preview Area", page 90).

- **Import**: You can define the formats for Boolean fields. The syntax to apply is True_Value;False_Value. If you import a Boolean field whose values are “Black” (for true) and “White” (for false), you can enter Black;White. The combo-box displays the most commonly used Boolean formats.

Filling Page



This page only appears when the file format is “Fixed length text”. It allows you to define the filling characters to use according to the type of the values.
Substitution characters can be used for three data types: **Alpha/Boolean** (booleans are exported as alphas), **Numeric** (Integer, Longint, and Real) and **Date/Time**.

The menus propose the most frequently used values. The entry areas to the right display the characters selected. If you wish to use other characters, enter them directly in the entry areas.

The menu at the top of the page allows you to view the filling characters in different formats: **Character**, **Decimal** (default format) and **Hexadecimal**.

New File Formats

In 4D 6.5, you can import and export data in two new file formats: **4th Dimension** and **Fixed Length text**.

4th Dimension Format

This new file format, which is specific to 4th Dimension, allows you to simply exchange records between different 4D databases.
Moreover, this format allows you to exchange all of 4D’s data types. In particular, you can import and export picture and BLOB fields. The 4D export file extension is “.4IE”.

Fixed Length Text Format

The “Fixed Length text” format’s main characteristic is to use values with a fixed length, which means that in each data column, the values must have the same number of characters.
Of course, it is impossible that values, such as names, have the same number of letters. “Filling” characters are therefore used to fill in the “spaces”. You can define the filling characters for each data type or use default characters (see the paragraph “Filling Page”, page 93). However, if the values have a number of characters that is greater than the number defined, the values will be truncated.

- **Export**: When you choose the “Fixed Length text” format, the Preview Area indicates the number of characters for each column.
  By default, the number of characters is set in the following manner: Text fields have a length of 80 characters, the length of Alpha fields is equal to the number of characters defined in the Structure editor, and numeric fields have a length of 10 characters. You can modify these default numbers by entering a new value in the entry area or by manually resizing each column.
  The filling characters are always added at the end of alphanumeric fields and at the beginning of numeric fields.
- **Import**: When importing a file whose format is “Fixed Length text”, the preview area also displays the number of characters for each column.

Import-Export Settings

The import and export dialog boxes allow you to save and load your settings in separate files on disk.

To save or load the import or export settings, click the corresponding button in the Command area:

Import and export settings files have the extension “.4SI” on Windows (they are of type “4DSI” on MacOS). It stores all the settings defined in the Import or Export editor:

- File name and its access path
- Tables and fields selected as well as the name of the form, if one has been chosen
- Import and export options (file type, delimiters, etc.)

You can, therefore, retrieve all your settings and automate as much as possible the importing or exporting of your data. Using a settings file is particularly useful with the new import/export commands (Please refer to the paragraph “Import-Export”, page 102 in the “Language” chapter).

Note An import/export settings file does not store filters because they are related to the serial communication. If necessary, you will need to load an import or export filter before the operation.

Label Editor

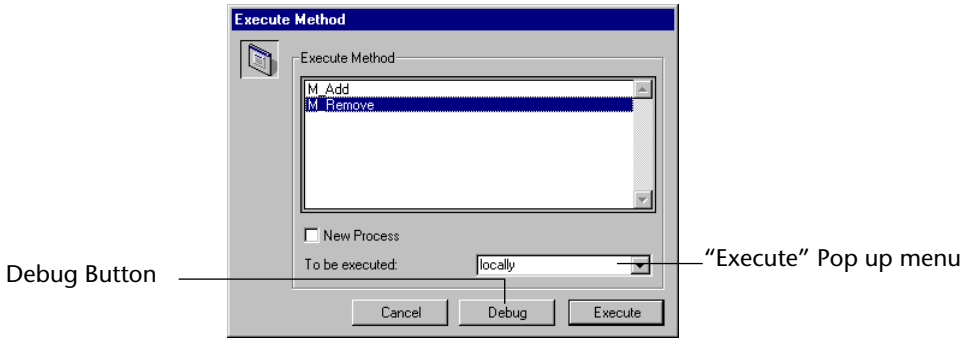
In the Label editor, the concatenation of fields now functions in the following manner:

- When you concatenate two fields by simply dragging and dropping them, the separator used is the one defined in 4D's resources. By default, the "space" character is used.
- You can concatenate two fields by using a "return" character as the separator. To do so, drag and drop the second field by holding down the **Shift** key.
  When the labels are being printed, if the second field is empty, its position is deleted and is replaced by the next field. If a field contains some text with Return characters, they are taken into account.
  For example, this feature allows you to create address labels by using many superimposed fields ([Clients]Address1, [Clients]Address2, etc.) without generating an empty line when an address only has one field filled in.

Note You can automatically resize a group of concatenated fields so that it corresponds to the number of lines that it contains. To do so, **Ctrl+click** (Windows) or **Command+click** (MacOS) on the lower right handle of the group.

Executing a Method

The method execution dialog box contains two new options:



- **Debug**: This button allows you to execute a method directly in Trace mode (cf. paragraph "Executing a method in trace mode", page 154).
- **"To be executed" Menu**: This option allows you to, in the case of a 4D Server database, execute the selected method directly **on 4D Server** (a stored procedure), or on another client workstation.

Note The **on 4D Server** option replaces the **Execute on server** option, which was available in previous versions of 4D.

So that other client workstations appear in the menu (and so that they can execute methods), they must have been previously "registered" (for more information, please refer to the description of the routine REGISTER CLIENT, page 113, as well as the paragraph "Registering Clients at Startup (4D Server only)", page 73).

Note By default, the **locally** option is selected. With the single version of 4D, only this option is available.

Adding Records to Subforms (MacOS)

Starting with version 8.5.1 of MacOS, users can switch between applications using the **Command+Tab** keyboard shortcut. This shortcut was previously used in 4D to add records to subforms.

In order to avoid any conflict, the standard keyboard shortcut for adding subrecords in 4D 6.5 is now **Command+/**.

4 Language

4th Dimension and 4D Server version 6.5 offer you numerous new commands. A few of the existing commands have also been modified.

The following three form events have been added to version 6.5: On After Keystroke, On Timer and On Resize. The existing form events On Clicked and On Double Clicked have also been modified.

In version 6.5, 4D's debugger has also been enhanced.

All of the modifications related to 4D's programming language are described in this chapter.

Note In version 6.5, the name of semaphores (as for variables) can contain up to 30 characters, including prefixes ($, <>).
A list item can now contain up to 255 characters.

New Commands

As in the 4D's Language manual, the commands in this manual are ordered by theme.

Arrays

BOOLEAN ARRAY FROM SET

BOOLEAN ARRAY FROM SET (booleanArr {; set})

Parameters	Type		Description
booleanArr	Boolean Array	←	Array to indicate if a record is in the set or not
set	String	→	Name of the set or UserSet if this parameter is omitted

This command fills an array of booleans indicating if each record in the table is or is not in set. There are as many elements in the booleanArr array than there are records in set. The elements in the array are ordered in the order in which the records are created in the table.

Each element of the array is:

- True if the corresponding record belongs to the set.
- False if the corresponding record does not belong to the set.

If N is the number of records in the table, element 0 of the array corresponds to record number 0, element 1 of the array corresponds to record number 1, etc.

If you don't pass the set parameter, the command will use UserSet in the current process.

LONGINT ARRAY FROM SELECTION

LONGINT ARRAY FROM SELECTION ({table;}recordArray {; selectionName})

Parameters	Type		Description
table	Table	→	Table of the current selection or default table if this parameter is omitted
recordArray	Longint array	←	Array of record numbers
selection-Name	String	→	Name of the named selection or the current selection if this parameter is omitted

This command fills the recordArray array with the (absolute) record numbers that are in selectionName.

If you do not pass the selectionName parameter, the command will use the current selection of table. It is only in this case that it is necessary to pass a table name as the first parameter.

Note The array element number 0 is initialized to -1.

Form Events

SET TIMER

SET TIMER (tickCount)

Parameters	Type		Description
tickCount	Longint	→	Tickcount

This command allows you to activate the On Timer form event and to set, for the current process, the number of ticks between each On Timer form event.

Note For more information about this new form event, please refer to paragraph “On Timer”, page 150.

If this command is called in a context in which it is not displaying a form, it will have no effect.

4D’s Web server can take advantage of this command as well as the On Timer form event to resend 4D forms. This feature allows you to obtain HTML pages updated in “real time” while economizing the bandwidth. Actually, updating a form in this case is not automatic; you must call the REDRAW command. You could then optimize the system by calling REDRAW only when the data has been modified.

Only browsers that interpret JavaScript allow you to automatically redraw pages. The period defined by SET TIMER will be used by the browser, which must be a few seconds (5 being a practical value) and the timeout of the Web process. Refer to the second example shown below.

To procedurally disable the triggering of the On Timer form event, call SET TIMER again and pass 0 to tickCount.

- Let's imagine that you want, when a form is displayed on screen, the computer to beep every three seconds. To do so, write the following form method:

```
If (Form event=On Load)
  SET TIMER(60*3)
End if

If (Form event=On Timer)
  BEEP
End if
```

- Let's imagine that you want your Web server to update a 4D form displayed on the Web browser every five seconds.To do so, write the following form method:

```
If (Form event=On Load)
  SET TIMER(60*5)
End if

If (Form event=On Timer)
  ... 'You can place a test here to see if the data is being modified and to
    'execute the following line only if this is true.
  REDRAW
End if
```

See Also: On Timer form event

Get edited text

Get edited text → Text

Parameters	Type		Description
This command does not require any parameters			
Result	Text	←	Text being entered

This command returns the text being entered in a form object.

This command is mainly to be used with the new form event On After Keystroke to retrieve the text as it is being entered. It can also be used with the On Before Keystroke form event.

Note To be in accordance with the new form event On After Keystroke, the existing event On Keystroke has been renamed, and is now called On Before Keystroke. For more information about the new features related to the form events, please refer to paragraph "Form Events", page 149.

In a context other than entering text in a form object, this function returns an empty string.

- The following method automatically puts the characters being entered in capitals:

```
If (Form event=On After Keystroke)
  [Trips]Agencies:=Uppercase(Get edited text)
End if
```

- Here is an example of how to process characters entered in a text field on the fly. The idea consists of placing in another text field (called "Words") all the words of the sentence being entered. To do so, write the following code in the field's object method:

```
If (Form event=On After Keystroke)
  $RealTimeEntry:=Get edited text
  PLATFORM PROPERTIES($platform)
  If ($platform#3) ` Macintosh or Power Macintosh
    Repeat
      $DecomposedSentence:=Replace string($RealTimeEntry;
                                          Char(32);Char(13))
    Until (Position(" ";$DecomposedSentence)=0)
  Else ` Windows
    Repeat
      $DecomposedSentence:=Replace string($RealTimeEntry;
          Char(32);Char(13)+Char(10))
    Until (Position(" ";$DecomposedSentence)=0)
  End if
  [Example]Words:=$DecomposedSentence
End if
```

Note This example is not exhaustive because we have assumed that words are separated uniquely by spaces (Char (32)). For a complete solution you will need to add other filters so as to extract all the words (delimited by commas, semi-colons, apostrophes, etc.)

Import-Export

Note 4D 6.5's import and export functions have been modified. These two new commands benefit from the new features proposed by version 6.5. For more information, please refer to paragraph "Import - Export", page 87.

IMPORT DATA

IMPORT DATA (fileName {; project{; *}})

Parameters	Type		Description
fileName	String	→	Access path and name of the import file
project	BLOB	→	Contents of the import project
		←	New contents of the import project (if the * parameter has been passed)
*		→	Displays the import dialog box and updates the project

This command allows you to import the data in the fileName file. 4D can import the data in the following formats: Text, Fixed length text, SYLK, DIF, DBF (dBase), and 4th Dimension.

If you pass an empty string to fileName, IMPORT DATA displays the standard save file dialog box, allowing the user to define the name, type, and location of the import file. Once the dialog box has been accepted, the *Document* system variable contains the access path and the name of the file. If the user clicks **Cancel**, the execution of the command is stopped and the *OK* system variable is equal to 0.

- If you don't pass the optional parameter project, the import dialog box is displayed. The user can define the import parameters or load an existing import project.

Note An import project contains all the import parameters such as the tables and fields in which to import, delimiters, etc. You define these parameters in the import dialog box. A project can be saved to disk and then loaded. For more information about the import dialog box, please refer to paragraph "Import - Export", page 87.

- If you pass a BLOB containing a valid import project to the project parameter, the import will be directly performed, without the user intervening. The project must already be predefined in the import dialog box, then saved. To do so, you have two possible solutions:
  - Save the project to disk, then load it by using the DOCUMENT TO BLOB command, in a field or a variable of type BLOB that you pass to project.
  - Use the IMPORT DATA command with an empty project parameter and the optional parameter *, then store the project parameter in a field of type BLOB (see below). This solution allows you to save the project with the datafile without having to load it from a BLOB on disk.

The optional parameter *, if it is specified, forces the display of the import dialog box with the parameters defined in project. This feature allows you to use a predefined project, while still having the possibility to modify one or more of the parameters. Furthermore, the project parameter contains, after closing the import dialog box, the parameters of the "new" project. You can then store the new project in a BLOB field, on disk, etc.

If the import was successful, the *OK* system variable is equal to 1.

See Also: EXPORT DATA

EXPORT DATA

EXPORT DATA (fileName {; project{; *}})

Parameters	Type		Description
fileName	String	→	Access path and name of the export file
project	BLOB	→	Contents of the export project
		←	New contents of the export project (if the * parameter has been passed)
*		→	Displays the export dialog box and updates the project

This command allows you to export the data to the fileName file. 4D can export the data in the following formats: Text, Fixed length text, SYLK, DIF, DBF (dBase), and 4th Dimension.

If you pass an empty string to fileName, EXPORT DATA displays the standard save file dialog box, allowing the user to define the name, type, and location of the export file. Once the dialog box has been accepted, the *Document* system variable contains the access path and the name of the file. If the user clicks **Cancel**, the execution of the command is stopped and the *OK* system variable is equal to 0.

- If you don't pass the optional parameter project, the export dialog box is displayed. The user can define the export parameters or load an existing export project.

Note An export project contains all the export parameters such as the tables and fields to export, delimiters, etc. You define these parameters in the export dialog box. A project can be saved to disk and then loaded. For more information about the export dialog box, please refer to paragraph "Import - Export", page 87.

- If you pass a BLOB containing a valid export project to the project parameter, the export will be directly performed, without the user intervening. The project must already be predefined in the export dialog box, then saved. To do so, you have two possible solutions:
  - Save the project to disk, then load it by using the DOCUMENT TO BLOB command, in a field or a variable of type BLOB that you pass to the project parameter.
  - Use the EXPORT DATA command with an empty project parameter and the optional parameter *, then store the project parameter in a field of type BLOB (see below). This solution allows you to save the project with the datafile without having to load it from a BLOB on disk.

The optional parameter *, if it is specified, forces the display of the export dialog box with the parameters defined in project. This feature allows you to use a predefined project, while still having the possibility to modify one or more of the parameters. Furthermore, the project parameter contains, after closing the export dialog box, the parameters of the "new" project. You can then store the new project in a BLOB field, on disk, etc.

If the export was successful, the *OK* system variable is equal to 1.

See Also: IMPORT DATA

Math

Euro converter

Euro converter (value; fromCurrency; toCurrency) → Real

Parameters	Type		Description
value	Real	→	Value to convert
fromCurrency	String	→	Code of the currency in which the value is expressed
toCurrency	String	→	Code of the currency into which the value must be converted
Result	Real	←	Converted value

The command Euro converter allows you to convert any value from and to the different currencies belonging to the “Euroland” and the Euro currency itself.

You can convert:

- a national currency into Euros,
- Euros into a national currency,
- a national currency into another national currency.
  In this case, the conversion is calculated by the intermediary of the Euro, as specified in the European reglementation. For example, to convert Belgian francs to Deutschemarks, 4D will perform the following calculations: Belgian francs -> Euros -> Deutchemarks.

Pass the value to convert in the first parameter.
The second parameter indicates the Currency code in which value is expressed.
The third parameter indicates the Currency code into which value must be converted.

To specify a Currency code, 4th Dimension proposes the following predefined constants, placed in the “Euro Currencies” theme:

Constant	Type	Value
Austrian Schilling	String	ATS
Belgian Franc	String	BEF
Deutsche Mark	String	DEM
Euro	String	EUR
Finnish Markka	String	FIM

Constant	Type	Value
French Franc	String	FRF
Irish Pound	String	IEP
Italian Lira	String	ITL
Luxembourg Franc	String	LUF
Netherlands Guilder	String	NLG
Portuguese Escudo	String	PTE
Spanish Peseta	String	ESP

If necessary, 4th Dimension performs rounding automatically on conversion results and keeps 2 decimals —except for conversions to Italian Liras, Belgian Francs, Luxembourg Francs and Spanish Pesetas, for which 4D keeps 0 decimal (the result is an integer number).

The conversion rates between the Euro and the currencies of the 11 participating Member States are fixed:

Currency	Value for 1 Euro
Austrian Schilling	13.7603
Belgian Franc	40.3399
Deutsche Mark	1.95583
Finnish Markka	5.94573
French Franc	6.55957
Irish Pound	0.787564
Italian Lira	1936.27
Luxembourg Franc	40.3399
Netherlands Guilder	2.20371
Portuguese Escudo	200.482
Spanish Peseta	166.386

- Here are some examples of conversions that can be done with this command:

```
$value:=10000   `Value expressed in French Francs
            `Convert the value into Euros
$InEuros:=Euro converter($value;French Franc; Euro)
            `Convert the value into Italian Lira
$InLires:=Euro converter ($value;French Franc; Italian Lira)
```

Named Selections

CREATE SELECTION FROM ARRAY

CREATE SELECTION FROM ARRAY ({table;} recordArray {; selectionName})

Parameters	Type		Description
table	Table	→	Table from which to create the selection, or Default table, if omitted
recordArray	Longint array or Boolean array	→	Array of record numbers, or Array of booleans (True = the record is in the selection, False = the record is not in the selection)
selection-Name	String	→	Name of the named selection to create or Apply the command to the current selection if the parameter is omitted

This command creates the named selection selectionName from:

- either an array of absolute record numbers recordArray from table,
- or an array of booleans. In this case, the values of the array indicate the belonging (True) or not (False) of each record in table in selectionName.

If you don't pass selectionName or if you pass an empty string, the command will be applied to the current selection, which will then be updated.

When you use a Longint array with this command, all the numbers of the array represent the list of record numbers in selectionName. If a number is incorrect (record not created), error -10503 is generated.

When you use a Boolean array with this command, the N^{th} element of the array indicates if the record number N is (True) or is not (False) in selectionName. The number of elements in array must be equal to the number of records in table. If the array is smaller than the number of records, only the records defined by the array can make up the selection.

Note With an array of booleans, the command uses elements from numbers 0 to N-1.

WARNING: *A named selection is created and loaded into memory. Therefore, make sure that you have enough memory before executing this command.*

See Also: CREATE SET FROM ARRAY

Object Properties

GET OBJECT RECT

GET OBJECT RECT ({*;} object; left; top; right; bottom)

Parameters	Type		Description
*		→	If specified = object is the name of the object (string) If omitted = object is a variable
object	Object	→	Object name (if * is specified) or Field or variable (if * is omitted)
left	Longint	←	Left coordinate of the object
top	Longint	←	Top coordinate of the object
right	Longint	←	Right coordinate of the object
bottom	Longint	←	Bottom coordinate of the object

This command returns the coordinates left, top, right and bottom (in points) in variables or fields of the object(s) of the current form defined by the parameters * and object.

