Static Icons, Picts, Patterns, Colors

Static-type controls can be linked to STR#, PICT, ICON, SICN, cicn, PAT , PAT#, CURS, acur, or clut resources to display the associated icons, pictures, patterns, and colors. The display of these resources can be further modified by varying control colors, frame size, indent size, content limits, etc.

Resource Lists ***IMPORTANT***

• True Res Lists

If the linked resource refers to a list of resources (SICN, PAT#, acur, or clut), then the control value is used to determine which icon, color, etc., from the list to display. Set Value equal to the index into the resource list, and Min and Max equal to the range of possible index values. (Warning: If you set Min = 0, then BaseCt will use the value of Max as the index to display and ignores the control's Value. This is an artifact of the way that BaseCt supports button, check box, and radio controls described in next topic.)

• Other Res Types

If the linked resource is not itself a list (PICT, cicn, ICON, PAT , CURS), and you are interested in showing just one such resource, then you can simply link the resource without worrying about Min, Max, and Value. If you would like to support "flipping" across several such resources in the same control, however, then you can do this by setting Min and Max equal to a range of resource ID numbers, where Min is equal to the same resource ID as that found in the Res Link part of the Control dialog. The Value of the control is then used to determine the resource displayed, where this value can be changed by SetVal or SetCtlValue (discussed below) to support switching resources.

• STR# Res Lists

You can alternatively link a control to an STR# list that specifies any number of resources, of any type, in any order. Each entry in the list should contain a resource type, resource ID, and, if necessary, an index, all separated by commas. The control's Min, Max, and Value are then used to display a resource from the list. For example, the list of strings,

```
PICT,1011
PICT,1012
SICN,1012,3
cicn,1050
```

would display PICT 1011 if Value = 1, PICT 1012 if Value = 2, the 3rd icon in SICN 1012 if Value = 3, and cicn 1050 if Value = 4. (Be careful to use Min > 0 to prevent the list from being used as parameter strings, as described in "Static Text" topic.)

Using string lists to determine which resource to display has the advantage of not requiring that the resources be unique or numbered consecutively, but the disadvantage of requiring management of the extra STR# resource.

Display Options

A linked resource is normally displayed at the top, left of the control, and the control's title is not drawn. If the control's content Max V or Max H is set equal to zero in the Bounds dialog, then the resource will be stretched in that dimension to fit the visible content area of the control. The "Pattern (PAT#)" example control is a good illustration of this option since the pattern that it displays is not distorted by stretching.

If the control's content Max V or Max H is set equal to a positive integer, then the resource is displayed without stretching in that dimension. The "Icon (ICON)" control illustrates this option. If you try stretching this icon control, white space appears at its right and bottom edges, but the ICON continues to be drawn at the top, left of the control.

If 16 is added to the control's VarCode, then the control's title will also be drawn above (= on top of) the displayed resource using the current justification (set in Style menu), and centered vertically.

If 32 is added to the control's VarCode, then the resource being displayed is aligned horizontally in the control's content area according to the control's current text justification (set in Style menu), and centered vertically without stretching (Max V and Max H are ignored). Setting this bit also causes the control's title to be drawn either above (if text just. = centered), at the right of (if left-justified), or at the left of (if right-justified) the resource. This option makes it easy, for example, to mimic standard check boxes and radio buttons with small icons (try the "Standard Mimic (SICN)" check box and radio button example controls).

If 64 is added to the control's VarCode, and the control is not a menu or scrollable list, then the body (background) of the control will not be drawn, even if the control is solid. This is useful in cases where the resource being drawn completely fills the content area of the control, thus making drawing of its body

unnecessary, but where you still want ViewIt to see the control as being "solid". For cursors (CURS, acur) and color icons (cicn), this option also results in not drawing the resource's "mask", which will further reduce the amount of "flashing" seen when redrawing the control. Warning: If the cursor or color icon is irregularly-shaped or does not fill the content area of the control, then do not use this option since the "mask" is necessary to support drawing such shapes.

Hand Scrolling

ViewIt has built-in support for "hand scrolling" (dragging the control's content with a hand cursor). This works best with static controls or views. To set up hand scrolling, first set the control's content Max V or Max H in the Bounds dialog to a value greater than zero, and then check the hand icon. If the content of the control is larger than the control (such as a large PICT), then you'll be able to hand scroll hidden parts of the content (the PICT resource) into view. Note that hand scrolling cannot be used with controls whose contents are centered (32 added to VarCode).

Data Linking

For static non-text controls linked to resource lists (SICN, PAT#, acur, clut, STR#, or resource range), data linking is based on the standard control value. This makes it easy to use SetVal or SetCtlValue to flip from one resource to another. An example of this appears in the vDemoXY program where its nested modal window contains an "animated" control which displays ICONs specified in a linked STR# list.

All other types of static controls base data linking on the linked resource in memory. The only advantage to data linking in this case is that it can be used to get ViewIt to do the SetHandleSize, BlockMove, & DrwCtl that would otherwise be needed to directly update the resource in memory and redraw the control. For example, if a handle variable "showPic" was linked to a static BaseCt PICT control (at v5c2),

```
FaceIt(nil,GetCtl,1005,0,2,5);
FaceIt(nil,LnkCtl,ord(cControl),
ord(@showPic),11,0);
```

then different pictures created by the program could be displayed in this control by calling SetVal:

```
showPic := myPic1;
FaceIt(nil,SetVal,1005,0,2,5);
...
showPic := myPic2;
FaceIt(nil,SetVal,1005,0,2,5);
...
```

where "myPic1" & "myPic2" are handles to pictures created by the program (or loaded from disk), and "1005" is an FWND ID.

Limitations

If a control linked to a resource list has a content size Max V or Max H (set in Bounds dialog) greater than zero, and 32 was not added to its VarCode, then the first resource in the list is used to determine the content area of the control. This limitation will only affect the display of resources in the same list that differ in size (i.e., PICTs larger or smaller than the first PICT in a list will be shrunk or stretched to fit the size of the first PICT). To display PICTs that vary in size within the same control, consider using data linking of a PICT handle in memory to flip from one picture to another (as described above), since this resets the content area of the control on SetVal.