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

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. "SetCtlValue" can be used to flip from one resource in the list to another. NOTE: If you set Min = 0, then BaseCt uses the Max value as the index to display and ignores the Value. This is an artifact of the way that BaseCt supports button, check box, and radio button controls - see next topic.

You can also link a control to an STR# list which specifies any number of resources of any non-text type. Each entry in the list should contain a resource type, ID, and, if necessary, an index, separated by commas. The control's Min, Max, and Value are then used (as described above) to display a resource from the list. For example, the list

```
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.

When linking a non-text control to an STR# list, be careful to use Min > 0 to prevent the list from being used as parameter strings (see "Static Text" topic).

Options

The following bit values can be added to VarCode to set various options:

32: A linked resource is normally displayed at the top, left of the control. If bit value 32 is added to the VarCode, however, then the resource is aligned horizontally in the control's content area according to the control's current text justification (set in Style menu), and centered vertically. Setting this bit also causes the control's title, if any, to be drawn either above (if text justification = centered), at the right (if left-justified), or at the left (if right-justified) of the resource. This makes it easy, for example, to mimic standard check boxes and radio buttons with small icons (try the "SICN Chk Box" and "SICN Rad Btn" examples in FCTL Import menu).

64: Resource types such as cursors (CURS, acur) and color icons (cicn) contain "masks" that are used to help display irregularly shaped pixel maps (white pixels that lie outside the mask are "transparent"). In some cases, drawing of the mask causes flashing of white when one pixel map after another is drawn in rapid succession. This flashing can be eliminated by adding 64 to the VarCode and making the control's body transparent. The color icon or cursor is then "painted" without any erasing or drawing of the mask. The penalty for doing this is that the control will not display a body color, and the icon or cursor cannot be irregularly shaped (white won't be transparent).

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, or STR#), 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.

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, and "1005" is an FWND ID.

Limitations

No support for colors beyond frame, body, & content.