If you pass the optional parameter *, it indicates that the object parameter is an object name (a string). If you don't pass the optional parameter *, it indicates that object is a field or a variable. In this case, you don't pass a string but a field or variable reference (only a field or variable of type object).

If you pass an object name to object and use the wildcard character ("@") to select more than one object, the coordinates returned will be those of the rectangle formed by all the objects concerned.

Note Since 4D version 6.5, it is possible to set the interpretation mode of the wildcard character ("@"), when it is included in a string of characters. This option has an impact on the "Object Properties" commands. Refer to paragraph "Managing the @ Character", page 70 in this manual.

If the object doesn't exist or if the command is not called in a form, the coordinates (0;0;0;0) are returned.

▼ Let's assume that you want to obtain the coordinates of a rectangle formed by all the objects that begin with "button":

GET OBJECT RECT(*;"button@";left;top;right;bottom)

See Also: MOVE OBJECT

MOVE OBJECT

MOVE OBJECT ({*;} object; moveH; moveV{; resizeH{; resizeV}}{; *})

Parameters	Type		Description
*		→	If specified= object is an object name (string) If omitted = object is a variable
object	Object	→	Object name (if * is specified) or Field or variable (if * is omitted)
moveH	Longint	→	Value of the horizontal move of the object (>0 = to the right, <0 = to the left)
moveV	Longint	→	Value of the vertical move of the object (>0 = to the bottom, <0 = to the top)
resizeH	Longint	→	Value of the horizontal resize of the object
resizeV	Longint	→	Value of the vertical resize of the object
*		→	If specified = absolute coordinates If omitted = relative coordinates

This command allows you to move the object(s) in the current form, defined by the * and object parameters moveH pixels horizontally and moveV pixels vertically.
It is also possible (optionally) to resize the object(s) resizeH pixels horizontally and resizeV pixels vertically.

The direction to move and resize depend on the values passed to the moveH and moveV parameters:

- If the value is positive, objects are moved and resized to the right and to the bottom, respectively.
- If the value is negative, objects are moved and resized to the left and to the top, respectively.

If you pass the first optional parameter *, you indicate that the object parameter is a parameter name (a string of characters). If you don't pass the * parameter, object is a field or a variable. In this case, you don't pass a string but a field or variable reference (only a field or variable of type object).

If you pass an object name to object and use the wildcard character ("@") to select more than one object, all the objects concerned will be moved or resized.

Note Since 4D version 6.5, it is possible to set the interpretation mode of the wildcard character ("@"), when it is included in a string of characters. This option has an impact on the "Object Properties" commands. Refer to paragraph "Managing the @ Character", page 70 in this manual.

By default, the values moveH, moveV, resizeH and resizeV modify the coordinates of the object relative to its previous position. If you want the parameters to define the absolute parameters, pass the last optional parameter *.

- The following line of code moves "button_1" 10 pixels to the right, 20 pixels to the top and resizes it to 30 pixels in width and 40 in height:

  MOVE OBJECT (*;"button_1";10;-20;30;40)

- The following line of code moves "button_1" to the following coordinates (10;20) (30;40):

  MOVE OBJECT (*;"button_1";10;20;30;40;*)

See Also: GET OBJECT RECT

Pictures

PICT TO GIF

PICT TO GIF (pict; blobGIF)

Parameters	Type		Description
pict	Picture	→	Field or variable of type Picture
blobGIF	BLOB	←	BLOB containing the GIF type picture

This command allows you to create a picture in GIF format from a picture (of type PICT) stored in a variable or in a 4D field.

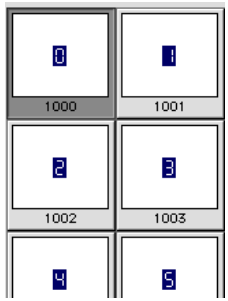
You pass a variable or a 4D field of type Picture to pict and a variable or a field of type BLOB to blobGIF. After executing the command, blobGIF contains the image in GIF format.

Note The GIF picture format can't contain more than 256 colors. If the original PICT picture contains more colors, certain colors may be lost. The GIF generated by this command is optimized according to the colors. It is of type 87a (opaque) and normal (not interlaced).

You can then save the picture in blobGIF in a file by using the BLOB TO DOCUMENT command or you can even publish it on the Web.

If the conversion was successful, the *OK* system variable is equal to 1. Otherwise, it will be equal to 0.

- Let's assume that you want to generate a GIF picture on the fly by displaying a connection counter. In the database's picture library, place all the numbers as pictures:



In the On Web Connection database method, you then write the following code:

```
If (Web Context)
   ...
Else
   C_BLOB ($blob)
   Case of
      ...
      : ($1="/4dcgi/counter") `Generating a GIF counter
            `When 4D detects this URL while sending the static page
         $blob:=gifcounter (◊nbHits)        `Calculates the GIF picture
            `The ◊nbHits variable contains the number of connections
         SEND HTML BLOB ($blob;"image/gif")
            `Insert the picture and send it to the browser
      ...
   End case
End if
```

Here is the *gifcounter* method:

```
C_LONGINT($1)
C_PICTURE($img)
C_BLOB($0)
If ($1=0)
  $ndigits:=1
Else
  $ndigits:=1+Length(String($1))
End if
If ($ndigits<5)
  $ndigits:=5
End if

$div:=10^($ndigits-1)
For ($i;1;$ndigits)
  $ref:=Int($1/$div)%10
  GET PICTURE FROM LIBRARY($ref+1000;picture)
  $img:=$img+picture
  $div:=$div/10
End for

PICT TO GIF($img;$0)
```

When sending a page to the Web browser, 4D displays a GIF picture of the following type:

Process (Communications)

Test semaphore

Test semaphore (semaphore) → Boolean

Parameters	Type		Description
semaphore	String	→	Name of the semaphore to test
Result	Boolean	←	True = the semaphore exists, False = the semaphore doesn't exist

This function allows you to test the existence of a semaphore.

The difference between the Semaphore function and the Test semaphore function is that Test semaphore doesn't create the semaphore if it doesn't exit. If the semaphore exists, the function returns True. Otherwise, it returns False.

▼ The following example allows you to know the state of a process (in our case, while modifying the code) without modifying semaphore:

```
$Win:=Open window (x1;x2;y1;y2;-Palette window)
Repeat
  If (Test semaphore("Encrypting code"))
    POSITION MESSAGE ($x3;$y3)
    MESSAGE("Encrypting code being modified.")
  Else
    POSITION MESSAGE($x3;$y3)
    MESSAGE("Modification of the encrypting code authorized.")
  End if
Until (StopInfo)
CLOSE WINDOW
```

See Also: Semaphore

Processes

REGISTER CLIENT

REGISTER CLIENT (clientName {; period} {; *})

Parameters	Type		Description
clientName	String	→	Name of the 4D Client session
period	Longint	→	Server's interrogation period (in seconds)
*	*	→	Local process

This command "registers" a 4D Client station with the name specified in clientName on 4D Server, so as to allow other clients or eventually 4D Server (by using stored methods) to execute methods on it by using the EXECUTE ON CLIENT command. Once it is registered, a 4D Client can then execute one or more methods for other clients.

Note You can also automatically register each client station that connects to 4D Server by using the "Register clients at startup" option in the Database properties dialog box (cf. paragraph "Registering Clients at Startup (4D Server only)", page 73).

When this command is executed, a process, named clientName, is created on the client station. This process can only be aborted by the UNREGISTER CLIENT command.
If you pass the optional * parameter, the created process is local. 4D will automatically add the dollar sign ($) at the beginning of the process name. Else, the process is global.

After executing the command, the client station will periodically ask 4D Server to see if another 4D Client or the server itself is calling it. By default, this interrogation is done every two seconds. You can modify this time period by modifying period. The minimum value is one second.

Note If this command is used with a single-user version of 4th Dimension, it has no effect.

Once the command is executed, it is not possible to modify 4D Client's name or the server's interrogation period on the fly. To do so, you must call the UNREGISTER CLIENT command, then the REGISTER CLIENT command.

Note More than one 4D Client can have the same registered name.

If 4D Client is correctly registered, the *OK* system variable is equal to 1. If 4D Client was already registered, the command doesn't do anything and *OK* is equal to 0.

▼ In the following example, we are going to create a small messaging system that allows the client workstations to communicate between themselves.

1) This method, *Registration*, allows you to register a 4D Client and to keep it ready to receive a message from another 4D Client:

```
  `You must unregister before registering under another name
UNREGISTER CLIENT
Repeat
  vPseudoName:=Request("Enter your name:";"User";"OK";"Cancel")
Until ((OK=0) | (vPseudoName # ""))
If (OK=0)
  ...` Don't do anything
Else
  REGISTER CLIENT(vPseudoName)
End if
```

2) The following instruction allows you to get a list of the registered 4D Clients. It can be placed in the On Startup database method:

```
PrClientList:=New process("4D Client List";32000;"List of registered clients")
```

3) The method *4D Client List* allows you to recuperate all the registered 4D Clients and those that can receive messages:

```
If (Application type=4D Client)
    ` the code below is only valid in client-server mode
  $Ref:=Open window(100;100;300;400;-(Palette window+
        Has window title);"List of registered clients")
  Repeat
    GET REGISTERED CLIENTS($ClientList;$ListeCharge)
      `Retrieve the registered clients in $ClientList
    ERASE WINDOW($Ref)
    GOTO XY(0;0)
    For ($p;1;Size of array($ClientList))
      MESSAGE($ClientList{$p}+Char(Carriage return))
    End for
      `Display each second
    DELAY PROCES(Current process;60)
  Until (False) ` Infinite loop
End if
```

4) The following method allows you to send a message to another registered 4D Client. It calls the *Display_Message* method (see below).

```
$Addressee:=Request("Addressee of the message:";"")
  ` Enter the name of the people visible in the window generated by the
  ` On Startup database method
If (OK # 0)
  $Message:=Request("Message:") ` message
  If (OK # 0)
    EXECUTE ON CLIENT($Addressee;"Display_Message";
                                        $Message) ` Send message
  End if
End if
```

5) Here is the *Display_Message* method:

```
C_TEXT($1)
ALERT($1)
```

6) Finally, this method allows a client station to no longer be visible by the other 4D Clients and to no longer receive messages:

UNREGISTER CLIENT

See Also: EXECUTE ON CLIENT, UNREGISTER CLIENT, GET REGISTERED CLIENTS.

UNREGISTER CLIENT

UNREGISTER CLIENT

This command does not require any parameters.

This command "unregisters" a 4D Client station. The client must have already been registered by the REGISTER CLIENT command.

Note A 4D Client is automatically unregistered when the user quits the application.

If the client workstation was not previously registered or if the command was executed on a single-user 4th Dimension, the command has no effect.

If the client is correctly unregistered, the *OK* system variable is equal to 1. If the client wasn't registered, *OK* is equal to 0.

▼ Refer to the example for the REGISTER CLIENT command.

See Also: REGISTER CLIENT, GET REGISTERED CLIENTS

GET REGISTERED CLIENTS

GET REGISTERED CLIENTS (clientList; methods)

Parameters	Type		Description
clientList	Text array	←	List of the saved 4D Clients
methods	Longint array	←	List of the methods to be executed

This command fills two arrays:

- clientLists contains the list of clients who were "registered" by using the REGISTER CLIENT command.
- methods supplies the list of each client's "workload". The workload is the number of methods that a 4D Client must still execute by calling the EXECUTE ON CLIENT command (for more information, please refer to the description of the EXECUTE ON CLIENT command).

If the operation was successful, the *OK* system variable is equal to 1.

▼ Let's assume that you want to obtain a list of all the registered clients and the methods that remain to be executed:

```
ARRAY TEXT($clients;0)
ARRAY LONGINT($methods;0)
GET REGISTERED CLIENTS($clients;$methods)
```

▼ Refer to the example of the REGISTER CLIENT command.

See Also: REGISTER CLIENT, EXECUTE ON CLIENT.

EXECUTE ON CLIENT

EXECUTE ON CLIENT (clientName; methodName {; param1; ...; paramN})

Parameters	Type		Description
clientName	String	→	4D Client's registered name
methodName	String	→	Name of the method to execute
param1... N		→	Method's parameter(s)

This command forces the execution of the methodName method, with the parameters param1... paramN, if necessary, on the registered 4D Client whose name is clientName. 4D Client's registered name is defined by the REGISTER CLIENT command.
This command can be called from a 4D Client or a stored method from 4D Server.

If the method requires one or more parameters, pass them after the name of the method.

The execution of the method on 4D Client is done in a global process automatically created on the client workstation, and its name will be the 4D Client's registered name.

If this command is called many times in a row on the same 4D Client, the execution orders will be stacked. Therefore, the methods will be treated one after another in asynchronous mode. The more methods that are stacked, the bigger the workload is for the 4D Client. You can know the state of the workload of each client by using the GET REGISTERED CLIENTS command.

Note The stacking of the execution orders cannot be modified or stopped unless 4D Client is unregistered by using the UNREGISTER CLIENT command.

You can simultaneously execute the same method on many or all of the registered 4D Clients. To do so, use the wildcard character (@) in the clientName parameter.

The *OK* system variable is equal to 1 if 4D Server has correctly received the execution request of a method; however, this does not guarantee that the method has been properly executed by 4D Client.

- Let's assume that you want to execute the "GenerateNums" method on the "Client1" client station:

  EXECUTE ON CLIENT("Client1";"GenerateNums";12;$a;"Text")

- If you want all the clients to execute the "EmptyTemp" method:

  EXECUTE ON CLIENT("@";"EmptyTemp")

- Refer to the example of the REGISTER CLIENT command.

  See Also: REGISTER CLIENT, UNREGISTER CLIENT, GET REGISTERED CLIENTS.

Process aborted

Process aborted → Boolean

Parameters	Type		Description
This command does not require any parameters			
Result	Boolean	←	True = the process is about to be aborted, False = the process is not about to be aborted

This command returns True if the process in which it is called is about to be interrupted unexpectedly, which means that the execution of the command was unable to reach its "normal" completion. For example, this can occur after calling QUIT 4D.

- This command can be used as a particular type of programming on the Web server, only in compiled mode. When you use a method to send Web pages by using a loop like While...End while (cf. example), the mechanism of the Web server doesn't allow you to stop the loop in case of a timeout (end of the inactivity period authorized) on a Web browser. If the Web process is not closed, a Web license is therefore still in use.
  The Process aborted command, placed in the initial test of the loop, will return True in case of a timeout. The loop can then be interrupted and the Web license can be freed.
  Here is a method that can be used to send HTML pages. In compiled mode, this loop cannot be interrupted in case of a timeout:

```
While (True)
   SEND HTML FILE (HTMLFile)
End while
```

The Process aborted command allows you to use the same type of method, while still being able to exit the loop and free the Web license in case of a timeout:

```
While (Not (Process aborted))
   SEND HTML FILE (HTMLFile)
End while
```

Queries

QUERY WITH ARRAY

QUERY WITH ARRAY (indexedField; array)

Parameters	Type		Description
indexedField	Field	→	Indexed field used to compare the values
array	Array	→	Array of the searched values

The QUERY WITH ARRAY command searches all the records for which the value of indexedField is equal, at least, to one of the values of the elements in array. The records found will become the new current selection.

This command allows you to quickly and simply build a search on multiple values.

Notes
- This command only works with indexed fields. It cannot be used with fields of type Text, Picture, Subfield, or BLOB.
- Remember that an array of type Longint is compatible with fields of type Time.

▼ The following example allows you to retrieve the records of both French and American clients:

```
ARRAY STRING (2;SearchArray;30)
SearchArray[1]:="FR"
SearchArray[2]:="US"
QUERY WITH ARRAY ([Clients]Country;SearchArray)
```

Find index key

Find index key (indexedField; value) → Longint

Parameters	Type		Description
indexedField	Field	→	Indexed field on which to execute the search
value		→	Value to search
		←	Value found
Result	Longint	←	Number of the record found or -1 if no record was found

This command returns the number of the first record whose indexedArray field is equal to value.
If no records are found, Find index key returns -1.

After calling this command, value contains the value found. This feature allows you to execute searches using the wildcard character ("@") on Alpha fields and then retrieve the value found.

This command doesn't modify the current selection or the current record.

This function is fast because it only uses the index, and is particularly useful to prevent creating double entries during data entry.

▼ In an audio CD database, during data entry let's assume that you want to verify the singer's name to see if it already exists in the database. Because homonyms can exist, you don't want the [Singer]Name field to be unique. Therefore, in the input form, you can write the following code in the [Singer]Name field's object method:

```
If (Form event=On Data Change)
  $RecNum:=Find index key([Singer]Name;[Singer]Name)
  If ($RecNum # -1)  ` If this name has already been entered
    CONFIRM("A singer with the same already exists. Do you want
                                        to see the record?";"Yes";"No")
    If (OK=1)
      GOTO RECORD([Singer];$RecNum)
    End if
  End if
End if
```

Records

Is new record

Is new record ({table}) → Boolean

Parameters	Type		Description
table	Table	→	Table of the record to examine or the default table if this parameter is omitted
Result	Boolean	←	True if the record is being created, Otherwise, False

This function returns True when the table's current record is being created and has not yet been saved in the current process.

Compatibility Note You can obtain the same information with the existing command Record number, by testing if it returns -3.
However, we strongly advise you to use Is new record instead of Record number in this case. Actually, the Is new record command insures a better compatibility with future versions of 4th Dimension.

▼ The following two instructions are identical. The second one is strongly advised so that the code will be compatible with future versions of 4D:

```
If (Record number([Table])=-3) `Not advised
   ` ...
End if

If (Is new record([Table])) `Strongly advised
   ` ...
End if
```

See Also: Record number

Is record loaded

Is record loaded ({table}) → Boolean

Parameters	Type		Description
table	Table	→	Table of the record to examine or Default table if this parameter is omitted
Result	Boolean	←	True if the record is loaded. Otherwise False

This function returns True if the table's current record is loaded in the current process.

▼ Instead of using the "Next record" or "Previous record" automatic actions, you can write object methods for these buttons to improve how they work. The "Next" button will display the beginning of the selection if the user is at the end of the selection and the "Previous" button will show the end of the selection when the user is at the beginning of the selection.

```
  ` Object method of the "Previous" button (without an automatic action)

If (Form event=On Clicked)
  PREVIOUS RECORD([Group])
  If (Not(Is record loaded([Group])))
    GOTO SELECTED RECORD([Group];Records in selection([Group]))
    `Go to the last record in the selection
  End if
End if

  ` Object method of the "Next" button (without an automatic action)

If (Form event=On Clicked)
  NEXT RECORD([Group])
  If (Not(Is record loaded([Group])))
    GOTO SELECTED RECORD([Groups];1)
      `Go to the first record in the selection
  End if
End if
```

Selection

HIGHLIGHT RECORDS

HIGHLIGHT RECORDS ({setName})

Parameters	Type		Description
setName	String	→	Set of records to highlight or UserSet if the parameter is omitted

This command allows you to highlight records in an output form. This operation is identical to manually selecting records in list mode by using the mouse or the **Shift+Click** or **Ctrl+Click** (**Command+Click** on MacOS) keyboard equivalents.
The "selected" records will be highlighted. The current selection is not modified.

Note The *UserSet* set is updated after redrawing the records; that is, after executing the entire calling method — and not just immediately after executing HIGHLIGHT RECORDS.

