LSetSelect Select or deselect a cell

#include <Lists.h> **List Manager Package**

```
void        LSetSelect(setIt, theCell, theList );
Boolean     setIt ;      TRUE=select theCell, FALSE=deselect it
Cell        theCell ;    the cell to select/deselect
ListHandle  theList ;    handle leading to a ListRec
```

LSetSelect selects or deselects a specified cell in a list. If drawing is on, the cell is redrawn (if necessary) to reflect its changed condition.

setIt is a Boolean that specifies one of:

FALSE	Deselect *theCell* (and unhighlight it)
TRUE	Select *theCell* (and highlight it)

theCell is a Cell (a.k.a. Point) that identifies the cell to select or deselect.

theList is a handle leading to a variable-length ListRec structure. It is a value previously obtained via **LNew**.

Returns: none

Notes: **LSetSelect** lets you select or deselect a cell (altering its highlighting) programmatically, without the user clicking the mouse. Before presenting a list, you may want to pre-select the top item, so the user has some sort of default.

This function is also helpful in handling character-based list searching as implemented in Standard File (for example, when you press the G key, Standard File scrolls to and highlights the first filename starting with "G").

Note: Even if you have set ListRec.selFlags to lOnlyOne (to permit only one selection at a time), **LSetSelect** will NOT automatically deselect before selecting another. You must deselect manually.

After selecting a cell, you can use **LAutoScroll** to bring the cell into the viewing area.

```
char    theChar;                    /* a keyboard character */
Cell    theCell;
pascal Boolean cmp1stChar();        /* custom comparison function */

if ( LSearch( &theChar, 1, cmp1stChar, &theCell, theList ) ){
    LSetSelect( TRUE, theCell, theList );
    LAutoScroll( theList );         /* make sure selection is visible */
}
```

See **LSearch** for an example of how to make a first-character search routine, similar to that used by Standard File.

Another use for **LSetSelect** might be to make a double-clicked selection

blink; e.g.:

```
if ( LClick( localPt, modifiers, theList ) ) { /* on double click */
    theCell = (*theList)->lastClick;           /* which cell was it? */
    for (j=0; j < MenuFlash; j++ ) {           /* off and on a few times */
        LSetSelect( FALSE, theCell, theList );
        Delay( 5, &junk );
        LSetSelect( TRUE, theCell, theList );
        Delay( 5, &junk );
    }
}
```