Macintosh Keyboard Compatibility Issues

Keyboard compatibility is not an issue unless your application needs to use the arrow/cursor keys. When the classic Mac was introduced, the folks at Apple didn't anticipate the need to move the cursor, but subsequent keyboards sported dedicated arrow keys and IM detailed guidelines for the few cases where they could be used.

The few points of divergence in keyboard **‘‘**key codes**’’** have been straightened out with the System 4.1 (and later) built-in key mapping technique, described below. Even so, when you interpret a keyDown event (see **GetNextEvent**), you may prefer to look for **‘‘**char codes**’’** rather than key codes. Some interesting keyCode/charCode combinations follow:

Key Name	charCode	keyCode
Left arrow	0x1C	0x7B
Right arrow	0x1D	0x7C
Up arrow	0x1E	0x7E
Down arrow	0x1F	0x7D
page up	0x0B	0x74
page down	0x0C	0x79
home	0x01	0x73
end	0x04	0x77
clear	0x1B	0x35 (note, same char...)
esc	0x1B	0x47 (...on *clear* and *esc*)
enter	0x03	0x4C
return	0x0D	0x24
forward delete	0x7F	0x75
delete	0x08	0x33
help	0x05	0x72
F1...F15	0x10	(random)

If you wish to differentiate between the ’+' and ’-' of the keypad from the ’+=' and ’_-' keys of the main keyboard, you must look for key codes and you must take precautions with older hardware; the key codes for these keys vary, depending upon the Mac model.

Also note the difficulty in watching for the arrow keys on the classic Mac keypads; the character code for a shifted arrow key will be one of +, *, /, and =. Therefore the common operation of using shifted arrow keys to select text is a bit difficult.

'KMAP' and 'KCHR' Resources

System 4.1 and later provide a flexible key mapping system which eliminates some inconsitencies and provides means to implement dead-key character sets.

First, the keyboard driver maps a scan code, direct from the keyboard, to a *virtual key code*. The mapping is contained in the '**KMAP**' resource in the system file. For the most part, the mapping changes nothing (that is, key code 0x22 maps to virtual key code 0x22). The arrow keys, however, come off the keyboard as 0x4C, 0x4D, 0x4E, and 0x46 and get mapped to virtual

codes of 0x7B through 0x7E. Thus, these values are now consistent across all Macs which have arrow keys and moderately recent system software.

The 'KMAP' resource also provides a mechanism by which any keystroke can cause the driver to send programming codes to the ADB keyboard. For instance, the *caps lock* key causes the caps lock LED indicator (on the **Extended Keyboard**) to toggle on and off.

Next, the keyboard driver figures out the character code for a keyCode+shift-key combination. It notes the current shift-key state and uses that combination to index into one of eight (or more...) tables of characters contained in the '**KCHR**' resource. For instance, the first table contains the unshifted (lowercase) key's character code, the second table is for the shifted (uppercase) state, and so forth.

The 'KCHR' resource contains information on dead-key combinations and there is a full system for building double-key codes such as ñ (opt-n n). See IM pg V-195 and TechNote 160 for related information.