- If you pass a valid set name to setName, the command will be applied to the records in that set.
- If you omit the setName parameter, the command will only highlight the records in the current *Userset* set.
- In an output form displayed by the MODIFY SELECTION command, you want the user to be able to perform searches without the current selection being modified. To do so, place a **Search** button in the form and associate to it the following method:

```
SET QUERY DESTINATION(Into Set;"UserSet")
QUERY
SET QUERY DESTINATION(Into Current Selection)
HIGHLIGHT RECORDS
```

When the user clicks the button, the standard query dialog box appears. Once the search has been validated, the records found will be highlighted without the current selection being modified.

Sets

CREATE SET FROM ARRAY

CREATE SET FROM ARRAY ({table ;} recordsArray; {setName})

Parameters	Type		Description
table	Table	→	The set's table or the default table if the parameter is omitted
recordsArray	Longint array or Boolean array	→	Array of the record numbers, or array of booleans (True = the record is in the set, False = the record is not in the set)
setName	String	→	Name of the set to create, or Apply the command to the UserSet if this parameter is omitted

This command creates setName from:

- either an array of absolute record numbers recordsArray from table,
- or from an array of booleans recordsArray. In this case, the values of the array indicate if each record in the table belongs (True) or not (False) to setName.

When you use this command and pass a Longint array to recordsArray, all the numbers in the array represent the list of record numbers that are in setName. If a number is invalid (for example, if a record has not been created), error -10503 is generated.

When you use this command and pass a Boolean array to recordsArray, the Nth element of the array indicates if the record number N is integrated (True) or not (False) of to setName. Usually, the number of elements in the array must be equal to the number of records in the table. If the array is smaller than the number of records, only the records defined by the array will be in the set.

Note With a Boolean array, this command uses the elements from number 0 to N-1.

If you don't pass the setName parameter or pass an empty string, the command will be applied to the UserSet system set.

See Also: CREATE SELECTION FROM ARRAY

Structure Access

SET DATABASE PARAMETER

SET DATABASE PARAMETER ({table;} selector; value)

Parameters	Type		Description
table	Table	→	Table for which to set the parameters or the default table if this parameter is omitted
selector	Longint	→	Code of the database's parameter to modify
value	Longint	→	Value of the parameter

This command allows you to modify various parameters internal to the 4D database for the current process.

selector designates the code of the database's parameter to modify. 4th Dimension offers the following predefined constants, which are in the "Database Parameters" category:

Constant	Type	Value
Seq Order Ratio	Longint	1
Seq Access Optimization	Longint	2
Seq Distinct Values Ratio	Longint	3
Index Compacting	Longint	4
Seq Query Select Ratio	Longint	5
Minimum Web Process	Longint	6
Maximum Web Process	Longint	7
Web Conversion Mode	Longint	8

value designates the value of the parameter. The value passed depends on the parameter that you wish to modify.

Here are the possible values for each selector:

Selector	Values	Description
1	0 → 100,000	Ratio of the selected records (in relation to the total number of records) below which the sorts are executed in sequential mode. This ratio must be expressed in one hundred thousands (100,000nds). The default value is 9,000 (= 9 %).
2	0 or 1 0: not optimized 1: optimized	Optimization mode for sequential accesses (sorts, searches, selection to array). In optimized mode, 4D tries to read numerous records at a time from disk but doesn't place them in the cache. This mode is especially interesting if the size of the cache is low. By default, the value is 1 (optimized mode). For more information, please refer to chapter 6, "Optimizations", page 195.
3	0 → 100,000	Ratio of the selected records (in relation to the total number of records) below which the DISTINCT VALUES command will be executed in sequential mode. This ratio must be expressed in one hundred thousands (100,000nds).
4	0 or 1 0: no, 1: yes	Activation or disactivation to compress the index pages. By default, the value is 1 (indexes are compacted if necessary). For more information, please refer to chapter 6, "Optimizations", page 195.
5	0 → 100,000	Ratio of the selected records (in relation to the total number of records) below which the QUERY SELECTION command will be executed in sequential mode. This ratio must be expressed in one hundred thousands (100,000nds).
6	0 → 32,767	Minimum number(1) of Web processes to maintain in non contextual mode. By default, the value is 0.
7	0 → 32,767	Maximum number(1) of Web processes to maintain in non contextual mode. By default, the value is 10.
8	1 or 2 1 : 6.0.x mode 2 : 6.5 mode	4D forms Web conversion mode(2). By default, the value is 2 (6.5 mode).

(1) For the Web server to be reactive, in non contextual mode, 4D delays the Web processes for 5 seconds and reuses them to execute any possible future HTTP queries. In terms of performance, the principle is actually more advantageous than creating a new process for each query. Once a Web process is reused, it is delayed again for 5 seconds, except if the maximum number of Web processes has been reached (in which case it is aborted). If no query has been attributed to a Web process within the 5 seconds, the process is aborted, except if the minimum number of Web processes has been reached (in which case it is delayed again).

These parameters allow you to adjust how your Web server functions according to the number of requests, the available memory, etc.

(2) In some cases, the Web conversion of 4D forms created with 4D 6.0.x may be incorrect with 4D 6.5, especially if a form contains a reference to a HTML page, such as {mypage.htm}. In this case, to ensure the compatibility of the form, you can activate the 4D 6.0.x conversion mode.
This mode is set only for the process (Web context) into which the SET DATABASE PARAMETER has been called. It can be called from within the On Web Connection Database Method to ensure the 6.0.x compatibility of all the forms of a database, or just before a single form is displayed. If the command is called outside the contextual mode or a Web process, it has no effect.

Note An additional Selector can be used with the command Get database parameter: Database Cache Size (9). This Selector cannot be used with the command SET DATABASE PARAMETER. For more information, see below the description of the command Get database parameter.

See Also: Get database parameter

Get database parameter

Get database parameter ({table;} selector) → Longint

Parameters	Type		Description
table	Table	→	Table number or default table if this parameter is omitted
selector	Longint	→	Code of the database's parameter
Result	Longint	←	Value of the parameter

This command allows you to read the current value of the 4D database's parameters for the current process.

The selector parameter designates the parameter to read.
4th Dimension offers you the following predefined constants, which are in the "Database Parameters" category:

Constant	Type	Value
Seq Order Ratio	Longint	1
Seq Access Optimization	Longint	2
Seq Distinct Values Ratio	Longint	3
Index Compacting	Longint	4

Constant	Type	Value
Seq Query Select Ratio	Longint	5
Minimum Web Process	Longint	6
Maximum Web Process	Longint	7
Web Conversion Mode	Longint	8
Database Cache Size	Longint	9

To know the values that could be returned by this function for the selectors 1 to 8, please refer to the description of the SET DATABASE PARAMETER command.

The Database Cache Size (9) selector allows you to get the current database cache memory size. The returned value is expressed in bytes. Indeed, you can set the **Maximum Cache** as well as, on Macintosh only, the **Minimum Cache** values in the Database Properties dialog box. But the actual size of the memory allocated to the database cache depends on these parameters and the current status of the computer memory resources. The command Get database parameter allows you to get the actual size of the memory allocated to the database cache by 4D.

Note You cannot set the database cache memory size using the language. In other words, the Database Cache Size selector cannot be used with the command SET DATABASE PARAMETER.

See Also: SET DATABASE PARAMETER

System Documents

Select folder

Select folder ({message}) → Alpha

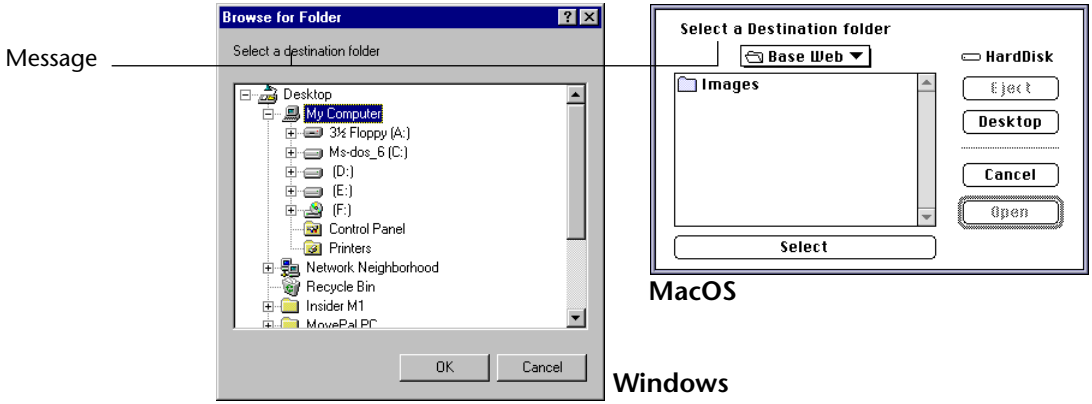
Parameters	Type		Description
message	Alpha	→	Title of the window
Result	Alpha	←	Access path to the selected folder

This command displays a dialog box that allows you to manually select a folder and to then retrieve the complete access path to the selected folder.

Note This command does not modify the 4D's current folder.

The Select folder command displays a standard dialog box to navigate in the workstation's volumes and folders.

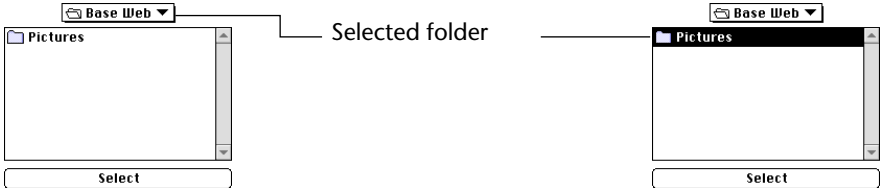
The optional parameter message allows you to display a message in the dialog box.



The user selects a folder and then clicks the **OK** button (Windows) or the **Select** button (MacOS). The access path to the folder is then returned by the function.

- On Windows, the access path is returned in the following format: “C:\Folder1\Folder2\SelectedFolder\”
- On MacOS, the access path is returned in the following format: “Hard Disk:Folder1:Folder2:SelectedFolder:”

Note On MacOS, whether or not the name of the folder is selected in the dialog box, the access path that is returned to you may be different.



4D Server This function allows you to view the volumes connected to the client workstations. It is not possible to call this function from within a stored procedure.

If the user validates the dialog box, the *OK* system variable is equal to 1. If the user clicks the Cancel button, the *OK* system variable is equal to 0 and the function returns an empty string.

Note On Windows, if the user selected certain incorrect elements, such as "Workstation", "Trash can", etc., the *OK* system variable is equal to 0, even if the user validates the dialog box.

▼ The following example allows you to select the folder in which the pictures in the picture library will be stored:

```
$PictFolder:=Select folder("Select a folder for your pictures.")
PICTURE LIBRARY LIST (pictRefs;pictNames)
For ($n;1;Size of array(pictNames))
  $vRef:=Create document($PictFolder+pictNames{$n};"PICT")
  If (OK=1)
    GET PICTURE FROM LIBRARY(pictRefs{$n};$vStoredPict)
    SAVE PICTURE TO FILE($vRef;$vStoredPict)
    CLOSE DOCUMENT($vRef)
  End if
End for
```

System Environment

LOG EVENT

LOG EVENT (message {; importance})

Parameters	Type		Description
message	String	→	Contents of the message
importance	Integer	→	Message's importance level

Note This feature is only available on Windows NT.

4D version 6.5 allows you to benefit from the "Log events" in Windows NT. This log file receives and stores messages coming from applications being executed. It therefore allows you to control the course of a worksession. For more information, please refer to paragraph "Event Viewer", page 85.

Note For this feature to be available, Windows NT **Log events** service must be started.

The LOG EVENT command allows you to add custom messages that will appear in Windows NT **Log events**.

Pass the message to write in the log events in message.

You can attribute a level of importance to message, which helps you to read and understand the log events. There are three levels of importance: Information, Warning, and Error. The importance parameter allows you to set the message's level of importance. 4th Dimension offers you the following predefined constants, placed in the "Windows NT Log Events" category:

Constant	Type	Value
Information Message (*default value*)	Integer	0
Warning Message	Integer	1
Error Message	Integer	2

If you don't pass anything to importance or pass an incorrect value, the default value (0) is used.

- If you want to have a record of when your database is opened, you could write the following line of code in the On Startup database method:

  LOG EVENT ("The Invoice database was opened.")

  Each time the database is opened, this information will be written in Windows NT's log events and its level of importance will be 0.

Web Server

SEND HTML BLOB

SEND HTML BLOB (blob; type {; noContext})

Parameters	Type		Description
blob	BLOB	→	BLOB to send to the browser
type	String	→	Data types of the BLOB
noContext	Boolean	→	True = Switch to non contextual mode False = Switch to contextual mode

This command allows you to send blob to the browser.

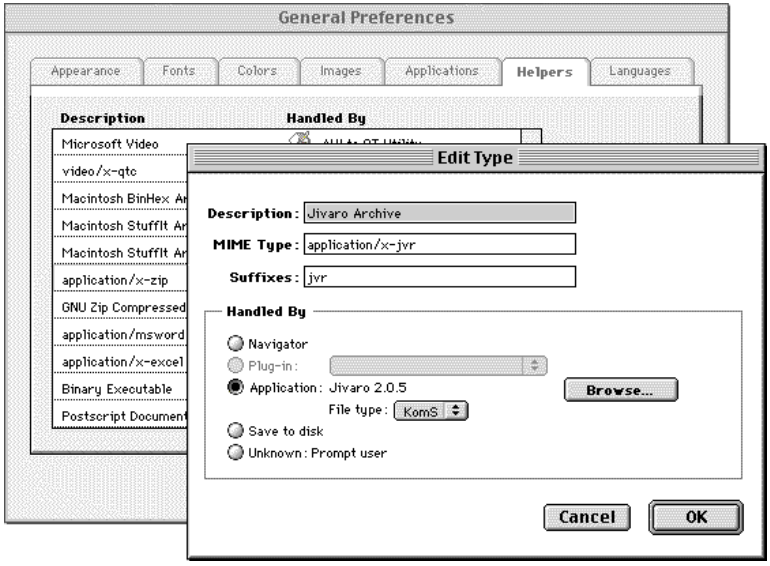
The type of data contained in the BLOB is indicated by type. This parameter can be one of the following types:

- type = **Empty String** (""): In this case, you don't need to supply any more information in the BLOB. The browser will try to interpret the contents of the BLOB.

- type = **File extension** (example: ".HTM", ".GIF", ".JPEG", etc.): In this case, you specify the MIME type of the data contained in the BLOB by indicating its extension. The BLOB will then be interpreted according to its extension. However, the extension must be a standard one so that the browser can correctly interpret it.
- type = **Mime/Type** (example: "text/html", "image/tiff", etc.): In this case, you directly specify the MIME type of data contained in the BLOB. This solution offers you more freedom. Besides the standard types, you can pass a custom MIME type to send proprietary documents via Intranet. To do so, you only need to configure the browsers so that they recognize the type sent and so that they can open the appropriate application. The value you pass to type is, in this case, "application/x-*[TypeName]*". In the client workstations's browser, you reference this type and associate it to the "Launch the application" action. The SEND HTML BLOB command allows you to therefore send all types of documents, the Intranet clients automatically open the associated application.

Note If the BLOB is of type "text/html" (.htm, .html, .shtm, .shtml), it is translated and analyzed as an HTML file.

Definition of MIME type **application/x-jvr** in Netscape 3.0 for Jivaro archives (ACI's compression/decompression file application)



Here is a list of the most common MIME types:

Extension	Mime/Type
.htm	text/html
.html	text/html
.shtml	text/html
.shtm	text/html
.css	text/css
.pdf	application/pdf
.rtf	application/rtf
.ps	application/postscript
.eps	application/postscript
.hqx	application/mac-binhex40
.js	application/javascript
.txt	text/plain
.text	text/plain
.gif	image/gif
.jpg	image/jpeg
.jpeg	image/jpeg
.jpe	image/jpeg
.jfif	image/jpeg
.pic	image/pict
.pict	image/pict
.tif	image/tiff
.tiff	image/tiff
.mpeg	video/mpeg
.mpg	video/mpeg
.mov	video/quicktime
.moov	video/quicktime
.aif	audio/aiff
.aiff	audio/aiff
.wav	audio/wav
.ram	audio/x-pn-realaudio
.sit	application/x-stuffit
.bin	application/x-stuffit
.z	application/x-zip

Extension	Mime/Type
.zip	application/x-zip
.gz	application/x-gzip
.tar	application/x-tar

The noContext parameter allows you to tell the 4D Web server that you want to switch from “contextual” mode to “non contextual” mode or vice-versa.
To use the non contextual mode, pass True to noContext. To use the contextual mode, pass False. If the parameter is omitted, the contextual mode is used by default.
The references to 4D variables and 4DACTION tags in the page are always analyzed, whatever the mode.

Note Supporting the “non contextual” mode is new in 4D version 6.5. For more information, please refer to paragraph “Non Contextual Mode”, page 180.

▼ Refer to the example of the routine PICT TO GIF, page 110.

Web Context

Web Context → Boolean

Parameters	Type		Description
This command does not require any parameters			
Result	Boolean	←	True = Contextual mode False = Non contextual mode

This function must be called from a Web process. It returns a boolean that indicates if the Web connection is executing in contextual mode (True) or in non contextual mode (False).

Note Called from a process other than a Web process, this function always returns False.

The use of this function is advocated in the On Web Connection database method (for more information, please refer to paragraph “Changes in the On Web Connection Database Method”, page 191).

Note The support of the “non contextual” mode is new in 4D version 6.5. For more information, please refer to paragraph “Non Contextual Mode”, page 180.

▼ Here is an example of the On Web Connection database method:

```
If (Web Context)
   WithContext ($1;$2;$3;$4;$5;$6)
Else
   WithoutContext ($1;$2;$3;$4;$5;$6)
End if
```

See Also: PROCESS PROPERTIES

WEB CACHE STATISTICS

WEB CACHE STATISTICS (pages; hits; usage)

Parameters	Type		Description
pages	Text array	←	Names of the most consulted pages
hits	Longint array	←	Number of hits for each page
usage	Number	←	Percentage of the cache used

This command allows you to obtain information about the most consulted pages loaded in the Web server's cache. Consequently, these statistics only concern static pages, GIF pictures, JPEG pictures <100 KB and style sheets (.css).

Note For more information about setting the 4D Web server's cache, please refer to paragraph "Cache for Static Pages", page 172.

The command fills the pages Text array with the names of the most consulted pages. The hits Longint array receives the number of "hits" for each page. The usage parameter receives the percentage of the Web cache used by each page.

▼ Let's assume that you want to generate a semi-dynamic page that displays the statistics of the Web cache. For this, in a static HTML page named "stats.shtm"[1], you place the tag <!--4DACTION/STATS--> (cf. paragraph "4DACTION Tag", page 186). Then you insert two 4D variables, *vPages* and *vUsage*.

1. The contents of the pages with the suffix ".shtm" is always parsed, cf. paragraph "Semi-dynamic Pages", page 185.

In the project method *STATS*, you write the following code:

```
C_TEXT ($1)
ARRAY TEXT (pages;0)
ARRAY LONGINT (hits;0)
C_LONGINT (vUsage)

WEB CACHE STATISTICS(pages;hits;vUsage)
vPages:=Char(1)
For ($i;1;Size of array(pages))
     'For each page present in the cache
  vPages:=vPages+pages{$i}+"   "
          +String(hits{$i})+"<br>"
     'Insert the name of the page and the HTML code
End for
SEND HTML FILE("stats.shtm")
```

SEND HTTP REDIRECT

SEND HTTP REDIRECT (url {; *})

Parameters	Type		Description
url	String	→	New URL
*		→	If specified = URL is not translated, If omitted = URL is translated

This command allows you to transform a URL into another one.

