**SetRect** Assign boundary coordinates to a Rect

#include <Quickdraw.h> **Quickdraw**

```
void    SetRect(theRect, rLeft, rTop, rRight ,rBottom );
Rect    *theRect ;      address of an 8-byte Rect structure
short   rLeft ;         top-left corner
short   rTop ;
short   rRight ;        bottom-right corner
short   rBottom ;
```

**SetRect** is a handy utility function for storing the four boundary coordinates of a rectangle into a Rect structure.

*theRect* is the address of an 8-byte Rect structure. Upon return, it has been filled with the values of *rLeft*, *rTop*, *rRight*, and *rBottom* .

*rLeft* and . .
*rTop* specify the coordinates of the top-left corner of the rectangle, in local coordinates.

*rRight* and . .
*rBottom* specify the coordinates of the bottom-right corner of the rectangle, in local coordinates.

**Returns**: none

---

Notes: **SetRect** provides a simple way to assign two coordinate pairs into an 8-byte structure. It is functionally equivalent to:

```
theRect.left = rLeft;
theRect.top = rTop;
theRect.right = rRight;
theRect.bottom = rBottom;
```

**Tip**: I remember parameter order by the mnemonic "**lit**te**rb**ug"

If you use this a lot (especially in time-critical section of code) you might want to optimize speed by defining a macro to set *theRect*'s fields directly (the Trap overhead takes about three times as long as simply setting the fields).

If you know two Points (e.g., from two recent mouse-down events), you may prefer to use **Pt2Rect** to generate the enclosing rectangle.

Macintosh C compilers are able to directly assign structures. For instance,

```
Rect    theRect;
theRect = thePort->portRect;
```

is a valid syntax. It copies the eight bytes of one Rect structure into another. In fact, you can directly assign the value of any structure to

another of the same type (but you can't compare the contents of two structures).