SetPortBits

Assigns a new bit map to the active GrafPort

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

```
void          SetPortBits(newBitMap );
BitMap        *newBitMap ;   pointer to a BitMap structure
```

SetPortBits replaces the portBits field of the active GrafPort with a new value, effectively changing the entire contents of the port.

Returns: none

Notes: **SetPortBits** is useful for performing off-screen drawing. For instance, you can use a series of Quickdraw calls to create an image in an off-screen memory buffer, then use **CopyBits** to copy the bit-mapped image into the normal screen.

Be sure that *newBitMap* is fully prepared before using this call; that is, the memory for the bit-image has been allocated and the baseAddr, rowBytes, and bounds fields have been set up.

Note that BitMap.rowBytes must be an even number and that it must be as large or larger, in bits, than the width of the BitMap.bounds. The total amount of memory needed for the off-screen bitMap is the product of the height of the rectangle times the bytes-per-row. Here's a formula that performs the calculation:

```
rectHigh = bounds.bottom - bounds.top;
rectWide = bounds.right - bounds.left;
rowBytes = ((rectWide -1) / 16) + 1) * 2;
buffSize = rowBytes * rectHigh;               /* size in bytes */
```

The third line above correctly adjusts for the required word alignment.

Example

```
#include <Quickdraw.h>
#include <Memory.h>

OffScreenDraw(short rWide, short rHigh, Rect rDest )
// rWide, rHigh;       size of off-screen rectangle
// rDest;              on-screen destination */
{
    BitMap     saveBits;
    BitMap     tempBits;
    short      bytesPerRow;

    saveBits = thePort->portBits;         /* save current */
                                          /* now create an off-screen "canvas" */
    bytesPerRow = (( (rWide -1) / 16) + 1) * 2;
    tempBits.baseAddr = (QDPtr)NewPtr( bytesPerRow * rHigh );
    tempBits.rowBytes = bytesPerRow;
    SetRect( &tempBits.bounds, 0,0, rWide, rHigh );
    SetPortBits( &tempBits );                  /* install the new BitMap */

    EraseRect( & (thePort -> portBits.bounds) );/* clear out old material */

        /*  ... Draw some ovals, etc.; normally generate a complex figure ... */

    SetPortBits( &saveBits );             /* restore the old BitMap */
                                          /* copy temp map onto real map */
    CopyBits( &tempBits, &saveBits,  &tempBits.bounds, &rDest, srcOr, nil );

    DisposPtr( (Ptr)tempBits.baseAddr );
}
```