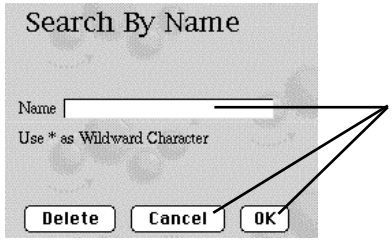
The url parameter contains the new URL that allows you to redirect the query. If this parameter is a url to a file, it must contain the reference to this file, like in non contextual mode, for example: SEND HTTP REDIRECT ("/MyPage.HTM").

When this command is called in contextual mode, the Web process is aborted and the Web license is freed just after being executed. The command prevails over commands that send data (SEND HTML FILE, SEND HTML BLOB, etc.) that may be in the same method.

This command also allows you to redirect a query to another Web server.

4D automatically encodes the URL's special characters. If you pass the * character, 4D will not translate them.

▼ You can use this command to execute custom queries in 4D by using static pages. Imagine that you have placed the following elements in a static HTML page:



In the On Web Connection database method part (or subroutine) that manages the non contextual mode, you insert the following code:

```
Case of
: ($1="/4dcgi/rech") ‘When 4D receives this URL
      ‘If the OK button has been used and the ‘name’ field contains aValue
  If ((bOK="OK") & (name # ""))
      ‘Change the URL to execute the query code,
      ‘placed farther down in the same method
    SEND HTTP REDIRECT("/4dcgi/rech?"+name)
  Else
      ‘Else return to the beginning page
    SEND HTTP REDIRECT("/page1.htm")
  End if
...
: ($1="/4dcgi/rech?@") ‘If the URL has been redirected
  ... ‘Put the query code here
End case
```

SET HTTP HEADER

SET HTTP HEADER (fields)

Parameters	Type		Description
fields	String	→	Fields in the HTTP header

This command allows you to set the fields in the HTTP header of the reply sent to the Web browser by 4D. It only has an effect in a Web process in non contextual mode.

This command allows you to manage “cookies”.

You pass the HTTP header fields to the fields parameter, of type Text, that you want to set.

Note The fields must always be separated by a cr/lf sequence (carriage return/line feed). For more information about the syntax, please refer to the R.F.Cs (*Request For Comments*) that can be found at the following Internet address: www.w3c.org.

If you do not specify a state, it will automatically be HTTP/1.0 200 OK.

The Charset, Expires, Content-Length, Content-Type, Date and Last-Modified fields are always set by 4D.

▼ Here is an example of a custom "cookie":

SET HTTP HEADER("SET-COOKIE: USER="+**String**(**Abs**(**Random**))+"; PATH=/")

OPEN WEB URL

OPEN WEB URL (url {; *})

Parameters	Type		Description
url	String	→	Startup URL
*		→	If specified = URL is not translated, If omitted = URL is translated

This command launches your Web browser and opens it on the URL passed in the url parameter.

If your Web browser was already opened when you execute this command:

- On Windows, an additional instance of the browser is executed and displays the specified page by url.
- On MacOS, the specified page by url replaces the current page in the browser.

If there is no browser on the volumes connected to the computer, this command has no effect.

4D automatically encodes the URL's special characters. If you pass the * character, 4D will not translate the URL's special characters. This option allows you to access and to send URLS of type "http://www.server.net/page.htm?q=something"

Note This command does not work when called from a Web process.

- When the following line of code is executed:

  OPEN WEB URL("file:///D:/web file.htm")

  The Web browser is launched and the URL is translated into "file:///D%3A/web%20file.htm"

- When the following line of code is executed:

  OPEN WEB URL("file:///D:/web file.htm";*)

  The Web browser is launched and the URL remains "file:///D:/web file.htm"

- The following line of code launches the browser and connects it to ACI's home page:

  OPEN WEB URL("http://www.aci-4d.com/")

SET HOME PAGE

SET HOME PAGE (homePage)

Parameters	Type		Description
homePage	String	→	Page name or HTML access path to the page or An empty string to not send the custom home page

This command allows you to define a custom home page for the current Web process. By default, in contextual mode, menu bar number 1 is usually the home page.

The defined page is linked to the Web process, you can therefore define the different home pages depending, for example, on the user that is connected. This page can either be static or semi-dynamic.

You pass the name of the HTML home page or the page's HTML access path to the homePage parameter.

To stop sending homePage as home page for the current Web process, execute SET HOME PAGE with an empty string ("") in homePage.

Note 4D version 6.5 also allows you to define a default home page in the Database Properties dialog box (For more information, please refer to paragraph "Defining a Default Home Page", page 167). In this case, the page applies to all the Web connections whatever the Web server's startup mode (contextual or non contextual).

Windows

Open form window

Open form window ({table;} formName {; type{; posH{; posV{; *}}}}) → WinRef

Parameters	Type		Description
table	Table	→	Table of the form or Default table, if omitted
formName	String	→	Name of the form
type	Longint	→	Window type
hPos	Longint	→	Horizontal position of the window
vPos	Longint	→	Vertical position of the window
*	*	→	Save current position and size of the window
WinRef	Longint	←	Window reference number

The Open form window command opens a new window using the size and resizing properties of the form formName.

Note that formName is not displayed in the window. If you want to display the form, you have to call a command which loads a form (ADD RECORD for example).

By default (if the type parameter is not passed), a plain window (type 8) with a close box is opened. Unlike the Open window command, no method is associated to the window's close box. Clicking on this close box cancels and closes the window, except if the On Close Box form event has been activated for the form. In this case, the code associated to the On Close Box event will be executed.

If formName is resizable, the window opened will contain a zoom box as well as a grow box.

Note To know the main properties of a form, use the GET FORM PROPERTIES command.

The optional type parameter allows you to specify a type for the window. You must pass one of the following predefined constants (placed in the “Open window” theme):

Constant	Type	Value
Plain window	Longint	8
Modal dialog box	Longint	1
Movable dialog box	Longint	5
Palette window	Longint	720

The optional parameter hPos allows you to define the horizontal position of the window. You can pass a defined position, expressed in points, to this parameter (refer to the Open window command) or one of the following predefined constants placed in the “Window position” theme:

Constant	Type	Value
Horizontally Centered	Longint	65536
On the Left	Longint	131072
On the Right	Longint	196608

The optional parameter vPos allows you to define the vertical position of the window. You can pass a defined position, expressed in points, to this parameter (refer to the Open window command) or one of the following predefined constants placed in the “Window position” theme:

Constant	Type	Value
Vertically Centered	Longint	262144
At the Top	Longint	327680
At the Bottom	Longint	393216

These parameters take into account the presence of the tool bar and menu bar as well as the current size of the application's window (on Windows).

If you pass the optional parameter *, the current position and size of the window are memorized when closed. When the window is reopened again, its previous position and size are respected. In this case, the vPos and hPos parameters are only used the first time the window is opened.

- The following statement opens a standard window with a close box and automatically adjusts it to be the same size as the "Input" form. Since the form has been defined as resizable, the window also has a grow and a zoom box:

  $winRef := **Open form window** ([Table1];"Enter")

- The following statement opens a floating palette in the upper left portion of the screen. This palette uses the last position it was in when the user closed it each time it is reopened:

  $winRef := **Open form window** ([Table1]; "Tools"; Palette window; On the Left; At the Top;*)

See Also: Open window, GET FORM PROPERTIES

GET FORM PROPERTIES

GET FORM PROPERTIES ({table;} formName; width; height{; numPages {; widthVar{; heightVar{; title}}}})

Parameters	Type		Description
table	Table	→	Table of the form or Default table, if omitted
formName	String	→	Name of the form
width	Longint	←	Width of the form (in pixels)
height	Longint	←	Height of the form (in pixels)
numPages	Longint	←	Number of pages in the form
widthVar	Boolean	←	True = Variable width False = Fixed width
heightVar	Boolean	←	True = Variable height False = Fixed height
title	Text	←	Title of the form's window

The GET FORM PROPERTIES command returns the properties of the form formName.

The width and height parameters return the form's width and height in pixels. These values are determined from the form's Default window size properties:

- If the form's size is **automatic**, its width and height are calculated so that all the form's objects are visible, by taking into consideration the horizontal and vertical margins that were defined.
- If the form's size is **set**, its width and height are those manually entered in the corresponding areas.
- If the form's size is **based on an object**, its width and height are calculated in relation to this object's position.

The numPages parameter returns the number of pages in the form, excluding page 0 (zero).

The widthVar and heightVar parameters indicate if the length and width of the form are resizable (the parameter returns True) or set (the parameter returns False).

The title parameter returns the title of the form's window as it was defined. If no name was defined, the title parameter returns an empty string.

See Also: Open form window

Modified Commands

This section describes the existing commands whose syntax has been modified in version 6.5. These commands are in alphabetical order.

All the new parameters are written in *italics*.

DISTINCT VALUES (Arrays)

DISTINCT VALUES (field; array)

Parameters	Type		Description
field	Field | Subfield	→	Indexable field or sub field to use
array	Array	←	Array to receive the values from the field

This command has been modified in the following two ways:

- DISTINCT VALUES now functions with all *indexable* fields, that is, whose type supports indexing without necessarily being indexed. However, executing this command on unindexed fields will be slower.
- Otherwise, DISTINCT VALUES now functions in transactions, which means that if a record is created during a transaction that has not yet finished, it will be taken into account by the command.

Note As this command now functions with indexed and unindexed fields, its execution mode can now be set by using the SET DATABASE PARAMETER command.

GOTO AREA (Entry Control)

GOTO AREA ({*;}object)

Parameters	Type		Description
*		→	If specified = object is an object name (string) If omitted = object is a field or a variable
object	object	→	Object name (if * specified) or Field or Variable (if * omitted)

The syntax of this command has been modified so that you can choose the parameters:

- either the field or variable reference desired (which is how the command has always functioned).
  Example: GOTO AREA ([Personnel]Name)
- or the object name.
  Example: GOTO AREA (*; "MyVar")

Note This command only functions in input forms.

Open document (System Documents)

Open document (docName{; type{; mode}}) → DocRef

Parameters	Type		Description
docName	Alpha	→	Name of the document
type	Alpha	→	Document type
mode	Integer	→	Document's opening mode
Result	DocRef	←	Document's reference number

The optional parameter mode has been added to this command. It allows you to define how docName is to be opened.

Four different open file modes are possible. 4th Dimension proposes the following predefined constants:

Constant	Type	Value
Read and Write (*default value*)	Integer	0
Write Mode	Integer	1
Read Mode	Integer	2
Get Pathname	Integer	3

If you pass 3 to mode, the function returns ?00:00:00? (no document reference). The document is not opened but the *Document* and *OK* system variables are updated:

- *OK* is equal to 1.
- *Document* contains either the name or the full access path and the name of docName, depending on the value passed in docName (if you passed a file name, *Document* contains this name, if you passed a full access path, *Document* contains this full access path).

Note If the file defined in docName is not found or if you pass an empty string in docName, an open file dialog box appears. If it is validated, *Document* and *OK* are updated as above. If it is not validated, *OK* is equal to 0.

▼ You can now read a document that is already open in write mode:

vDoc:=**Open document** ("PassFile";"TEXT") ` the file is open

Before the file is closed, it is possible to consult it in read only mode:

vRef:=**Open document** ("PassFile";"TEXT";Read Mode)

PROCESS PROPERTIES (Processes)

PROCESS PROPERTIES (process; name; state; time{; procVisible{; uniqueID{; origin}}})

Parameters	Type		Description
process	Longint	→	Process number
name	Alpha	←	Process name
state	Integer	←	Process state
time	Integer	←	Cumulated process execution time
procVisible	Boolean	←	Visible (True) or hidden(False)
uniqueID	Integer	←	Unique process number
origin	Longint	←	Origin of the process

This command now accepts two new optional parameters: uniqueID and origin.

uniqueID returns the unique process number.
Actually, from the 4D version 6.5, each process has attributed to it a process number as well as a **unique process number** per session. The unique number allows you to differentiate between two processes or two process sessions. It corresponds to the process number having been started during 4th Dimension's session.

origin returns a value that describes the origin of the process. 4th Dimension offers the following predefined constants (in the theme Process Type):

Constant	Type	Value
Web Process with License	Longint	-11
Other 4D Process	Longint	-10
External Task	Longint	-9
Event Manager	Longint	-8
Apple Event Manager	Longint	-7
Serial Port Manager	Longint	-6
Indexing Process	Longint	-5
Cache Manager	Longint	-4
Web Process with no License	Longint	-3
Design Process	Longint	-2
User or Custom Menus Process	Longint	-1
None	Longint	0
Created from Programming	Longint	1
Created from Menu Command	Longint	2
Created from User Mode	Longint	3
Other User Process	Longint	4

Note 4D's internal processes return a negative value and the processes generated by the user return a positive value.

Semaphore (Process (Communications))

Semaphore (semaphore {; tickCount}) → Boolean

Parameters	Type		Description
semaphore	Alpha	→	Semaphore to test and to position
tickCount	Integer	→	Maximum waiting time
Result	Boolean	←	semaphore has been correctly created (False) or semaphore was already created (True)

The optional parameter tickCount has been added to this command. This parameter allows you to specify a waiting time (in ticks) if semaphore has already been positioned.

In this case, the function will wait either for the semaphore to be freed or the waiting time to expire before returning True.

If you do not want to use this option, pass 0 to tickCount or omit this parameter completely.

▼ Consider the following code:

```
t0:=Tickcount
onContinue:= True
While (Semaphore( "flag") & onContinue)
   DELAY PROCES(Current process;10)
   If (Tickcount - t0 > 600)
      onContinue:= False
   End if
End while
If (Not(Semaphore("flag")))
   `Let's do something...
End if
```

... could be replaced by:

```
If (Not(Semaphore("flag";600))) `we don't wait more than 10 seconds
   `Let's do something...
End if
```

SEND HTML FILE (Web Server)

SEND HTML FILE (HTMLfile)

Parameters	Type		Description
HTMLfile		→	HTML access path to HTML file or an empty string to terminate SEND HTML FILE

The SEND HTML FILE command now modifies the *OK* system variable. If the file to be sent exists and if the timeout has not run out, *OK* is equal to 1. Otherwise, it is equal to 0.

SET INDEX (Structure Access)

SET INDEX(field; index{; mode}{; *})

Parameters	Type		Description
field	Field | Subfield	→	Field from which to create or delete the index
index	Boolean	→	Create the index (True) or delete the index (False)
mode	Longint	→	Indexing mode (in percentage)
*		→	Asynchronous indexing if * is passed

The syntax of this command has been modified so that it can take advantage of the new indexing mode offered by 4D version 6.5 (For more information, please refer to paragraph "New Index Mode", page 195).

If you pass *mode*, the command will use the new indexing system. Otherwise, the standard mode will be used. The *mode* parameter is only called if the command is able to create the index (that is if the *index* parameter is True).

You must pass an Integer value that represents a percentage to the *mode* parameter. This value allows you to indicate the usage type for which you want the index to be most efficient. It must be between the following limits:

- *mode* = 0: the index will be most efficient when adding or inserting records.
- *mode* = 100: the index will be most efficient when performing queries.

▼ The following code allows you to index the [Clients]Name field. This field is mainly used for queries.

SET INDEX([Clients]Name;**True**;100)

▼ You want to index the [Prospects]Name field. This field will be used while adding and inserting records as well as when performing queries. However, the first usage type will be the most frequent.

SET INDEX([Prospects]Name;**True**;30)

SET WEB TIMEOUT (Web Server)

SET WEB TIMEOUT(timeout)

This command does not affect anymore all the Web Connection processes, when it is called from a Web process: in this case, the value of timeout is applied to this process only.

If the command is not called from a Web process, it works as in 4D version 6.0.x.

Form Events

Three new form events have been added to 4D version 6.5:

- On After Keystroke (moreover, in order to harmonize the names of the form events that manage the key strokes, the On Keystroke event has been renamed and is now called On Before Keystroke).
- On Timer
- On Resize

Moreover, the form event On Clicked can now be generated in an output form.

On After Keystroke

The new form event On After Keystroke allows you to better control and filter the characters that could be entered in an area. It is meant to be used mainly with the new command Get edited text, and in conjunction with the On Before Keystroke event (the new name for On Keystroke, see above).

This form event functions in the following manner:

- As soon as a character is typed, the event On Before Keystroke is first generated. In this event, the Get edited text function returns the contents of the area **before** the last keystroke. For example, if the area contains "PA" and the user types "R", Get edited text returns "PA" in the On Before Keystroke event. If the area doesn't contain anything before the user begins to type, Get edited text returns an empty string.
- Then, the new form event On After Keystroke is generated. In this event, the Get edited text function returns the contents of the area **including** the last character entered.
For example, if the area contained "PA" and the user typed "R", Get edited text returns "PAR" in the On After Keystroke event.

These two events are only generated in the object methods concerned. They can also be activated by the POST KEY command.

On Timer

This new form event is generated each time the number of ticks defined by the SET TIMER command is attained while a form is being displayed on screen.

The test of the On Timer form event must be preceded by a call to the SET TIMER command. This command's role is to activate the generation of the event and to define the interval time (in ticks) at the end of which each event is generated (for more information, please refer to the description of the routine SET TIMER, page 99).

The SET TIMER command must be called in the form method or in a form's object method.
The On Timer form event can only be called in the form method.

Note The 4D Web server takes advantage of this feature to allow you to update dynamic HTML pages on the browsers "on the fly". For more information, please refer to the description of the routine SET TIMER, page 99.

On Resize

This new form event is generated when the form window is resized. Note that a window can be resized in many different ways and the event is generated when 4D must resize the form objects.

The On Resize form event is therefore generated in the following cases:

- Enlarging or reducing the window by using the mouse: using the **resize box**, moving the edges of the window (on Windows), clicking on the **Enlarging/Reducing boxes**,...
- Calling the MINIMIZE WINDOW or MAXIMIZE WINDOW command.
- Opening a form, going from output to input mode and vice-versa, opening a subrecord, etc.

The combination of this form event and the MOVE OBJECT/GET OBJECT RECT commands allows you to perfectly control the positioning of the objects in a form.

On Clicked (modified)

The On Clicked form event is now generated in an output form (list mode) displayed by the MODIFY SELECTION or DISPLAY SELECTION commands, when the user clicks in the body of the form. This new feature allows you to, for example, enable or disable buttons in the footer or menu items depending on the selected records (you only need to test the UserSet set when the On Clicked form event is generated).

Note This new feature does not apply to lists displayed in User mode or to subforms.

On Double Clicked (modified)

The On Double Clicked form event is now generated in an output form (list mode) displayed by the MODIFY SELECTION or DISPLAY SELECTION commands, when the user double-clicks on a record.

This event allows you to change the input form on the fly. You can also refuse access to the input form by using the FILTER EVENT command.

Note This new feature does not apply to lists displayed in User mode or to subforms.

Debugger

4D version 6.5's debugger has been modified and improved in terms of how it functions as well as the information that it provides:

- Additional information is now displayed.
- The window's management has been modified (it is now possible to display several debugger windows simultaneously).
- New functions: saving the window's current parameters, selections of code.
- The keyboard combination to switch to Trace mode has been modified.

