List boxes

By Jean-Yves Fock-Hoon, Quality Assurance Manager, 4D Inc.

Technical Note 06-25

Overview

The purpose of this two-part technical note is to introduce list boxes and their use through the Design mode and the language. Part I of this technical note focuses on the properties of list boxes.

A new object

In former versions, the developer could create several scrollable areas and group them in order to display data in multiple columns while being able to select one entire row at a time. Grouped scrollable areas were very basic and had significant limitations. In 4D 2004, a new object has been introduced: the list box. This is an object that allows you to display arrays and lets you have more control over value display and entry.

Creating a list box from the Form editor

As any form object, a list box is created in a form, in the Design environment. From the toolbar, you can select a List Box object and draw that object in the form.
The default object will have default settings and one column.

Selecting a list box in the form editor can be confusing: You can click once on a list box and it will be selected. If you click again in the list box, you can either select a column or a header for that list box. Therefore, it is advised to make sure the right object or aspect is selected when assigning properties to a list box. As example, you can select the **Preview** menu item from the **Demonstration** menu bar. We can see a dialog with our list box and 4 columns in the [Contacts]Preview form. These 4 columns and all their attributes have been defined in the list box object.

Let’s open that form in Design mode, select that list box object and have a look to the list box’s properties.

Property List for the List box object:
- Theme Object:
Name and object name:
We can see that our list box has a name. Our list box is in fact a Boolean array.

- Theme List Box:
Number of columns:

We can see the current number of columns, 8. This is the number of columns that 4D displays in that object when used for the first time. When increasing that number, 4D will insert new columns with default names in that object. Arrays defined in that list box would have to be previously defined. All arrays should have the same number of elements. If not, the list box will use the size of the smallest array. If a list box is displayed as empty, it is most likely that there is an array that is empty. You can use 4D commands to insert or delete columns in a list box.

Number of Static columns:
This is the number of columns that are static (i.e. that cannot be moved). This applies to the first columns of the list box. In this example, we can see 1. This means that the first column, aNum, cannot be moved. If the value was 2, the 2 first columns, aNum and aString, would not be able to move. They would have to keep their current position. If you do not want to be able to move any column, enter the number of columns. This property cannot be defined through the language.

Show Column Header:
This check box allows you to hide or show column titles. This does not mean that those variables won't be used. They are just hidden. You can use the SET VISIBLE command on one header variable to hide or show the header line. In that example, the Hid Header button will hide and show all headers.

Multiple Selections:
Selecting this check box allows users to select multiple rows. You cannot define this property through the language. In the current example, you can select multiple lines.

Row Style Array, Font Color Array, Background Font Color Array:
Enter in this area the name of the array that will contain the styles and colors for each row. In Design mode, two grey tones have been defined as background and the style for each column is white and plain, except for the City column that is in bold and italic.

In the Custom menu environment, we can see that the arrays are displayed with different background and foreground colors as well as a different style. This is because 3 arrays have been defined for that property, a value for each row. The size of the arrays cannot be greater than the number of rows. If the sizes of these arrays are smaller than the number of rows, the remaining rows will use the settings defined in the list box object. In our example, we can see 8 rows and only the first 6 rows changed their colors and style. This is because the arrays have a size of 6 instead of 8. If we click on the last name to sort that column, our 2 grey lines are now in position 1 and 2. The Switch button is going to swap the values of our arrays. Their sizes won't change.

From the property list, we can see that the first 6 rows changed colors while the last 2 rows remained in Yellow. If you are using these arrays, make sure that you do not change their values later since it can also affect the list box (unless this is what you want to achieve). You cannot assign these arrays through the language. However, you can use 4D commands such as SET COLOR, FONT or FONT SIZE to define the style and background/foreground colors to each column instead of each row. If you are

using these arrays, this will overwrite the settings that you can define in the list box or in each column. If you want to disable temporarily those colors, resize the arrays to 0 elements. In that example, we are using 3 arrays. Click on all buttons in the Background Color area in order to change contents of these arrays or to reset the array to 0 elements and you will see the effects in the list box.

- Theme Gridlines:
In this area, you can define the gridlines. You can make vertical and/or horizontal gridlines visible or hidden. You can also change these properties through the language using the commands SHOW LISTBOX GRID and SET LISTBOX GRID COLOR.

- Theme Coordinates and Sizing:
These are the coordinates and size of the current list box object in your form. You can use the MOVE OBJECT command to move the list box in your form if needed.
You can also define the row height in that property list. This can also be redefined with the language. Use GET LISTBOX ROWS HEIGHT to retrieve the current height and SET LISTBOX ROWS HEIGHT to set the desired height. Click on the Change button in the Row Height area to see the height has been doubled. The list box only displays one line per row. Changing the height does not meant that text over 2 lines can be visible. You can double the height and text arrays will still display only the first line.

- Theme Resizing options:
These options allow you to define the behavior of the list box during the resizing of the window. As other objects, it can have a fixed size, move or grow horizontally or vertically. These options cannot be defined through the language.

