From: austing@Apple.COM (.i.Glenn L. Austin;)
Subject:Re: .c2.Drawing to an off screen bitmap (with code)

In article <8912070531.AA07560@cadman.nyu.edu> deragon@CADMAN.NYU.EDU (John Deragon) writes:

> Hi folks,
> I did a small animation program on my Macintosh Plus,
>it steps through drawing each frame. The problem is, it is damn slow.
>So, faced with the fact that I cant afford a Mac IIcx, I would like to
>know how I could draw frames to an offScreen Bitmap. I am using LSC 4.0.

Drawing into an off-screen bitmap is really quite easy, if you follow a few rules:

1) The rowBytes *MUST* be even, otherwise the odd rows will be on an odd byte, just begging for address error!
2) Create a GrafPort for the new bitmap by calling OpenPort on a newly created GrafPort structure.
3) Save the current port and restore it after drawing to the off-screen port.

Some sample code (off the top of my head) in C:

```
BitMap  bm;  /* my off-screen bitmap */
GrafPort gp; /* my "off-screen grafport */
GrafPtr  oldPort; /* the current grafport */

GetPort(&oldPort); /* get the current grafport */
bm.rowBytes = 20; /* 160 pixels / 8 = 20 bytes */
SetRect(&bm.bounds, 0, 0, 160, 160); /* set the bounds (remember l,t,r,b!) */
bm.baseAddr = NewPtrClear(20 * 160); /* create a new ptr and zero the mem */
if (bm.baseAddr != nil)
{
  OpenPort(&gp); /* create a new, temporary grafport */
  SetPortBits(&bm); /* set the portBits to our bitmap */
  BlockMove((Ptr) &bm.bounds, (Ptr) &gp.portRect, sizeof(Rect)); /* copy the
          rect from bounds to portRect */
  RectRgn(gp.visRgn, &bm.bounds); /* change the visRgn to match */
  SetPort(&gp); /* use our port */

  /* Draw in the port just like you would draw on the screen! */

  SetPort(oldPort); /* restore the old port before disposing our ours! */
  ClosePort(&gp); /* so that the port memory is released *
  /* The bitmap is still around and valid and contains a bitmap of your */
  /* drawing */

  DisposPtr(bm.baseAddr); /* dispose of our off-screen bitmap memory */
}
```

It couldn't be easier! In fact, if you specify the port returned from the PrOpenPage with a SetPort, you are actually drawing into an off-screen bitmap that gets sent to the printer!

Glenn L. Austin

•••

From:-.i.Kelesi
Subject:.c2.How does one set up an offscreen buffer

>How does one set up an offscreen buffer so that CopyBits may be used to
>refresh the window when responding to an update event?

This is really a rather simple procedure. The idea is to draw into an offscreen port and then CopyBits the wanted information to the desired window. To setup an offscreen port for drawing, you can do something like this...

```
#define myBM myPort->portBits
GrafPtr SetupPort()
{
   GrafPtr tmpPort;
   register GrafPtr myPort;

   GetPort(&tmpPort);
   myPort = (GrafPtr) NewPtr(sizeof(GrafPort));
      /* -- Error checking for nil pointer goes here -- */
      /* Create the internal structures for the port */
   OpenPort(myPort);
      /* Now we need to set aside enough memory to draw into */
   myPort->portBits.baseAddr = NewPtr(myPort->portBits.rowBytes * (myPort-
>portBits.bounds.bottom -
      myPort->portBits.bounds.top));
      /* -- Error checking for nil pointer goes here -- */
      /* And return the resulting port, voila, an offscreen graph port! */
   SetPort(tmpPort);
   return myPort;
}
#undef myBM
```

There are several problems with the program as it stands above. 1) it does not have any error conditions. 2) It eats up about 6K of the heap everytime it's called (or more if your doing color, but if that's the case, you'll _need_ to look at Macintosh Technote #120)

Notice that I saved the current port on entry, this is because I didn't really want

SetupPort to change my current graph port. If you don't care, you can remove these lines (but you may never know when you may need them, I spent several hours tracking down a single missing SetPort once...not a pleasant experience.)

You can save memory by passing a window to the procedure and then resizing the port to the window size. If you do this though, you have to resize the port everytime you resize the window, something of a bother. Enough of this, let's describe how to draw, etc.

The procedure to draw into the newly created port may look something like this (in Pascal):

```
DrawInMyPort(port)
GrafPtr port;
{
  GrafPtr tmpPort;

  GetPort(&tmpPort);
  SetPort(port);
     /* various Quickdraw calls here */
  SetPort(tmpPort);
}
```

Pretty easy huh? The way the port is setup, you have the entire 'screen' to draw into, we'd have to do a few more calls if the window are 'larger' than the screen (e.g. paint programs). Finally, we want to do an update event, drawing from the offscreen port to a window:

```
#define NIL (0L)
UpdateWind(wind, port);
WindowPtr wind;
GrafPtr port;
{
  GrafPtr tmpPort;

  GetPort(&tmpPort);
  SetPort(wind);
  BeginUpdate(wind);
  CopyBits(&port->portBits, &wind->portBits, &port->portRect,
     &wind->portRect, srcCopy, NIL);
  EndUpdate(wind);
  SetPort(tmpPort);
}
```

These are all pretty much basic routines here but they should work. If you have any more questions, you might refer to the Macintosh technical notes. (You might be able to find the most recently distributed tn's on comp.mac.sources (comp.sources.mac?)) Specifically, TN#120 (which I know was uploaded recently) Hope this helps!

-Kelesi

•••

From: rcfische@polyslo.CalPoly.EDU (.i.Raymond C. Fischer)
Subject: Re: .c2.Writing large PICT files???

In article <22163.25d040f6@kuhub.cc.ukans.edu> brownrigg@kuhub.cc.ukans.edu writes:
>Can someone describe for me how to create a PICT file; consisting of only
>a bitMap?

This is from memory, so check the calls against Inside Mac.
Assuming the existence of a GrafPort that has your BitMap in it ...

```
    SetPort (your GrafPtr);
    OpenPicture
    CopyBits (thePort^.portBits, thePort^.portBits,
              thePort^.portRect, thePort^.portRect,
              srcCopy, NIL);
    ClosePicture
```

This will create a PICT that has nothing but the bitmap in it. To turn this into a PICT file, create a file of type PICT, write 512 bytes of zeros, then write entire handle contents returned from the OpenPicture/ClosePicture.

Make sure that at some time after opening the port and BEFORE calling OpenPicture, you set the ClipRgn of the port. One way to do this is

```
    ClipRect (thePort^.portRect);
```

If you have only a BitMap and no GrafPort, then open a GrafPort, set the portBits to the bitmap record (using SetPortBits?), set the portRect (as in: thePort^.portRect := thePort^.portBits.bounds), and set the ClipRgn. The do the above CopyBits and then close the port;

If the GrafPort is a CGrafPort, this will produce a type 2 PICT (color) which won't work with any Mac that doesn't have color quickdraw.

Of course, GrafPort is synonymous with WindowRecord in this example.
Any questions?

Ray Fischer   rcfische@polyslo.calpoly.edu