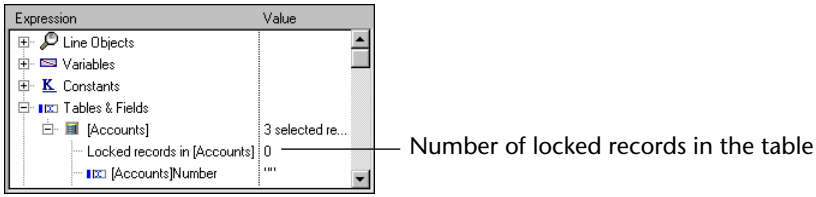
Additional Information

New information is now available in the debugger.

Note Most of the information offered in the debugger's expression window is accessible in all of 4D's modes, with the help of the Runtime Explorer (please refer to paragraph "Runtime Explorer", page 49).

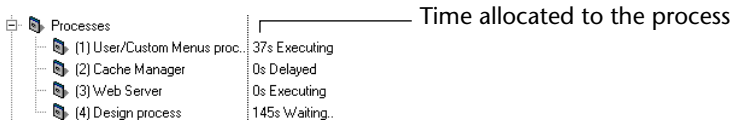
Number of locked records

The “Tables & fields” list contains a new line, placed above each table name, called “Locked records in [TableName]”. It indicates the current number of locked records in the table.



Time used by each process

In the process list, the time used by each process is now displayed.



Cache Statistics

This new list displays statistics regarding the use of tables, index pages, and named selections that are loaded in 4D’s cache.

Window Management

Managing the debug window has been modified. Now, 4D 6.5’s debug window is placed at the same “level” as the active window. When many other windows are displayed, going from one window to another no longer makes the debug window disappear. Consequently, it is now possible to simultaneously open different windows in the Trace mode. Moreover, if the window that calls the debugger is modal, the debug window also becomes modal.

New Functions

The debugger offers new functions to improve how the executing of the methods is controlled as well as its ease of use.

Saving a window’s configuration

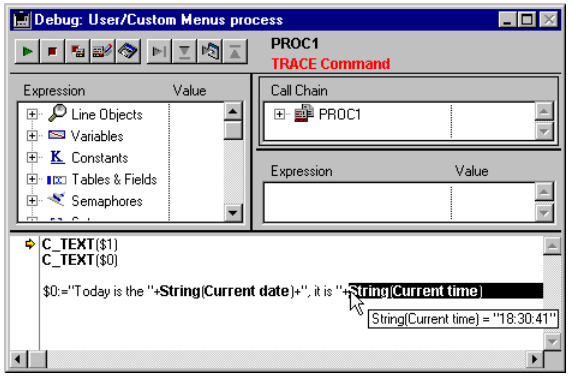
Displaying a new debug window uses the same configuration (size and position of the window, placing of the division lines and contents of the area that evaluates the expressions) as the last window displayed in the same session.

The current configuration of the debug window can be saved, so that it will be used by default each time the database is opened. In this case, you must click the Save button that is in the debug’s control panel or click the **F3** key.

These parameters are stored in the database’s structure file.

Selecting and evaluating portions of text

It is now possible to select a portion of the text in the area displaying the code being executed. In this case, when the cursor is placed above the selected text, a tip displays the selected object's value.



When you click on a variable name or field, it is automatically selected.

You can place the selected text in the expression evaluation area in three different ways:

- by simply dragging and dropping: click on the selected text, drag it and drop it in the evaluation area.
- by clicking on the selected text while holding down the **Ctrl** (Windows) or **Command** (MacOS) key.
- by using the **Ctrl+D** (Windows) or **Command+D** (MacOS) key combinations.

Entering Trace mode

A new option in the User mode allows you to directly trace a method. The "manual" interruption procedure of a method has also been changed.

Note You can now set the stop points directly in the method editor. This new option is described in the paragraph "Inserting Break Points", page 68.

Executing a method in trace mode

You can now directly trace a method in the User mode. In previous versions of 4D, you had to insert the TRACE command at the beginning of the method.

To do so, click the **Debug** button in the method execution dialog box after selecting the method name (for more information, please refer to the paragraph “Executing a Method”, page 96).

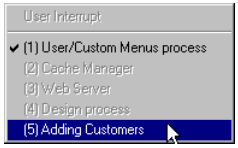
4D Server The **Debug** button is disabled when the option “On 4D Server” has been chosen from the “Execute” menu.

Shortcut to interrupt a method has been modified

The way to interrupt a method from executing so that you can trace it has been modified. In previous versions of 4D, you had to use the **Alt+Click** (Windows) or **Option+Click** (MacOS) key combination. Now, you must use the following key combinations:

- **Alt+Shift+Right Click** on Windows
- **Control+Option+Command+Click** on MacOS

When you execute this action, a pop-up menu listing all the processes being executed appears. You can then select the process you wish to trace.



A mark is placed next to the process name in the window in which the user clicked. The processes that were unable to be traced are shown in gray.

Note In previous versions of 4D, when an ON ERR CALL was set and a method was interrupted by **Alt+Click** (or **Option+Click**), error 1006 was returned to 4D’s *Error* system variable. With this new feature, no error is generated.

5 Web Server

In 4D version 6.5, the integrated Web server has received numerous new features and improvements. The new features are organized in the following areas:

- **Connection Security** The security of the connections has been reinforced thanks to new options as well as a new database method, On Web Authentication. The "Generic Web User" option simplifies access management inside the database. Moreover, the ability to define a default HTML root folder allows you to restrict access to files on disk.
- **Customizing Sessions** Numerous parameter possibilities for the Web server have been added: you can define a default home page, use Javascript to control data entry, use the new context referencing mode, and define how HTML characters are to be converted. 4D 6.5 allows you to use a cache for static pages, pictures, and style sheets. Also, you can set the IP address on which the server must receive HTTP queries.
- **Information about the Web site's functioning** Particular URLs allow you to obtain information about how the Web site is functioning. It is now also possible to generate a log file containing information about the connections to the Web site.
- **HTML Support** Converting 4D forms in HTML and, more generally, supporting HTML have been improved in 4D 6.5.
- **Non contextual mode** The non contextual mode is a major innovation to 4D 6.5 Web server. It allows you to free yourself from the context management and the 4D Web server becomes a standard HTTP server. In this mode, you can take advantage of 4D's capabilities by using "semi-dynamic" pages.

Note Many commands have been added and modified in the "Web server" theme of 4D 6.5's programming language. For a complete description of these commands, please refer to chapter "Language", page 97.

Connection Security

The security of 4D Web server connections has been reinforced by the following elements:

- A new password management system for Web access.
- A new database method: On Web Authentication.
- The definition of a “Generic Web User”.
- The setting of a default HTML root folder.

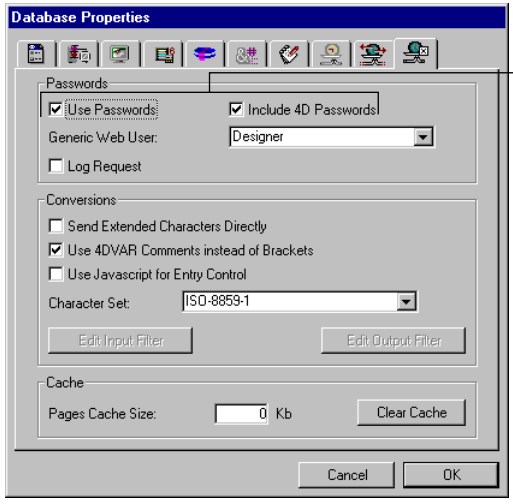
Password Options

You can now define, in the Database Properties dialog box, the access control system you want to apply to your Web server.

▶ To define the password system:

1 In the Database Properties dialog box, click the “Web Server II” tab.

The following window appears :



Two options are available to you: **Use Passwords** and **Include 4D Passwords**. The second check box is only active if the first one has been selected.

- **Use Passwords**: activates the Web server's password system. For each connection, a dialog box appears on the browser so that the user can enter his name and password.
  These two values, as well as the connection parameters (IP address and port, URL...) are sent to the On Web Authentication database method so that you can process them.

Note In this case, if the On Web Authentication database method doesn't exist, the connection is refused.

- **Include 4D Passwords**: allows you to use, instead of or in addition to your own password system, 4D's database password system (as defined in 4D).

The Web access control system results from the combination of these options and the On Web Authentication database method. For more information about this database method, please refer below to the paragraph "On Web Authentication Database Method", page 157. For a complete description of 4D Web server's access system, please refer to paragraph "4D Web Server's Access System", page 160.

2 Define your options and click the OK button to accept the dialog box.

On Web Authentication Database Method

A new database method has been added to 4D version 6.5: On Web Authentication. This database method is in charge of managing accesses to the Web server.

This method receives six parameters: $1, $2, $3, $4, $5, and $6 (like the On Web connection database method, cf. paragraph "Changes in the On Web Connection Database Method", page 191). The contents of these parameters are as follows:

Parameters	Type	Description
$1	Text	URL
$2	Text	HTTP header
$3	Text	IP address of the Web client
$4	Text	IP address of the server
$5	Text	User name
$6	Text	Password

Note All the On Web Authentication database method's parameters will not be eventually filled in. The information received by the database method depends on the options that you have previously selected in the Database Properties dialog box (cf. paragraph "4D Web Server's Access System", page 160).

For more information about the $1 and $2 parameters, refer to the description of the On Web connection database method in the 4D documentation.

The $3 parameter receives the IP address of the browser's machine. This information can allow you to distinguish between Intranet and Internet connections.

The $4 parameter receives the IP address of the 4D Web server. 4D version 6.5 allows for *multi-homing*, which allows you to exploit machines with more than one IP address. For more information, please refer to paragraph "Defining the IP Address for the HTTP Queries", page 173.

The $5 and $6 parameters receive the user name and password entered by the user in the standard identification dialog box displayed by the browser. This dialog box appears for each connection, if the **Use Passwords** option has been selected in the Database Properties dialog box.

The On Web Authentication database method returns a boolean in $0:

- If $0 is True, the connection is accepted.
- If $0 is False, the connection is refused.

The On Web Connection database method is only executed if the connection has been accepted by On Web Authentication.

Do not call any interface elements in the On Web Authentication database method (ALERT, DIALOG, etc.), otherwise it will be interrupted and the connection will be refused. The same is true if an error occurs while the database method is being executed.

▼ Here is a typical example of the On Web Authentication database method:

```
C_TEXT($5;$6;$3;$4)
C_TEXT($user;$password;$BrowserIP;$ServerIP)
C_BOOLEAN($4Duser)
ARRAY TEXT($users;0)
ARRAY TEXT($nums;0)
C_LONGINT($upos)
C_BOOLEAN($0)

$0:=False

$user:=$5
$password:=$6
$BrowserIP:=$3
$ServerIP:=$4

  `For security reasons, refuse names that contain @
If (WithWildcard($user) | WithWildcard($password))
  $0:=False
     `The WithWildcard method is described below
Else
     `Check to see if it's a 4D user
  GET USER LIST($users;$nums)
  $upos:=Find in array($users;$user)
  If ($upos > 0)
     $4Duser:=Not(Is user deleted($nums{$upos}))
  Else
     $4Duser:=False
  End if

  If (Not($4Duser))
       `It is not a user defined 4D, look in the table of Web users
     QUERY([WebUsers];[WebUsers]User=$user;*)
     QUERY([WebUsers]; & [WebUsers]Password=$password)
     $0:=(Records in selection([WebUsers]) = 1)
  Else
     $0:=True
  End if
End if
     `Is this an intranet connection?
If (Substring($BrowserIP;1;7) # "192.100.")
  $0:=False
End if
```

▼ The *WithWildcard* method is as follows:

```
C_INTEGER($i)
C_BOOLEAN($0)
C_TEXT($1)

$0:=False
For($i;1;Length($1))
  If (Ascii(Substring($1;$i;1)) = Ascii("@"))
     $0:=True
  End if
End for
```

4D Web Server's Access System

The system that filters connections to the 4D Web server depends on the combination of two parameters:

- The Web password options in the Database Properties dialog box (cf. paragraph "Password Options", page 156),
- The existence of the On Web Authentication database method (cf. paragraph "On Web Authentication Database Method", page 157).

No option is selected

- If On Web Authentication exists, it is executed, besides $1 and $2, only the IP addresses of the browser and the server ($3 and $4) are returned, the user name and password ($5 and $6) are left empty. In this case, you can filter the connections according to the browser's IP address and/or the server's IP address.
- If On Web Authentication doesn't exist, the connection is automatically accepted.

The "Use Passwords" option is selected and the "Include 4D Passwords" option is not selected.

- If On Web Authentication exists, it is executed and all its parameters are returned. You can therefore filter more precisely the connections according to the user name, password, and/or the browser's or Web server's IP address.
- If On Web Authentication doesn't exist, the connection is automatically refused and a message indicating that the Authentication method doesn't exist is sent to the browser.[1]

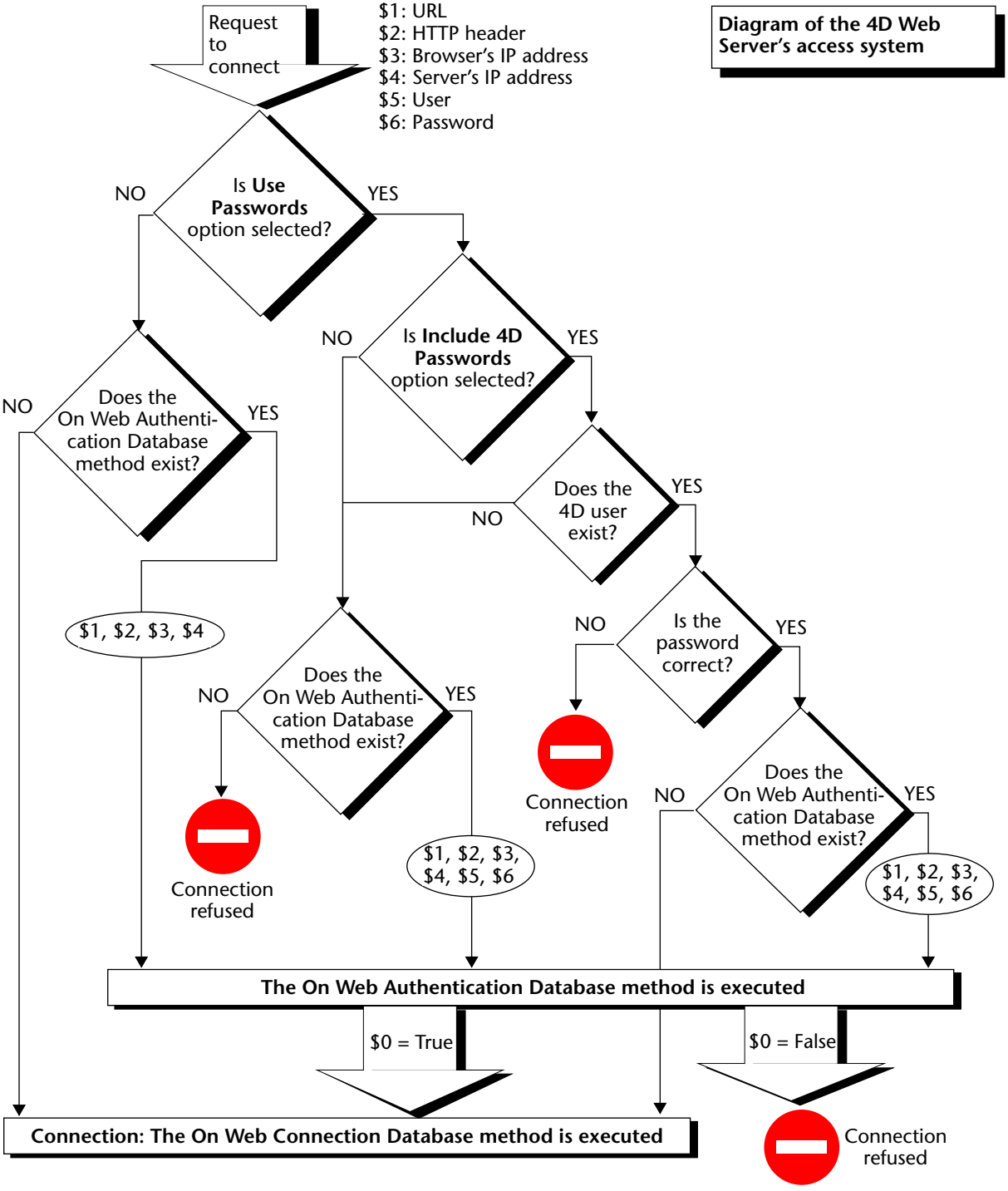
The "Use Passwords" and "Include 4D Passwords" options are selected.

Note Databases created with a previous version of 4D 6.5 are opened with these options selected by default.

- If the user name sent by the browser exists in the table of 4D users and the password is correct, the connection is accepted.[2]
  If the password is incorrect, the connection is refused.
- If the user name sent by the browser doesn't exist in 4D, two results are therefore possible:
  - if On Web Authentication exists, the parameters $1, $2, $3, $4, $5, and $6 are returned. You can therefore filter the connections according to the user name, password, and/or the browser's or Web server's IP address.
  - if On Web Authentication doesn't exist, the connection is refused.

4D Web server's access system is summarized in the following diagram:

1. If the user name sent by the browser is an empty string and if the On Web Authentication database method doesn't exist, a password dialog box is sent to the browser.
2. If the user name sent by the browser exists in 4D, the $6 parameter (the user's password) is not returned for security reasons.


Request to connect
$1: URL
$2: HTTP header
$3: Browser's IP address
$4: Server's IP address
$5: User
$6: Password
Diagram of the 4D Web Server's access system
Is Use Passwords option selected?
NO
YES
Is Include 4D Passwords option selected?
NO
YES
Does the On Web Authentication Database method exist?
NO
YES
Does the 4D user exist?
NO
YES
Is the password correct?
NO
YES
Does the On Web Authentication Database method exist?
NO
YES
Does the On Web Authentication Database method exist?
NO
YES
$1, $2, $3, $4
$1, $2, $3, $4, $5, $6
$1, $2, $3, $4, $5, $6
Connection refused
Connection refused
Connection refused
The On Web Authentication Database method is executed
$0 = True
$0 = False
Connection: The On Web Connection Database method is executed

A Security Note about Robots

Certain robots (query engines, spiders...) scroll through the Web servers and static pages. If you want robots to be able to access your entire site, you can define which URLs they are not allowed to access.

To do so, put the ROBOTS.TXT file at the server's root. This file must be structured in the following manner:

```
User-Agent: <name>
Disallow: <URL> or <beginning of the URL>
```

- For example:

```
User-Agent: *
Disallow: /4D
Disallow: /%23%23
Disallow: /GIFS/
```

"User-Agent: *" means that all robots are affected.
"Disallow: /4D" means that robots are not allowed to access URLs beginning with /4D.
"Disallow: /%23%23" means that robots are not allowed to access URLs beginning with /%23%23.
"Disallow: /GIFS/' means that robots are not allowed to access the /GIFS/ folder or its subfolders.

- Another example:

```
User-Agent: *
Disallow: /
```

In this case, robots are not allowed to access the entire site.

Generic Web User

You can designate a user, previously defined in the 4D password table, as a "Generic Web User." In this case, each browser that connects to the database can use the access authorizations and restrictions associated to this generic user. You can therefore simply control the browser's access to the different parts of the database.