- Theme Display:
Invisible by default:
This option allows you to make the list box invisible when displaying the form in your window. To make it visible or invisible through the language, you can use the SET VISIBLE command. In our example, you can click on the Hide columns button to hide or show all columns.

- Theme Appearance:
Platform:
The platform is available only in databases created with 4D 2004. You can choose between System and Printing. Neither will make a difference since you cannot print a list box. For converted databases, the platform will not be available; the list box will use a hard-coded interface, System. Like other objects, you cannot programmatically define this property.

Horizontal/Vertical scrollbars:
When displaying a list box, you may decide to display or hide a vertical or horizontal scrollbar. Check or uncheck the appropriate check box to show or hide both scrollbars. You can use the SET SCROLLBAR VISIBLE command to show or hide your scrollbars. In our example, you can click on the Hide Scrollbars button to hide or show all scrollbars.

- Theme Background and Borders:
Background Color/Alternate Background Color:
You can define a background color and an alternate background color for each row. Colors can be computed from formulas defined in SET COLOR or SET RGB COLORS

Border Line Style:
You can define a border, system, sunken, dotted or mo border at all if needed. This feature behaves like the same one for other 4D objects. You cannot define the border programmatically.

- Theme Text:
Those attributes are the same used for other objects such as fields and variables. You can define the font size, the font name or color for example. You can specify the use of a style sheet or just hard-code some properties to the list box object.
All chosen text attributes will be applied to all objects in the list box, columns and headers. All text attributes can be set from the language, such as FONT, FONT SIZE, FONT STYLE, SET COLOR or SET ALIGNMENT for example. In our example, you can click on the 2 buttons in the "Text Attr." area to see the changes on some fonts.

- Theme Action:
Method:
This is the object method for the list box.

Draggable:
The list box, or more likely, the current row is draggable, i.e. can be dragged onto another object that can handle the On Drop event. You cannot turn this property on or off by language.

Droppable:
The list box can be defined as droppable, i.e. can receive dropped events. You can drag any item from 4D and drop it on the current object. You can test the On Drag over and on Drop form events to handle the drag and drop in your list box.
You cannot turn this property on or off by language.

Movable rows:
Rows in your list box can be moved. You can change the disposition of your rows by drag and dropping rows between themselves. You cannot drag and drop a row that will become the last row of the list box. You can define that row as second last, then drag and drop the last row just before that second last row.

Sortable columns:
By clicking on the header, you can sort the current column. As soon as the column has been sorted, a small icon will display the current order. Click again on the column header to revert the sorting order.

- Theme Events:
This is the list of all events that the list box can fire and execute its object method.

If you click again inside the list box, you may now select either a column or a column header.

Property List for the column header:
If you select a column header, you can see the following in the Property list window:

- Theme Objects:
Object Name and variable name:
That would be the object name and the variable name of the header. Object names must be unique inside the form, among all objects in the form. Variable names must also be unique. You cannot use variables names that are already in use in the form. For example, you cannot have two columns where the header variables are the same nor can you use two list objects using the same variable headers.

Title:
This is the title of your column. You can enter a title up to 63 characters. You can define a static text or a STR# string (:15000,1 for example). You can change the title with the BUTTON TEXT command

- Theme Picture:
Icon and Name/ID:
You can define a very small picture, an icon, next to the title of your column. You would have to define its origin. It can be an icon from a variable and you would just have to define the name of that variable. You can also select a picture from the picture library and enter the picture ID, or you can select a Resource and define the picture ID for that PICT resource. The picture height would be 14 pixels. You cannot assign a picture through the language. However, you can define a default and empty picture variable and change the contents of that picture later. However, you would have to reload the list box to see the picture change (i.e. close and reload the form).

Mirror Effect:
This feature, available under Windows only allows you to display a mirrored picture. This is available only under Windows and when the system writing direction is from right to left (Arabic, etc.).

- Theme Coordinates and sizing:
Width:
This allows you to define the default width of the column when the list box is displayed for the first time. You may also change that width programmatically later with the SET LISBOX COLUMN WIDTH command. However, you cannot change nor define the width of the last column. This width is computed by 4D as a buffer between the current width accumulated by all other columns and the current width of the list box object itself.

- Theme Text:
This allows you to define text attributes for that column header. You can define the desired settings as already selected for the list box object. However, any of these settings will change the height of the column header, currently hard-coded within 4th

Dimension. You can define the color of the text but you cannot define a background color (from the Properties nor from the language).

- Theme Help:
Help Tip:
This allows you to define a help tip or not for your header column. If your title is not obvious enough, this would be a good choice to expand its definition. You cannot change the Help tip by language. However, you can change the contain of an help tip by using STR# resources. You can define an STR# resource inside the help tip and define that resource string in your structure file.

Property List for the column:
On the other hand, if you select a column, you can see the following in the Property list window.

