## Scrollable Lists

Scrollable list controls must be of type "List or Menu", and be linked to an STR# resource.   The STR# string list can be created and edited with ResEdit.   UtilIt commands are also available for manipulating string lists in memory (see the UtilIt Guide).   The current list "value" (= the selected item(s)) is set or read through data linking using SetVal & GetVal.

## List Items

The strings in the linked STR# can be either plain text or references to PICT, ICON, SICN, cicn, PAT , PAT#, CURS, acur, or clut resources.   The following string, for example, would display the 4th pattern in PAT# 1000 followed by "Ronald":
    PAT#,1000,4,Ronald
Items can be made inactive by preceding the string with a "-" character:
      -PICT,1005,hello
Note how string elements are separated by commas, and that SICN, PAT#, acur, and clut types require an index value in addition to the resource ID.

## Options

The following bit values can be added to VarCode to set various list options:
2,4,8,16,32,64,128 = "selFlags" defined by List Manager in Volume 4 of Inside Macintosh:
 2 = lNoNilHilite = do not select empty cells
 4 = lUseSense = clicked cell determines action
 8 = lNoRect = (not used by 1-dimensional lists)
 16 = lNoExtend = do not extend Shift selections
 32 = lNoDisjoint = deselect others on mouse click
 64 = lExtendDrag = allow dragging w/o Shift key
 128 = lOnlyOne = one selection at a time
Adding 128 to the VarCode, for example, creates a single-selection type of list, and setting none of the above flags creates a typical multi-selection list.
256 = display list horizontally (default is vertical)
512 = do not show vertical or horizontal scroll bar
1024 = use solid frame rectangle to hilite list items (instead of inverting items)
2048 = always update list with current contents of linked STR# resource when SetVal is called, and update STR# with list contents when GetVal is called
4096 = ignore references to non-text resources in list items and display as text only (do not use this with horizontal lists or option 8192)
8192 = show text below non-text resource (instead of to its right (vertical) or not at all (horizontal))
Horizontal lists have rectangular cells and work best when displaying non-text items.   Vertical lists have cells as wide as the list control and as tall as the list's font, and work best when displaying text items, or text items preceded by small icons.   If the "text below" option is used (8192 in VarCode), then both horizontal and vertical lists have cells that are as tall or wide as the control, with space for displaying text below the non-text resource.

## STR# Link

BaseCt uses the Macintosh List Manager to create and display scrollable lists.   The ListHandle for each list can be found in cHiData (after calling GetCtl) and used with List Manager toolbox calls.
The strings from the linked STR# are copied into the data handle maintained by the List Manager for each list, meaning that later changes to the STR# list will not affect the displayed list.   You can alternatively force the list data and STR# to "track" one another on calls to GetVal and SetVal by adding 2048 to the list control's VarCode.   One artifact of such tracking is that the list is completely redrawn on SetVal, so you should minimize such calls to prevent unnecessary list redrawing.

## Data Linking

A list control's private "value" is an 8-byte integer that corresponds to the items selected in the list.   For single-selection lists (VarCode contains 128), this value is equal to the number of the currently selected list item.   For multi-selection lists, the list's value is equal to the sum of the bit values corresponding to

the selected items in the list.   For example, if the first, third, and fifth items are selected in a multi-selection list, then the list's "value" = 1 + 4 + 16 = 21.   If no items are selected, then the list's value = 0 (for both list types).

   Scrollable lists can be linked to program variables (typically 2, 4 or 8-byte integers - see "Data Links" in the ViewIt Guide for more info on data linking).   On GetVal, an integer value is returned that indicates which items in the list are selected.   On SetVal, the selected list items are updated to reflect the value of the linked program variable.   As described above, you can also have the list contents track the linked STR# resource on GetVal and SetVal.

   The following code, for example, links a 2-byte integer variable named "myInt" to a single-selection list control that is the 3rd control in the 2nd view of the window based on FWND 1005, and then selects the 5th list item:

```
FaceIt(nil,GetCtl,1005,0,2,3);
FaceIt(nil,LnkCtl,ord(cControl), ord(@myInt),2,0);
...
myInt := 5;
FaceIt(nil,SetVal,1005,0,2,3);
```

## Long Lists

   The internal 8-byte (64-bit) integer used to store a list's "value" will be a problem if a multi-selection list has (or can have) more than 64 items.   In this case, the list can be linked to any contiguous block of memory (instead of a single integer) that will be used to get and set selection flags.   This is done by setting the Variable Type of the linked block to 1310 to inform BaseCt that it should treat the data address as the address of a block of bits corresponding to the selection status of items in the list.

   The linked block of bits must be as large or larger than the number of items in the list, otherwise bits beyond the block will get clobbered on GetVal.   The bits in the block should be tested, set, and cleared using the toolbox calls BitTst, BitSet, and BitClr, where the bit offset will be one less than the number of the list item (i.e., zero-based).

   The following code, for example, links a 100-byte block of bits named "myBlock" to a multi-selection list control that is the 3rd control in the 2nd view of the window based on FWND 1005, and then selects the 5th and 6th list items:

```
myBlock : array [1..50] of integer;
...
for i:= 1 to 50 do
  myBlock[i] := 0;
FaceIt(nil,GetCtl,1005,0,2,3);
FaceIt(nil,LnkCtl,ord(cControl), ord(@myBlock),1310,0);
...
BitSet(@myBlock,4);
BitSet(@myBlock,5);
FaceIt(nil,SetVal,1005,0,2,3);
```

## Tips

   Lists work best if there are no partial cells showing.   To create a list with a whole number of cells: 1) first create a list control that is slightly smaller than the desired size, 2) scroll to the end of the list, and 3) go back into editing mode and expand the list 1 pixel at a time until the last item "jumps" into position.

## Limitations

   Although scrollable lists act as single Mac controls, the "GetCtl..." and "SetCtl..." toolbox calls do not get or set the list's state since BaseCt makes no use of the control's "contrlValue", "contrlMin", or "contrlMax".

   When using GetVal or SetVal with multi-selection lists linked to integers, the maximum number of list items that can be affected is 64 since the list uses an 8-byte (64-bit) integer to store its "value".   This restriction does not apply to single-selection lists or to the "long lists" described above.

   Scrollable lists currently do not support right or center justification of list contents.   Lists look and work best with solid bodies, one-pixel frames, and no indent.   No support for control colors beyond frame, body, and content.   No support for hand scrolling.