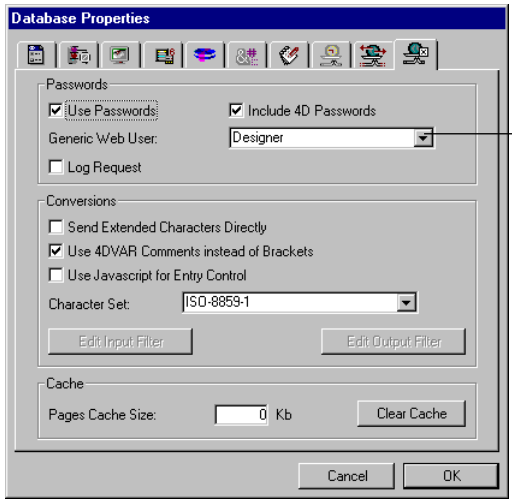
Note Do not confuse this option, which allows you to restrict the browser's access to different parts of the database (tables, menus, etc.), with the Web server's connection control system, managed by the password system and the On Web Authentication database method (cf. paragraph "Password Options", page 156).

► To define a Generic Web User:

1 **In the Design mode, create a user in the Password Editor.**
You can associate a password to the user if you wish.

2 **In the different 4D editors, authorize or restrict access to this user.**

3 **In the Database Properties dialog box, click on the "Web Server II" tab.**
The following page appears:



By default, the Generic Web user is the Designer and the browsers have full access to the entire database.

4 **Choose a user in the "Generic Web User" list.**

5 **Accept the dialog box.**
All the Web browsers that are authorized to connect to the database will benefit from the access authorizations and restrictions associated to this generic Web user (except if the "Include 4D Passwords" option has been selected and the user that connects does not exist in the 4D password table, see below).

Interaction with the Web Password System

The **Use Passwords** option does not influence how the generic Web user operates. Whatever the state of this option, the access authorizations and restrictions associated to the "Generic Web User" will be applied to *all* the Web browsers that are authorized to connect to the database.

However, when the **Include 4D passwords** option is selected, two possible results can occur:

- The user's name and password don't exist in 4D's password table. In this case, if the connection has been accepted by the On Web Authentication database method, the generic Web user's access rights will be applied to the browser.
- If the user's name and password exist in 4D's password table, the "Generic Web User" parameter is ignored. The user connects with his own access rights.

Define a HTML Root Folder by Default

This new option in Database Properties allows you to define the folder in which 4D will search for the static HTML pages and the pictures to send to the browsers.

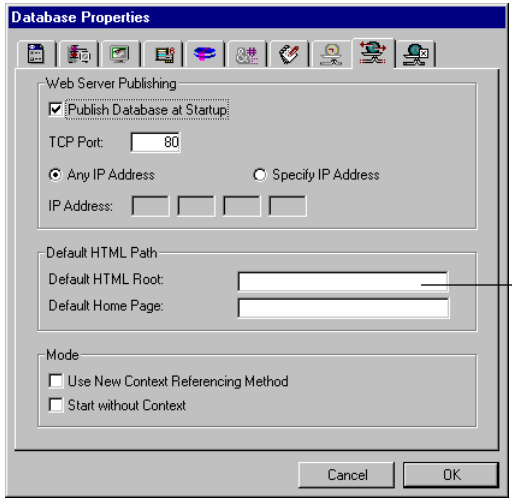
Moreover, the HTML root folder defines the hierarchical level above which the files will not be accessible.
This access restriction applies to URLs sent to Web browsers as well as to 4D's Web server commands, such as SEND HTML FILE. If a URL is sent to the database by a browser or if a 4D command tries to access a file located above the HTML root folder, an error is returned indicating that the file has not been found.

By default, the HTML root folder is the one that contains the structure file of the database. Be careful because in this case there are no access restrictions (users can access all the volumes).

► To define a HTML root folder by default:

1 In the Database Properties dialog box, click on the "Web Server I" tab.

The following page appears:



2 In the "Default HTML Root" entry area, enter the access path of the folder that you wish to define.

The access path entered in this dialog box is relative because it is established from the folder containing the structure of the database.
For multi-platform compatibility of your databases, the 4D Web server uses particular writing conventions to describe access paths. The syntax rules are as follows:

- folders are separated by a slash ("/")
- the access path ends with a slash ("/")
- to "go up" one level in the folder hierarchy, enter ".." (two periods) before the folder name
- the access path must not start with a slash ("/") (except if you want the HTML root folder to be the database's folder, see below).

For example, if you want the HTML root folder to be the "Web" subfolder in the "4DDatabase" folder, enter "**4DDatabase/Web/**". If you want the HTML root folder to be the database folder, but the access to the folders above to be forbidden, enter "/" in the area. For a completely free access to the volumes leave the "Default HTML Root" area empty.

3 Accept the dialog box.

Note When the HTML root folder is modified in the Database Properties dialog box, the cache is cleared so as to not store files whose access is restricted.

Database Properties and SET HTML ROOT

Once a HTML root folder is defined in the Database Properties dialog box, you can always modify it by using the SET HTML ROOT command. The modification therefore only applies to the current Web process for the worksession. The cache of the HTML pages is therefore cleared.

However, the SET HTML ROOT command takes into account the default HTML root folder when it is defined in Database Properties. If the folder defined in the Database Properties dialog box is "WebPages/" and if you pass the instruction SET HTML ROOT("Folder"), the default HTML root folder becomes "WebPages/Folder/". Also in this case, the access restrictions are only maintained for the folders located above the "WebPages" folder.

Note The SET HTML ROOT command has no effect when the Web server is in non contextual mode (For more information, please refer to paragraph "Non Contextual Mode", page 180).

Customizing Web Sessions

4D 6.5 allows you to customize your Web sessions even more depending on your needs. The following options are available:

- Defining a default home page
- Using Javascript for data entry control
- New mode to insert 4D variables in static pages
- Modifying the context referencing mode
- Setting the conversion of HTML characters and defining Web filters
- Using a cache for static pages
- Defining the IP address from which the server receives HTTP queries

Defining a Default Home Page

You can now define a default home page for all the browsers that connect to the database. This page can be static or semi-dynamic[1].

If you specify a default home page, this page is sent to each browser that connects to the database, no matter which mode (contextual or non contextual) has been defined for the Web sessions[2]. Unlike

1. For more information, please refer to the paragraph "Semi-dynamic Pages", page 185.
2. For more information, please refer to the paragraph "Defining the Non Contextual Mode at Startup", page 184.

previous versions of 4D, the current menu bar is not sent to the browser in contextual mode.

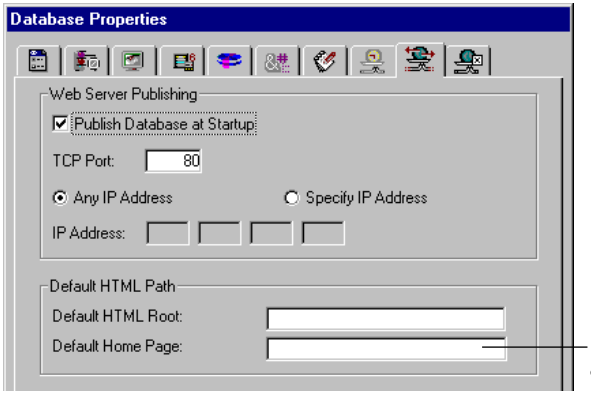
By default, no home page is defined. If you do not specify a custom home page, the behavior of the Web server will differ depending on the startup mode:

- If the Web Server starts up in contextual mode (by default), the current menu bar — by default, menu bar number 1 — is sent as the home page, as in previous versions of 4D.
- If the Web server starts up in non contextual mode, only the On Web Connection database method is called. It's up to you to process the query procedurally.

▶ To define a default home page:

1 **In the Database Properties dialog box, click on the "Web Server I" tab.**

The following page appears:



2 **In the "Default Home Page" entry area, enter the access path relative to the folder that you want to define.**

The access path entered in this dialog box is relative because it is established from the folder containing the structure file of the database. In order to ensure multi-platform compatibility of your databases, the 4D Web server uses particular writing conventions to define access paths. The syntax rules are as follows:

- folders are separated by a slash ("/")
- the access path ends with a slash ("/")
- to "go up" one level in the folder hierarchy, enter ".." (two periods) before the folder name
- the access path must not start with a slash ("/")

For example, if you want the default home page to be “MyHome.htm”, and it is located in the “Web” folder, enter “**Web/MyHome.htm**”.

3 **Accept the dialog box.**

Note You can also define a default home page for each Web process by using the routine SET HOME PAGE, page 139.

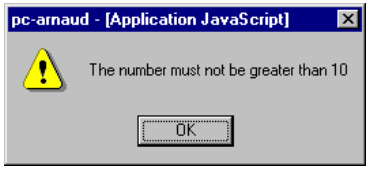
Using Javascript for Data Entry Controls

In 4D 6.5, a part of the data entry controls can be done on the browsers by using automatic Javascripts.

On the browser, the data entry controls and the data types (fields or variables) to which they can be applied are as follows:

- minimum value (for numeric values)
- maximum value (for numeric values)
- mandatory value (for numeric and alphanumeric values)

Generated Javascripts, which are small in size, display alert dialog boxes without preventing the user from accepting a data entry (it is still 4D’s responsibility).
Actually, if a data entry area contains an incorrect value, an alert message is displayed on the browser when the user clicks on a button (OK, Cancel, etc.):



Once the alert dialog box is validated, if the user clicks a second time on the button, the button’s action is then taken into account.

The complete data entry control is done on the Web server (as in previous versions of 4D), in User and Custom Menus mode.

► To use the Javascript data entry controls:

1 **In the Database Properties dialog box, click on the “Web Server II” tab.**

2 **Select the “Use Javascript for Entry Control” option.**
By default, this option is not selected.

3 **Click the OK button.**

New Mode to Insert 4D Variables in Static Pages

The **Use 4DVAR Comments instead of Brackets** option, which is on the "Web server II" page of Database Properties, allows you to define the notation to use when inserting 4D variables on static pages.

- When the option is checked, the syntax you need to use is the standard HTML notation (<!--4DVAR MAVAR-->)[1].
- When the option is not checked (default value), the syntax you need to use is the notation with square brackets ([MAVAR]) — which is a proprietary solution.

New Mode to Reference Contexts

By default, the 4D Web Server sends the number of the current context to the browser for each of the user's action. For instance, if a page contains two paragraphs and a picture, 4D will send the context number three times.

It is now possible to change this operation, and then to accelerate the speed at which the pages are being sent. With the new context referencing mode, the context number is placed in the basic URL of the document. In this way, it is not repeated for each object.

► To make use of the new context referencing mode:

1 **In the "Web Server I" page of Database Properties, select the "Use New Context Referencing Mode" option.**

By default, this option is not selected.

2 **Click the OK button and reopen your database so that the modification takes effect.**

Directly Sending Extended ASCII Characters

By default, the 4D Web server converts the extended ASCII characters in the dynamic and static Web pages according to HTML standards before sending them. They are then interpreted by the browsers.

You can set the Web server so that the extended ASCII characters are sent "as is", without converting them into HTML entities. This option has shown a speed increase on most foreign operating systems (especially the Japanese system).

► To allow the 4D Web server to directly send extended ASCII characters:

1 **In the Database Properties dialog box, click on the "Web Server II" tab.**

---

1. A space character must be inserted between 4DVAR and the variable name.

2 **Select the "Send Extended Characters Directly" option.**

3 **Accept the dialog box.**

Modifying Conversion Character Sets in 4D

You can now directly modify the ASCII character conversion tables (Web filters) when inputting and outputting data in 4D. This feature was previously available in Customizer Plus.

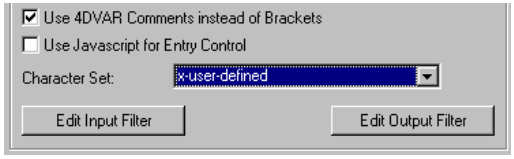
Modifying the conversion tables is possible only when the "x-user-defined" character set is selected in the Database Properties dialog box.

► To modify the ASCII character conversion tables:

1 **In the Database Properties dialog box, click the "Web Server II" tab.**

2 **In the "Character Set" scrolldown list, choose the "x-user-defined" option.**

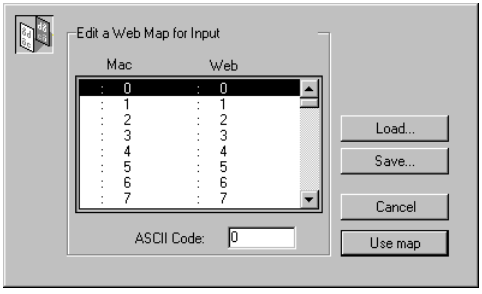
The **Edit Input Filter** and **Edit Output Filter** buttons are now enabled.



3 **Click the button that corresponds to the filter that you want to modify.**

Input filter interprets characters sent by the browser to the 4D Web server and **Output filter** interprets characters sent by the 4D Web server to the browser.

The following dialog box appears:



Note This dialog box looks similar to 4D's import/export filters dialog box. However, the two are independent: the Web filters are NOT in any way related to the import/export filters.

4 **In the scrollable area, look for and click on the Mac character that you want to filter.**
OR
Load a previously saved Web filter by clicking the "Load" button (then go to step 8).

5 **In the "ASCII Code" entry area, enter the character's new ASCII code.**

Note To ensure database compatibility, the default values of the filters are identical to those used in 4D version 6.0.6 (from the "MapC" — if defined with Customizer Plus).

6 **Repeat this operation for all the characters you want to filter.**

7 **Click the "Save" button to save the filter, if you wish.**

8 **Click the Use map button.**

9 **Click the OK button in the Database Properties dialog box.**
The modified input and/or output Web filters are now active.

Cache for Static Pages

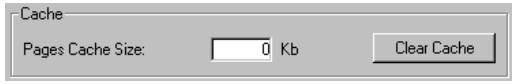
The 4D Web Server now has a cache that allows you to load static pages, GIF images, JPEG images (<100 kb) and style sheets (.css files) in memory, as they are requested.
Using the cache allows you to significantly increase the Web server's performance when sending static pages.

The cache is shared between all the Web processes. You can set the size of the cache in Database Properties.

By default, the cache of the static pages is not enabled (its size is equal to 0).

▶ To enable the cache of the static pages:

1 **In the "Web server II" page of the Database Properties dialog box, set a value (to be expressed in Kb) for the cache of the static pages.**



The value you set depends on the number and size of your Web site's static pages, as well as the resources that the host machines has at its disposal.

Note While using your Web database, you can check the performance of the cache by using the routine WEB CACHE STATISTICS, page 135. If, for example, you notice that the cache's rate of use is close to 100%, you may want to consider increasing the size that has been allocated to it. The /4DSTATS and /4DHTMLSTATS URLs allow you to also obtain information about the cache's state. Please refer to paragraph "Web Server Management URLs", page 174.

2 Click on OK.

Once the cache has been enabled, the 4D Web server looks for the page requested by the browser first in the cache. If it finds the page, it sends it immediately. If not, 4D loads the page from disk and places it in the cache.
When the cache is full and additional space is required, 4D "unloads" the oldest pages first, among the least demanded ones.

Clearing the Cache

At any moment, you can clear the cache of the pages and images that it contains (if, for example, you have modified a static page and you want to reload it in the cache).
To do so, you just have to click on the **Clear Cache** button in the "Web Server II" page of Database Properties. The cache is then immediately cleared.

Defining the IP Address for the HTTP Queries

In the "Web Server I" page of Database Properties, it is now possible to define the IP address on which the Web server must receive HTTP queries.

By default, the server responds to all IP addresses (**Any IP Address** option) as in the previous versions.

When you select the **Specify IP Address** option, the "IP Address" area is enabled so that you can enter a specific address, like "194.166.100.101". In this case, the server only responds to queries sent to this address.

This feature is for 4D Web Servers located on machines with multiple TCP/IP addresses. It is, for example, frequently the case of most Internet host providers.

Information about the Web Site

4D version 6.5 allows you to obtain information about the functioning of your 4D Web site.

- You can control the site by using particular URLs (/4DSTATS, /4DHTMLSTATS and /4DCACHECLEAR).
- You can generate a log of all the queries.

Web Server Management URLs

4D Web Server version 6.5 accepts three particular URLs: /4DSTATS, /4DHTMLSTATS and /4DCACHECLEAR.

Note These URLs are only available to the Designer and Administrator of the database. If the database's 4D password system has not been activated, these URLs are available to all the users.

/4DSTATS

The /4DSTATS URL returns the following information in pure text form:

- the number of "hits" (low-level connections),
- the number of contexts created,
- the number of contexts that could not be created (once the maximum number of licenses has been attained),
- the number of available Web licenses,
- an estimation of the number of additional licenses required[1],
- the number of password errors,
- the number of pages stored in the cache,
- the percentage of cache used,
- the list of pages and GIF files stored in the cache of the static pages[2].

This information can allow you to check the functioning of your server and eventually adapt the corresponding parameters.

Note The new command WEB CACHE STATISTICS allows you to also obtain information about how the cache is being used for static pages (please refer to the routine WEB CACHE STATISTICS, page 135).

1. Theoretical minimum estimate (from the statistics)
2. For more information about the cache of static pages and pictures, please refer to paragraph "Cache for Static Pages", page 172.

/4DHTMLSTATS The /4DHTMLSTATS URL returns, also in pure text form, the same information as the /4DSTATS URL. The difference is that in the last field (contained in the cache), only the list of HTML pages — without the .GIF files — present in the cache is returned.

/4DCACHECLEAR The /4DCACHECLEAR URL immediately clears the cache of the static pages and images. It allows you to therefore "force" the update of the pages that have been modified.

Connection Log File

4D 6.5 allows you to obtain a log of queries. The log is presented in the form of a file named "weblog.txt" automatically placed at the same level as the structure file of the database. This file is in CLF (*Common Log File*) format or NCSA format, recognized by most Web site analysis tools.

Each line of the file represents a query, such as:

```
host rfc931 user [DD/MMM/YYYY:HH:MM:SS] "query" state length
```

Each field is separated by a space and each line ends by the CR/LF sequence (character 13, character 10).

- host: IP address of the client (ex. 192.100.100.10)
- rfc931: information not generated by 4D, it's always - (a minus sign)
- user: user name as it is authenticated, or else it is - (a minus sign). If the user name contains spaces, they will be replaced by _ (an underscore).
- DD: day, MMM: a 3-letter abbreviation for the month name (Jan, Feb,...), YYYY: year, HH: hour, MM: minutes, SS: seconds
  The date and time are local to the server.
- query: query sent by the client (ex. GET /index.htm HTTP/1.0)
- state: response given by the server.
- length: size of the data returned (except the HTTP header) or 0.

Note For performance reasons, the operations are saved in a memory buffer in packets of 1Kb before being written to disk. The operations are also written to disk if no query has been sent every 5 seconds.

The possible values of state are as follows:
200: OK
204: No contents
302: Redirection
400: Incorrect query
401: Authentication required
404: Not found
500: Internal error

▼ Examples of lines generated by the query log:

■ 192.100.100.10 - - [25/Jan/1998:12:54:06] "GET /index.htm" 200 6524
The Web client whose address was 192.100.100.10 was not authenticated. It asked for page *index.htm*, which was sent (it contains 6,524 bytes).

■ 192.100.101.25 - - [25/Jan/1998:12:54:09] "GET /123456.htm" 404 125
The Web client whose address was 192.100.101.25 was not authenticated. It asked for page *123456.htm*, which was not found (4D sent a message of 125 bytes).

■ 192.100.101.31 - - [25/Jan/1998:12:54:10] "GET /secret.htm" 401 0
The Web client whose address was 192.100.101.31 was not authenticated. It asked for page *secret.htm*, the server requested Authentication.