- Theme Objects:
Object Name and variable name:
That would be the object name and the variable name of the column. It follows the same limitations as defined previously. The variable name would be the name of the array that is to be displayed.
Variable Type:
This is the type of your column. The array type can be Alpha, Text, Numeric, Date, Time, Picture and Boolean. Based on the type, you will be able to choose some display formats. This option will not overwrite the current type of the array.

- Theme Data Source:
Choice List:
You can define a choice list as data source. The choice list will be loaded in the list box only once. A small icon displayed at the right bottom of the cell will let you know that a choice list is available. You can click on that icon to display a popup menu and choice one of available values. You can use the SET CHOICE LIST command to define a choice list for that object. Pass an empty string as list name if you want to remove a previously assigned choice list. You cannot change the contents of that popup menu just by changing the list; you will need to reload the list. This can be done by closing the dialog and redisplaying the list box, or just by executing SET CHOICE LIST with the current list name as parameter. When defining a choice list, the user can enter a value or can choose a value from the popup menu. Be sure that the width of the column is large enough to hold the longest value in your list. The width of the dropdown list displayed by the list is based on the width of the column. If the column is too narrow, the dropdown list will also be too narrow.

Coordinates Sizing:
Width:
This allows you to define the default width of the column when the list box is displayed at the first time. This is the same width that is defined for the column header.

Minimum Width:

This is the minimum width that the column will accept when being resized. The default value is 10 pixels. Be sure that the minimum width is smaller or equal to the maximum width. This property cannot be changed from the language.

Maximum Width:
This is the maximum width that the column can be resized to. The default value is 32000 pixels. You might really consider changing that value for a more realistic value. Be sure that the maximum width is greater or equal to the minimum width. This property cannot be changed from the language.

-Theme Resizing Options:
Resizable:
This item allows you to define whether the column can be resizable or not. If this option is checked, the maximum and minimum widths will not be used. This property cannot be changed from the language.

- Theme Entry:
Enterable:
This option allows you to make that column enterable or not. You will need to set the column to enterable if you want to use the list features. You can use the SET ENTERABLE command to change this setting.

Entry Filter:
This option allows you to define an entry filter; this is similar to all other enterable objects in 4th Dimension (See the 4D 2004 Design Reference manual on page 53). You can use the SET FILTER command to define an entry filter.

- Theme Ranges of Values:
Required list:
This allows you to define a list for required values to be chosen. If you define a choice list and a required list, the required list will overwrite the choice list. The difference with the choice list is that the user cannot enter any value in the column. He can only select a value from the pop-up menu defined by the required list. Required lists still require the column to be enterable in order to enable the popup menu. You cannot define a required list from the language nor can you change that list on the fly. You would have to close and re-open your dialog to see the changes in your required list.

Excluded list:
This allows you to define a list for excluded value. This list does not replace lists defined as Choice Lists or Required Lists. The current behavior is that the user chooses a value from either a Choice List or a Required List. That value is then checked against the values in the excluded list. If that value is in that list, an alert will be displayed ("This value is not allowed") and the value will be rejected. You cannot define an excluded list from the language, nor can you change that list on the fly. You would have to close and re-open the dialog to see any changes to your excluded list applied.

- Theme Display:

Alpha Format, Numerical Format, date Format, Time Format, Picture Format:
This allows you to define the display format for those data type. You can use the default settings or use your own settings. For more information about display formats, you might want to read page 444 of the 4D 2004 Design reference manual. You can use the SET FORMAT command to set these formats. In our example, you can click on the two buttons in the "Formatting" area to see the changes on some columns.

Boolean Format:
The Boolean format will allow you to set how you want to display a Boolean value. You can choose a checkbox with a title or a popup menu with a title. You can use the following syntax **SET FORMAT**(ab_Paid;"Paid") to display a checkbox via the language. If you want to display the popup menu; you would have to use the following syntax: **SET FORMAT**(ab_Paid;"Paid;Unpaid")

Invisible:
The Invisible checkbox allows you to make the column invisible when the list box is displayed for the first time. You can make it invisible or invisible using the SET VISIBLE command.

- Theme Background and Borders:
Background Color/Alternate Background Color:
You can define a background color and an alternate background color (to alternate background colors row by row). Colors can be computed from formulas defined in SET COLOR or SET RGB COLORS and applied to the specific column.

- Theme Text:
Those attributes are the same as those used for other objects such as fields and variables. You can define the font size, the font name or color for that specific column for example. You can specify the use of a style sheet or just hard-code some properties for the list box object.

All chosen text attributes will be applied to that column only. All text attributes can be set from the language (FONT, FONT SIZE, FONT STYLE, SET COLOR, SET ALIGNMENT, etc.). The color commands will allow you to define your background colors. However, the use of constants for default system colors with the SET RGB COLORS command does not work with columns.

- Theme Action:
Method:
This is the object method for the current column.

- Theme Events:
This is the list of all events that the column can fire in order to execute its object method.

Summary

A list box is a very powerful object that allows you to display your arrays with more controls than scrollable areas. Part II of this technical note will focus on handling list boxes through the language.