Picklist *- An enhanced listbox control for Borland's Delphi*

Introduction

The Delphi listbox control is basically a wrapper around the standard Windows listbox control. However, through the extensibility inherent in the VCL, we can enhance the listbox, or any other standard control, to improve its appearance or to add functionality.

I have often had to use multiple-selection listboxes in applications that I've written, and I never really liked how "busy" a listbox got once the user had selected a number of items. I preferred a listbox like the ones in WinCIM, which looked like a series of checkboxes, or the ones in Quicken, where in balancing your checkbook the items you've reconciled are shown in bold.

The **Picklist** control offers these two looks for selected items, and also offers a number of design-time and run-time improvements over the standard Windows control.

I've certainly downloaded more useful free stuff from CompuServe and the Internet than I can ever hope to upload, so I can't see charging for **Picklist**. You may, therefore, use **Picklist** in your applications at no charge. You may freely distribute **Picklist** as long as all files are included intact and the copyright notice in the source code is maintained.

I hope the control is helpful to you!

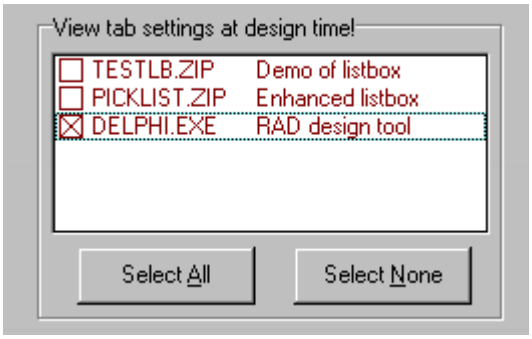
How Picklist Works

Picklist operates just like a regular listbox, both to the developer and to the user, except that it offers a couple of new properties, a couple of new methods, and one new event. One of the new properties is called "SelectedStyle," which offers these four choices:

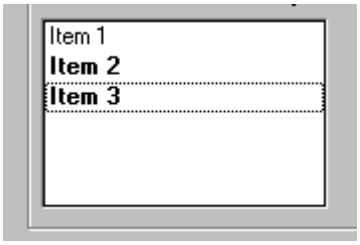
- *psStandard* - the standard Windows listbox
- *psCheckbox* - a Checkbox appears to the left of each listbox string
- *psBoldText* - selected items appear in a bold font
- *psOwnerDraw* - the standard VCL listbox owner-draw mode.

The two new looks appear as shown below:

psCheckbox (Figure 1)



psBoldText (Figure 2)



Another useful feature for the developer is the easy implementation of tab stops. The standard Delphi listbox does not recognize the tab character embedded in listbox strings. With the **Picklist**, there is a property called "UseTabs" which allows you to have the listbox recognize tabs. There is also an easy way to set the tab widths. The Windows API function that sets the tabs uses device units (basically pixels). But how do you figure out how many pixels over to set the tab stop for a column of, say, eight characters? And what if you decide to change the font size?

Picklist lets you set a simple property to establish the tab stops, and even better, lets you see the result at design time. Tab stops are set using the "TabStops" property, which is a list of integral values separated by semicolons (e.g, "10;24;48"). The numbers represent the number of *characters* you want for the tab stop, rather than the number of *pixels*. The control will automatically calculate the appropriate pixel values based on the width of an average character using the listbox font and font size.

Picklist also offers an easy way to implement "Select All" and "Clear Selection" commands. The control has methods called, appropriately enough, "SelectAll" and "ClearSelection." If the **Picklist** is either MultiSelect or ExtendedSelect, invoking these methods will either select all items or remove the selection from all items *very* quickly. In addition, **Picklist** fires an "OnChange" event when the selection changes (Thanks to John Newlin for posting the original code for the "OnChange" method, which I have shamelessly copied virtually note for note.)

Reference

Properties

Align	Hint	SelCount
BorderStyle	IntegralHeight	Selected
Canvas	ItemIndex	**SelectedStyle**
Color	ItemHeight	ShowHint
Columns	Items	Showing
ComponentIndex	Left	Sorted
Ctl3D	MultiSelect	TabOrder
Cursor	Name	TabStop
DragCursor	Owner	**TabStops**
DragMode	Parent	Tag
Enabled	ParentColor	Top
ExtendedSelect	ParentCtl3D	TopIndex
Font	ParentFont	**UseTabStops**
Height	ParentShowHint	Visible
HelpContext	PopupMenu	Width

[Note - the "Style" property from the standard listbox is not published; the SelectedStyle property replaces it]

SelectedStyle

property SelectedStyle: TSelectedStyle;

The SelectedStyle property (one of *psStandard, psCheckbox, psBoldText,* or *psOwnerDraw*) determines the physical appearance of the Picklist. The *psStandard* and *psOwnerDraw* are comparable to the standard *lbStandard* and *lbOwnerDraw* styles. The *psCheckbox* style appers as a checkbox before each list item (see Figure 1 above). The *psBoldText* style displays selected items in a bold font (see Figure 2 above).

Example:

```
MyPicklist.SelectedStyle := psCheckbox;
```

TabStops

property TabStops: string;

The TabStops property defines a list of tab stops for the Picklist control. The list is in the form of integers separated by semicolons. The values represent the number of *characters*, based on the width of the average character for the current font and font size. If the property cannot be parsed as a list of semicolon-separated integers, the control will raise an exception. If only one value is provided, tab stops will be established at each interval of that value (e.g., in the second example below tab stops will be set every twelve characters).

Example:

```
MyPickList.TabStops := '12;24;48';      {sets tabstops at 12, 24, and 48}
MyPickList.TabStops := '12'             {sets tabstops every 12
characters};
```

UseTabStops

property UseTabStops: boolean;

The UseTabStops property allows the Picklist to interpret tab characters (#9) embedded in the listbox strings.

Methods

BeginDrag	GetTextLen	ScrollBy
BringToFront	Hide	**SelectAll**
Clear	ItemAtPos	SendToBack
ClearSelection	Invalidate	SetBounds
ClientToScreen	Refresh	SetFocus
Dragging	Repaint	SetTextBuf
EndDrag	ScaleBy	Show
GetTextBuf	ScreenToClient	Update

ClearSelection

procedure ClearSelection;

Unselects all items in the Picklist. This method uses a Windows API call to clear the selections very quickly. This method is ignored if the Picklist is not ExtendedSelect or MultiSelect.

Example:

```
MyPickList.ClearSelection;
```

SelectAll

procedure SelectAll

Selects all items in a Picklist. This method uses a Windows API call to set the selections very quickly. This method is ignored if the Picklist is not ExtendedSelect or MultiSelect.

Example:

```
MyPickList.SelectAll;
```

Events

OnChange	OnEndDrag	OnMeasureItem
OnClick	OnEnter	OnMouseDown
OnDblClick	OnExit	OnMouseMove
OnDragDrop	OnKeyDown	OnMouseUp
OnDragOver	OnKeyPress	
OnDrawItem	OnKeyUp	

OnChange

property OnChange: TNotifyEvent;

(See online help or VCL Reference, pp. 508ff.)