■ 192.100.101.31 - ZZZZ [25/Jan/1998:12:54:11] "GET /secret.htm" 401 0
The Web client whose address was 192.100.101.31 was authenticated as ZZZZ. It asked for page *secret.htm*, the user name was unknown.

■ 192.100.101.31 - ACI [25/Jan/1998:12:54:12] "GET /secret.htm" 200 2543
The Web client whose address was 192.100.101.31 was authenticated as ACI. It asked for page *secret.htm*, which was sent (it contains 2,543 bytes).

WARNING : *The log file can be imported into a spreadsheet or directly into 4D. However, you must imperatively stop the Web server before importing it.*

By default, the log file of queries is not generated.

► To request the generation of a log file of all the Web queries:

1 In the "Web Server II" page of Database Properties, select the "Log Request" option.

2 Click the OK button.

HTML Support

The “automatic” functions that convert database objects into HTML and, in general, that support HTML have been optimized and improved in version 6.5 of the 4D Web server.
This section explains these modifications.

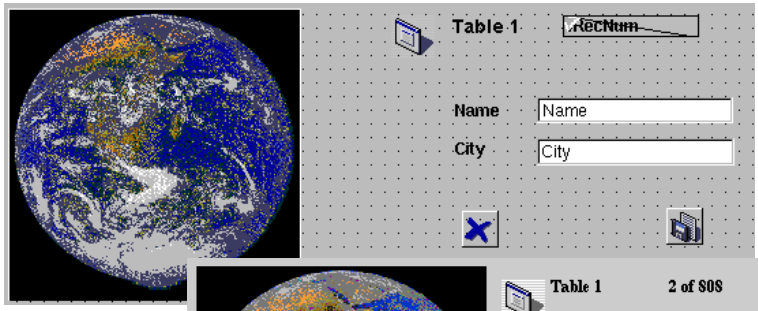
Static Pages

- 4D is now completely compatible with HTML 3.2.
- 4D now supports HTML frames in static pages.

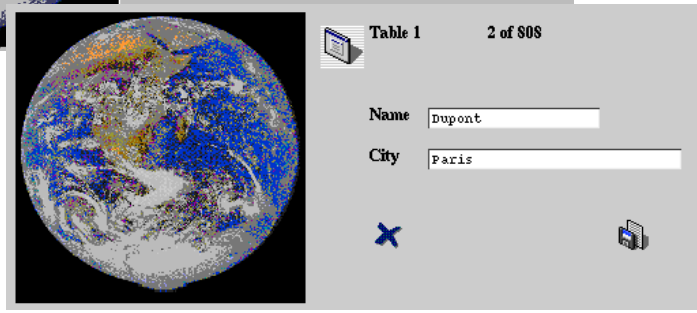
Dynamic Pages

- The **placing of the form’s objects** sent to the browser has been improved. Objects now have the same position on the browser as they do in the 4D form.

Form editor



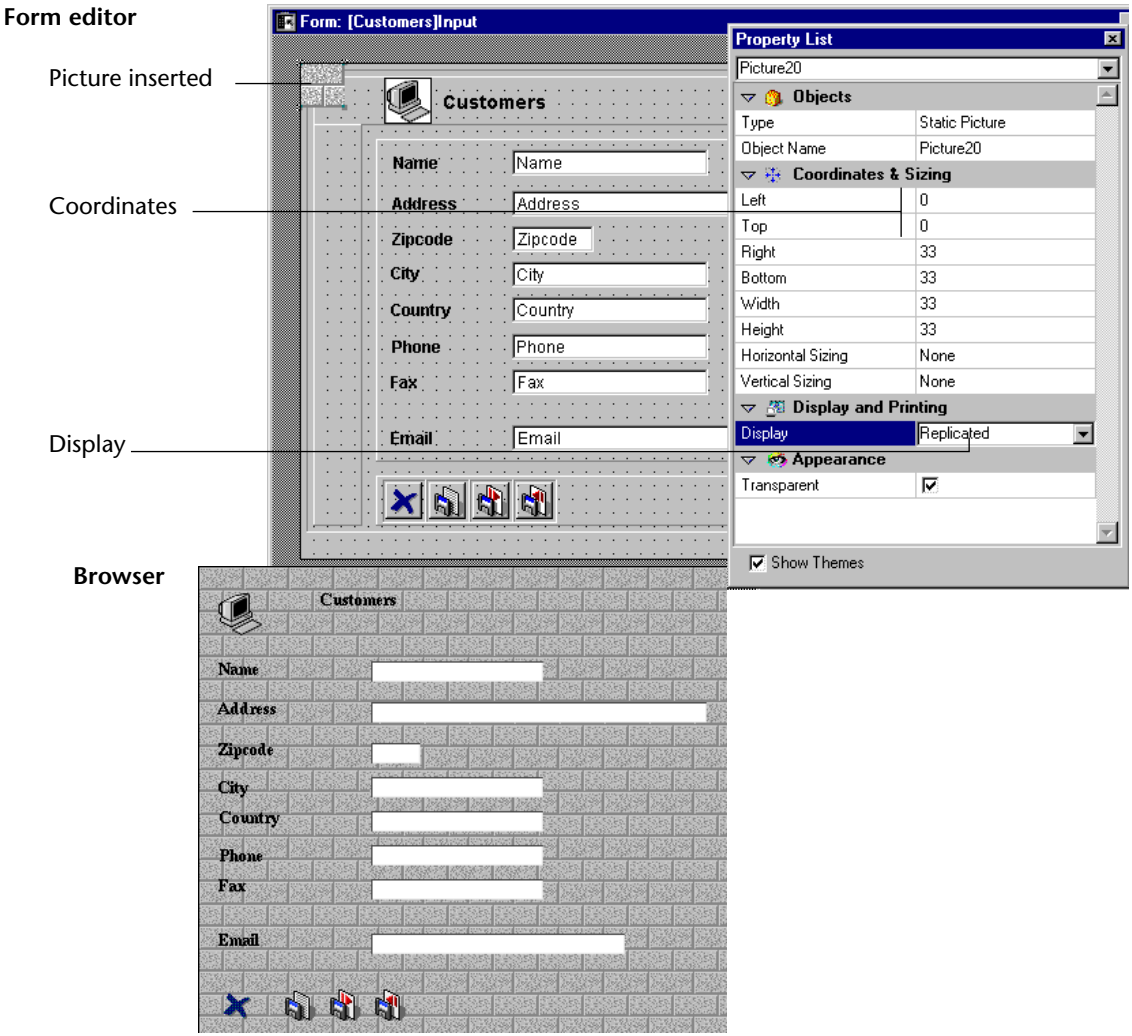
Browser



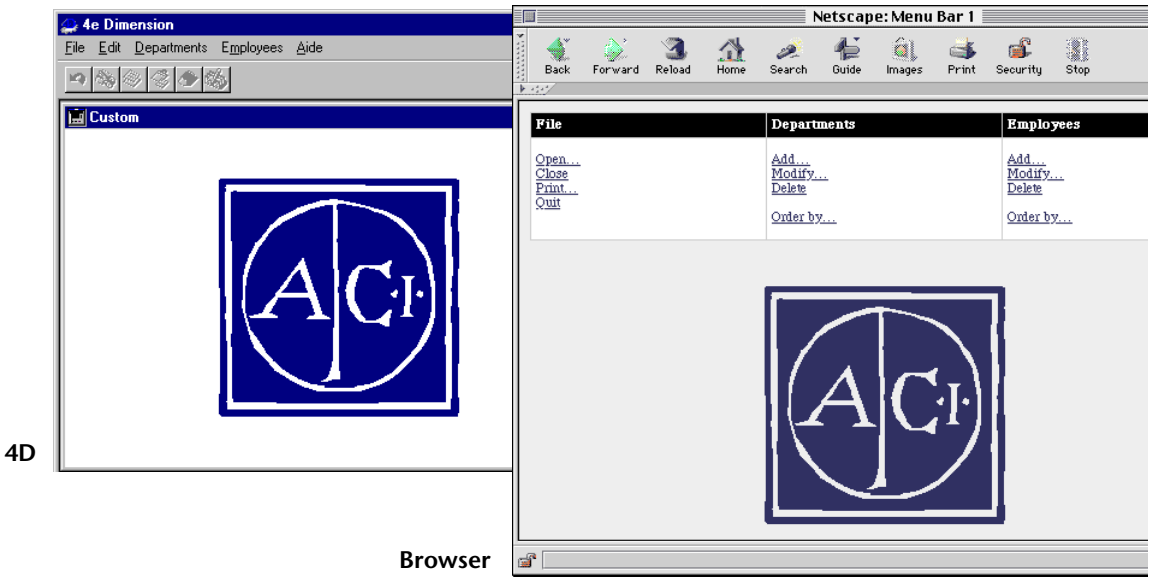
- The **plug-in areas** are now publishable on the Web, by first being converted into HTML, Image or Image Map. This last solution allows you to manage mouse clicks inside the plug-in area (for example, the integrated plug-in 4D Chart is published in an Image Map and the ACI_Pack *External Clock* area is published as an Image).

Note The way in which a plug-in area, which is on a 4D form, is published on the Web depends on the plug-in editor’s specifications.

- The **objects on page 0** (zero) of a form are now visible on the browser.
- When a **4D picture variable** is referenced in an HTML page, the picture is then correctly displayed on the browser.
- **Background pictures**: a picture replicated ("Replicated" display) inserted in the (0,0,x,x) coordinates in 4D's form editor is now sent as a background picture on the browser.

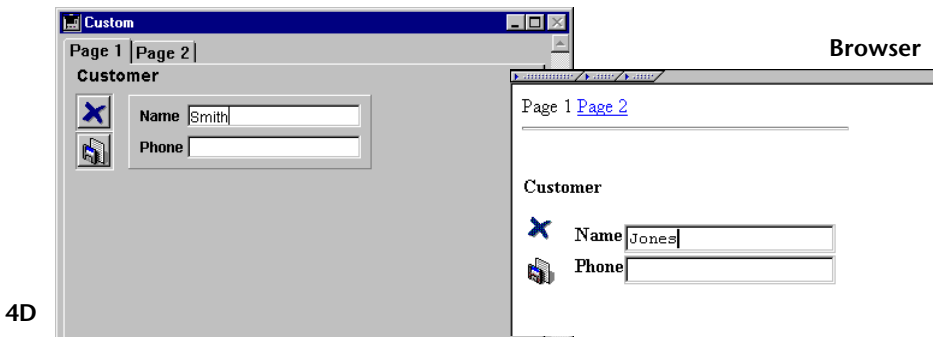


When a 4D menu bar is displayed, the picture associated to the menu bar is placed below the menus on the browser:



Please note that dark pictures should be avoided.

- The **tips** associated to buttons displayed as pictures in the form editor now appear on the browser — if the browser allows these tips to be displayed.
- The **tabs controls** (of type array or created by using the values defined in the Object properties) are converted into URL lists.



If the array elements are empty strings, 4D displays 1, 2, 3... on the browser.

Note In some cases, the Web conversion of 4D forms created in version 6.0.x may be incorrect with 4D 6.5, especially if a form contains a reference to a page, such as {mypage.htm}.
In this case, it is possible to activate the 6.0.x-based conversion mode, using the SET DATABASE PARAMETER command (cf. page 125).

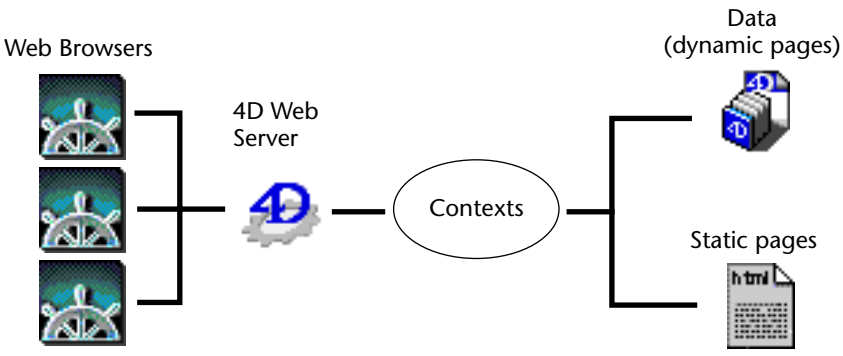
Non Contextual Mode

The 4D version 6.5 Web server can now be used in *non contextual mode*. In this mode, 4D becomes a "classical" HTTP server.

Contextual Mode and Non Contextual Mode

In previous versions of 4D, the program's Web features were based on the notion of *contexts*: a Web browser that connects to a database creates a context in which it places its own current selection, its variables, etc. In a way, each browser is considered a 4D Client (cf. the figure below).
This system allows the 4D Web Server to perfectly control the actions of the browsers and to guarantee the integrity of the data. It turns out to be less appropriate because 4D is used as a "classical" HTTP server especially when 4D sends static HTML pages. In this case, the context generated by the browser remains active until the specified timeout has been attained. If, for example, the browser quits the site in the meantime, the context as well as the Web license are "wasted". Moreover, certain standard browser functions (**Reload**, **Previous page**, etc.) are not usable.

How the 4D Web Server 6.0.x functions



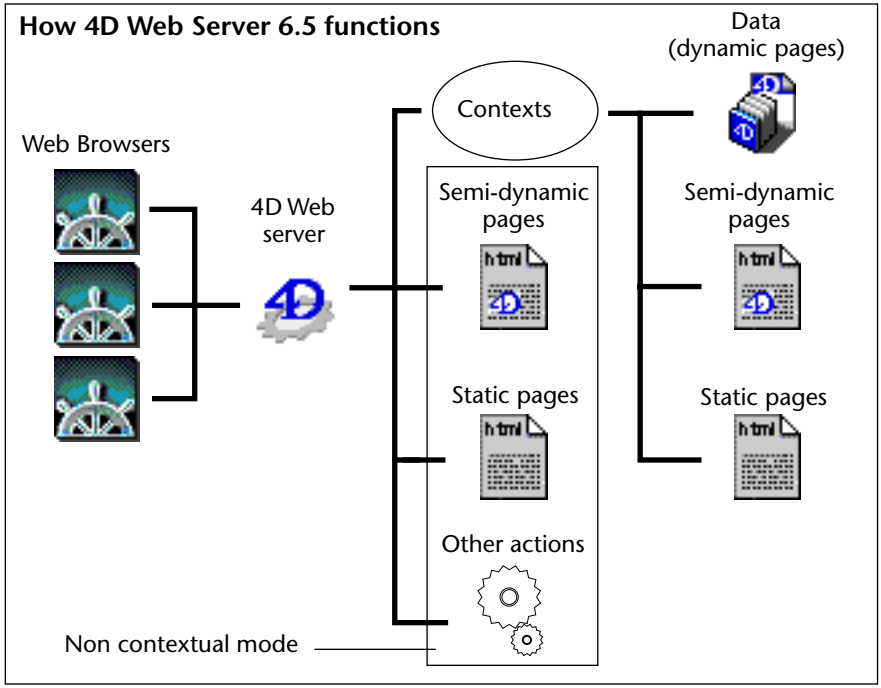
To use the 4D Web Server as a classical HTTP server when sending static pages, 4D version 6.5 introduces the *non contextual mode*. In this

mode, the 4D Web Server becomes a standard HTTP server: the static Web pages are sent without the necessity to maintain a context or to use a Web license.[1]

These static pages can then benefit from an adjustable cache (cf. paragraph “Cache for Static Pages”, page 172). Also in this case, you can obtain statistics about the consultation of these pages by using the routine WEB CACHE STATISTICS, page 135.

Of course, contexts are always used to control the navigation of the browsers on the records.
Actually, you can use the 4D Web Server in any mode you wish: contextual mode, non contextual mode, or switching between the two modes on-the-fly according to your needs (For more information, please refer to the paragraph “Switching from one Mode to Another”, page 182).

How 4D Web server 6.5 functions is summarized in the following figure:



1. You can now find out if a Web process consumes a Web license or not by using the PROCESS PROPERTIES command (cf. paragraph “Number of Licenses in Use”, page 183).

Note Semi-dynamic pages allow a new type of communication between the Web server and 4D. For more information, please refer to the paragraph “Semi-dynamic Pages”, page 185.

Switching from one Mode to Another

By default, version 6.5 of the 4D Web server functions as in previous versions as far as managing contexts is concerned: a context is generated at each Web connection.
You have to explicitly indicate that you want to switch to non contextual mode.

Going from Contextual Mode to Non Contextual Mode

While using your database, you can, at any moment, switch from contextual mode to non contextual mode, or vice-versa.

Note By default, the 4D Web Server starts in contextual mode. You can however define to startup in non contextual mode. For more information, please refer to the paragraph “Defining the Non Contextual Mode at Startup”, page 184.

For the 4D Web Server to switch to non contextual mode, three solutions are available:

- Execute the new command SEND HTML BLOB by passing True to the optional parameter (*noContext*). For example:
SEND HTML BLOB (MyBlob; “.HTM”; **True**)
4D sends the BLOB in non contextual mode. The process that manages the context is immediately stopped and the license is freed.

Note Not passing the last parameter is the same as passing False. If the non contextual mode is already active, the parameter is ignored. You can know the current mode by using the Web Context function (cf. page 134).

- Execute the new command SEND HTTP REDIRECT.
This command allows you to redirect the query to the URL passed as a parameter. Called from the contextual mode, it closes the Web process and frees the license.
- Call a URL starting with /4DACTION (cf. paragraph “4DACTION Tag”, page 186).

  The non contextual mode remains to be used while no URL calling the contextual mode is called.

Switching from Non Contextual Mode to Contextual Mode

Generally, 4D's "classical" contextual mode is used as soon as the database is solicited, which means as soon as the Web server must generate and send dynamic pages.

Calling the 4D database from the non contextual mode is done, as in 4D 6.0.x, through a particular URL: "/4DMETHOD/MyMethod". You just have to place this URL in a static page (for example, in a button).

When this URL is enabled, the 4D Web server creates a new context and executes the following operations:

- the On Web Authentication database method is executed (if it exists),
- the On Web Connection database method is executed (if it exists)[1],
- the method is executed in the newly created context.

The Web server functions as it did in previous versions of 4D. The contextual mode remains used as long as no command enabling the non contextual mode is called.

Note In 4D 6.5, the On Web connection database method now has an "entry point" role in 4D (from the non contextual mode). You can put different instructions depending on the origin of the call in the database method (cf. paragraph "Changes in the On Web Connection Database Method", page 191).

Number of Licenses in Use

Depending on the actions performed, certain Web processes use Web licenses, while others do not.
The licenses are managed internally by 4D. You can however know the number of licenses being used by using the PROCESS PROPERTIES command for any Web process — either in contextual or non contextual mode. This command indicates in the new parameter origin if a Web license is being used (-11, Web Process with license) or not (-3, Web Process with no license). Refer to the paragraph "PROCESS PROPERTIES (Processes)", page 145.

1. In this particular case, $1 is equal to "/4DMETHOD/MyMethod" instead of "/" (slash).

Defining the Non Contextual Mode at Startup

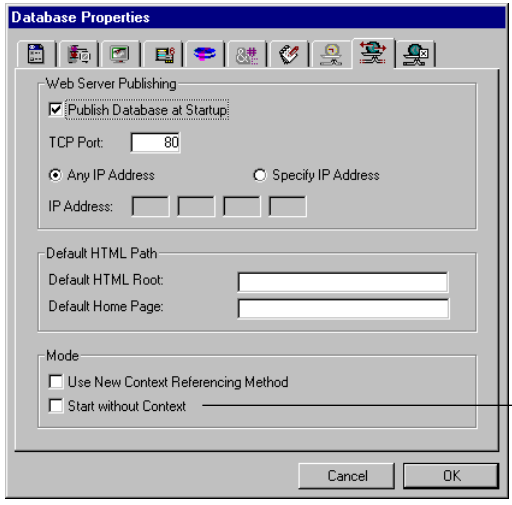
You can start the 4D Web server so that it is automatically in non contextual mode. This means that when a user connects to the database, **no context is generated**.

▶ To define the non contextual mode at startup:

1 **In the Database Properties dialog box, click on the "Web server II" tab.**

2 **Select the "Start without Context" option.**

By default, this option is not selected, the 4D Web server starts in contextual mode:



3 **Accept the dialog box.**

The 4D Web Server sends the home page in non contextual mode.

- If you have specified a custom home page in the "Default Home Page" area of Database Properties (cf.paragraph "Defining a Default Home Page", page 167), the page is sent[1].
- If you have not specified a default home page or if the specified home page does not exist, the On Web Connection database method is called (in non contextual mode). You can also manage the connection in a personalized manner. For example, you can use the new command SET HOME PAGE to select a home page for each user that connects to the database.

1. If the URL ends with "/", the name of the home page is added and 4D will try to send it. For example, if the URL is "/Folder/" and the default home page is "MyPage.HTM", 4D will look for "/Folder/MyPage.HTM". If it is not found, the On Web Connection database method is called.

Semi-dynamic Pages

4D version 6.5 introduces a new type of Web page, which are generated in a **semi-dynamic** manner. Semi-dynamic pages are Web pages whose contents come totally or partially from an action performed by 4D.

Semi-dynamic pages can be usable in both contextual and non contextual mode. However, they are mostly used in non contextual mode. Actually, in this mode, 4D does not maintain any variables, selections, etc. related to the browser's navigation. They are maintained in an entirely free manner without 4D being informed. The purpose of semi-dynamic pages is to allow you to take advantage of 4D in your Web pages while keeping the advantages related to the non contextual mode.

You can generate semi-dynamic pages in the following three ways:

- By sending the "/4DCGI/<action>" URL to the 4D Web server.
- By sending an HTML page with the extension ".shtm" or "shtml".
- By using the "/4DACTION/" tag (this specific tag is described in paragraph "4DACTION Tag", page 186).

URL /4DCGI/<action>

When the 4D Web server receives the /4DCGI/<action> URL, it calls the On Web Connection database method by sending the URL "as is " to $1.

The "4DCGI/" URL does not correspond to any file. Its role is to call 4D. The "<action>" parameter can contain any type of information.

This URL allows you to perform any type of action. You just need to test the value of $1 in the On Web connection database method or in one of its submethods and have 4D perform the appropriate action. For example, you can build completely custom static HTML pages to add, search, or sort records or to generate GIF images on-the-fly. Examples of how to use this URL are in the descriptions of the PICT TO GIF and SEND HTTP REDIRECT commands.

When issuing an action, a "response" must be returned, by using commands that send data (SEND HTML FILE, SEND HTML BLOB, etc.).

WARNING : *Please be sure to execute the shortest possible actions so as not to hold up the browser.*

".shtm" File

When the 4D Web Server sends an HTML page whose extension is ".shtm" (or ".shtml"), it analyzes the references to 4D variables and the 4DACTION comments (see below) that may be present in the page before sending it to the browser.

This feature allows you to perform processings in 4D for displaying an HTML page while ensuring that the dynamic references in the static pages are updated. Make sure however that you execute the shortest possible actions so as not to hold up the browser.

Note In non contextual mode, it is not possible to make a reference to 4D picture variables.

▼ Please refer to the example of the routine WEB CACHE STATISTICS, page 135.

4DACTION Tag

The 4DACTION tag allows you to execute 4D methods when sending static HTML pages. You can use this tag in three ways:

- As an HTML comment (<!--4DACTION ...-->),
- As a URL (<A HREF= "/4DACTION/...">),
- To post a form.

4DACTION as an HTML Comment

The presence of the /4DACTION/MyMethod/MyParam tag in a static page as an HTML comment forces the execution of the *MyMethod* method with the MyParam parameter as a string in $1. When loading the home page, 4D calls the On Web Authentication database method (if it exists). If it returns True, 4D executes the method.
The method returns text in $0. If the string starts with the ASCII code character 1, it is considered as HTML (the same principle is true for the variables).

WARNING : *For this mechanism to function, the "Use 4DVAR Comments instead of Brackets" option must be selected in Database Properties (cf. paragraph "New Mode to Insert 4D Variables in Static Pages", page 170).*

The analysis of the contents of the page is done when either SEND HTML FILE (.htm, .html, .shtm, .shtml) or SEND HTML BLOB (blob of type text/html) is called.

Remember that in non contextual mode, the analysis is done when a URL points to a file that has either the ".shtm" or ".shtml" extension (for example http://www.server.com/dir/page.shtm).

Note In contextual mode, the method is executed in the context.

- For example, let's say that you insert the following comment "Today is <!--4DACTION/MYMETH/MYPARAM-->" into a static page. When loading the page, 4D calls the On Web Authentication database method (if it exists), calls the *MYMETH* method and passes the string "/MYPARAM" as the parameter $1.
  The method returns text in $0 (for example "10/28/98"), the expression "Today is <!--4DACTION/MYMETH/MYPARAM-->" therefore becomes "Today is 10/28/98".
  The *MYMETH* method is as follows:

  C_TEXT($0) `This parameter must always be declared
  C_TEXT($1) `This parameter must always be declared
  $0:=**String(Current date)**

Note A method called by 4DACTION must not call interface elements (DIALOG, ALERT...).

As 4D executes methods in their order of appearance, it is absolutely possible to call a method that sets the value of many variables that are referenced further in the document, whichever mode you are using.

Note You can insert as many <!--4DACTION...--> comments as you want in a static page.

4DACTION as a URL

When 4D receives a /4DACTION/MYMETH/PARAM query, the On Web Authentication method database (if it exists) is called. If it returns True, the MYMETH method is executed with the /PARAM string as the parameter (in $1). The syntax of the URL must be in the following form:
<A HREF="/4DACTION/MYMETH/PARAMS">Do Something</A>

Note This feature is identical to /4DMETHOD except that it is applied to the non contextual mode (cf. 4D's *Language* manual).

In contextual mode, when 4D receives a query of type /4DACTION, the context is aborted and the method is executed outside of the context. It is precisely the opposite of calling /4DMETHOD in non contextual mode (cf. paragraph "Switching from Non Contextual Mode to Contextual Mode", page 183).

Note A method called by 4DACTION must not call interface elements (DIALOG, ALERT...).

▼ You insert the following instructions in a static HTML page:
<IMG SRC="/4DACTION/PICTFROMLIB/1000">
The *PICTFROMLIB* method is as follows:

```
C_TEXT($1) `This parameter must always be declared
C_PICTURE($PictVar)
C_BLOB($BlobVar)
C_LONGINT($Number)
  `We retrieve the picture's number in the string $1
$Number:=Num(Substring($1;2;99))
GET PICTURE FROM LIBRARY($Number;$PictVar)
PICT TO GIF($PictVar;$BlobVar)
SEND HTML BLOB ($PictVar;"Pict/gif")
```

4DACTION to post forms The non contextual mode offers an additional possibility when you want to use "posted" forms, which are static HTML pages that send data to the Web server. The POST[1] type must be associated to them and the form's action must imperatively start with /4DACTION/MethodName.

In this case, when the Web server receives a posted form, it calls the *COMPILER_WEB* project method (if it exists, see below), then the On Web Authentication database method (if it exists). If it returns True, the *MethodName* method is executed. 4D analyzes the HTML fields present in the form, retrieves their values and automatically fills the 4D variables with their contents. The field in the form and the 4D variable must have the same name.

Note A method called by 4DACTION must not call interface elements (DIALOG, ALERT...)

1. 4D accepts the posted forms of type GET. However in this case, the developer must manually process the HTML contents of the generated URL. Moreover, the COMPILER_WEB method is not called (see below). Note that 4D also accepts multipart/form-data forms.

The HTML syntax to apply in the form is of the following type:

- to define the action in a form:
  <FORM ACTION="/4DACTION/MethodName" METHOD=POST>
- to define a field in a form:
  <INPUT TYPE=Field type NAME=Field name VALUE="Default value">

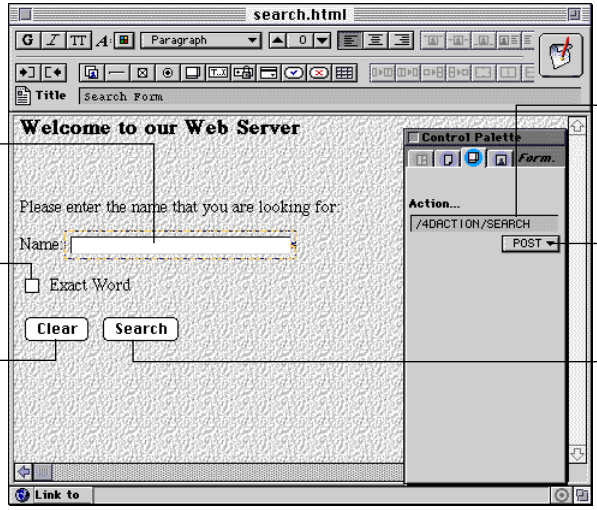
For each field in the form, 4D sets the value of the field to the value of the variable with the same name.
For the form options (for example, check boxes), 4D sets the associated variable to 1 if it is selected, otherwise 0.
For numerical data entries, 4D converts the value of the field from Alpha–>Real.

Note If a field in the form is named OK (for example the Submit button), the *OK* system variable is set to 1 if the value of the field is not empty, otherwise it is set to 0.

COMPILER_WEB Method: When the 4D Web server receives a posted form, it calls the project method called *COMPILER_WEB* (if it exists). This method must contain all the typing and/or variable initialization directives, which are the variables whose names are the same as the field names in the form. It will be used by 4D Compiler when compiling the database.
The *COMPILER_WEB* method is common to all the forms.

▼ In a 4D Web database started and used in "non contextual" mode, we hope that the browsers can search records by using a static HTML page. This page is called "search.htm". The database contains other static pages that allow you to, for example, display the search result ("results.htm"). The POST type has been associated to the page, as well as the /4DACTION/SEARCH action.

Here is the page as it appears in an HTML editor, in our case Adobe® PageMill™:



Here is the HTML code that corresponds to this page:

```
<FORM ACTION="/4DACTION/PROCESSFORM" METHOD=POST>
<INPUT TYPE=TEXT NAME=VNAME VALUE=""><BR>
<!-- Usually we put the name of the button in VALUE, for interpretation reasons, you must put a number in VALUE-->
<INPUT TYPE=CHECKBOX NAME=EXACT VALUE="1">Whole word<BR>
<!-- OK is a particular case-->
<INPUT TYPE=SUBMIT NAME=OK VALUE="Search">
</FORM>
```

During data entry, type "ABCD" in the data entry area, check the option and validate it by clicking the **Search** button.

4D then calls the *COMPILER_WEB* project method, which is as follows:

```
C_TEXT(VNAME)
VNAME:=""
C_LONGINT(vEXACT)
vEXACT:=0
OK:=0     `particular case
```

In the example, *VNAME* contains the string "ABCD", *vEXACT* is equal to 1 and *OK* is equal to 1 (because the button's name is **OK**).
4D calls the On Web Authentication database method (if it exists), then the *PROCESSFORM* project method is called, which is as follows:

```
If (OK=1)
  If (vEXACT=0) `If the option has not been selected
    vNAME:=VNAME+"@"
  End if
  QUERY([Jockeys];[Jockeys]Name=vNAME)
  vLIST:=Char(1)   `Return the list in HTML
  FIRST RECORD([Jockeys])
  While (Not(End selection([Jockeys])))
    vLIST:=vLIST+[Jockeys]Name+" "+[Jockeys]Tel+"<BR>"
    NEXT RECORD([Jockeys])
  End while
  SEND HTML FILE("results.htm")   `Send the list to the results.htm form
    `which contains a reference to the variable vLIST (for example
    `<!--4DVAR vLIST-->)
  ...
End if
```

Changes in the On Web Connection Database Method

The On Web Connection database method has been modified. It now receives 6 parameters and its role in the 4D Web server has been reinforced.

New Parameters

The On Web Connection database method now has four new parameters ($3, $4, $5, $6) besides the existing ones, $1 and $2. These parameters are identical to those received by the On Web Authentication database method (cf. paragraph "On Web Authentication Database Method", page 157).

These parameters, which are of type Text, provide you with additional information about the connection:

Parameters	Type	Description
$1	Text	URL
$2	Text	HTTP header
$3	Text	Browser's IP Address
$4	Text	Server's IP Address
$5	Text	User name
$6	Text	Password

For more information about the $1 and $2 parameters, please refer to the description of the On Web Connection database method, in 4D's documentation.

For more information about parameters $3 to $6, please refer to the paragraph "On Web Authentication Database Method", page 157.

Note If the user name sent to the browser exists in 4D, parameter $6 is not returned for confidentiality reasons (an empty string is returned).

On Web Connection and the Non Contextual Mode

The On Web Connection database method, as in previous versions of 4D, is called when a browser connects to the database. But also, it plays a role as the entry point into 4D from the Web server's new "non contextual" mode.
Actually, switching from non contextual mode to contextual mode is done through the On Web Connection database method (for more information, please refer to the paragraph "Non Contextual Mode", page 180).

Note In non contextual mode, calling a 4D command that displays an interface element (ALERT, DIALOG...) ends the processing.

The On Web connection database method is therefore called in the following cases:

- when connecting a browser to the 4D Web server — as with 4D 6.0.x. The database method is called with the /<action>... URL.
- when switching from non contextual mode to contextual mode. The database method is called with the /4DMETHOD/MethodName URL.
- when 4D is called by a semi-dynamic page, through the /4DCGI/<action> URL. The database method is called with the URL (cf. paragraph "Semi-dynamic Pages", page 185).
- when a Web page is called in non contextual mode and a URL of type <path>/<file> is not found. The database method is called with the URL.
- when a Web page is called in non contextual mode with a URL of type <file>/ and no home page has been defined by default. The database method is called with the URL.

To know the origin of where the On Web connection database method was called so that you can adapt the processing that needs to be executed, the new function Web Context has been added to 4D 6.5 (please refer to the description of the routine Web Context, page 134).

Consequently, we suggest that you structure the On Web Connection database method in 4D 6.5 in the following manner:

```
  `On Web connection database method
If (Web Context) `If in contextual mode
  WithContext ($1;$2;$3;$4;$5;$6)
    `The WithContext contains everything that was in the
    `On Web connection database method in 4D 6.0.x
Else
  NoContext ($1;$2;$3;$4;$5;$6)
    `The NoContext method executes the non contextual
    `processing of queries (generally short)
End if
```

6 Optimizations

Besides the new features added to 4D version 6.5 described in the other chapters, this new version also includes numerous “internal” optimizations, which do not modify the general functioning of 4^{e} Dimension or how your methods are written, but improve the performance of your 4D databases.

New Index Mode

4D allows you to now choose between two index modes: the “traditional” mode, which is the mode used in previous versions of 4D, and the new “fast” mode, which in most cases allows for an important increase in speed.

Selecting an Index Mode

Selecting an index mode is only possible if the number of records is greater than 1,000. The mode can only be changed in Design mode.

► To choose an index mode:

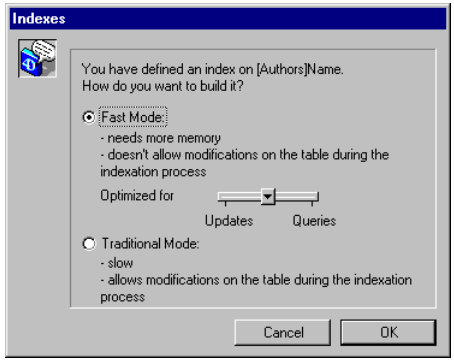
1 **Go to the Structure editor.**

2 **Select the field to index or reindex and select Field Properties... in the Structure menu.**
OR
Double-click on the field to index or reindex.

3 **In the Field Properties palette, select the Indexed option and click on the Apply button.**

Note You can also click the **right mouse button** (Windows) or **Ctrl+click** (MacOS) the field name and choose **Indexed** or **Reindex** in the contextual menu.

The following dialog box appears:



4 **Click the radio button that corresponds to the mode you wish to use.**

To know which mode to choose, please refer to the section below.

5 **If you have chosen the fast mode, set the index's optimization rate by using the thermometer.**

This rate must be set according to how the index will be used and must be between the two following limits:

- **Updates**: The index is going to be constantly modified. This is the case when data is being constantly added to the table.
- **Queries**: The index will no longer be modified and will be mainly used for searches, sorts, etc.

Please note that this feature concerns the index's optimization rate and not its definition. Even if you have placed the thermometer to the extreme right for Queries, the index can still be modified. However, its performance will not be optimal.

Which Mode to Choose?

Each mode has its advantages and inconveniences:

Comparative table of the two index modes

	Advantages	*Inconveniences*
Fast Mode	- Fast construction[1] of the index - Optimization possibility according to usage	- Needs more memory - "Locks" any modifications to the table during the indexing process
Traditional Mode	- Needs less memory - Allows you to modify the table during the indexing process	- Slow construction of the index - No optimization possible

1. The increase in speed depends mainly on the number of records to index, the fragmentation of the data and the available memory (RAM).

In sum, the fast mode (which is selected by default) is more efficient and can be optimized. It should be used in most cases. To obtain a maximum increase in speed, the indexing must be done as often as possible in memory (you can verify, in the Evaluation page of the Runtime Explorer, if the memory cache is sufficient to contain the index). However, if you don't have enough memory or if you don't want a table in your database to be locked (in write mode) for a certain amount of time, you can choose the traditional mode.

Speed Acceleration of Database Functions

Many of 4D's database functions have been optimized, which means that their execution speed has been increased tremendously. Below are the modified functions:

Searches and Sequential Sorts

Searches and sequential sorts have been optimized. 4D tries to read more records at a time from disk. This mode is particularly interesting because the size of the cache is low.
Moreover, the performance of the functions that call these mechanisms has also been improved:

- exporting commands
- indexing commands
- the SELECTION TO ARRAY command

Note You can disable the optimized mode by using the routine SET DATABASE PARAMETER, page 125.

Indexed Searches

Indexed searches have been optimized when data is made up of a large number of repeated values.

Delay in Displaying the Progression Thermometer

When executing certain instructions, a progression thermometer is displayed. From now on, for the functions described below, the displaying of the progression thermometer is delayed so that it only appears if the processing time is long enough (>1 second). In previous versions, when the execution time was very short, the thermometer was taking more time to display than a query did to be completed.

Note This feature already applies to queries and indexed sorts.

The following commands are affected: DISTINCT VALUES, RELATE ONE SELECTION, RELATE MANY SELECTION, ORDER BY, QUERY, as well as the function that deletes an index.
The functions **Apply to selection**, **Query by formula** and **Quick Reports** also benefit from this optimization.

Compacting Index Pages

Index pages can, in databases containing a lot of indexes and records, consume a lot of space in 4D's memory cache.
Now, when the cache is full and 4D needs additional space, the data in the cache is not directly unloaded anymore. Before executing this operation, the program is going to check if it can gain space by compacting the index pages. This alternative allows you to avoid reloading the data later.

Note You can disable this feature by using the routine SET DATABASE PARAMETER, page 125.

Index

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W