Subject: Optimization Mystery...
Posted by mirek on Sun, 30 Sep 2007 08:31:57 GMT

View Forum Message <> Reply to Message

Well, I decided it is time to perform low-level check... So I have put together Win98 machine with 400Mhz Celeron and compiled UWord.

Found it running very slow, so I started to investigate.

In couple of minutes I have found that following change in the Image::Data::Paint code mostly fixes the problem:

```
/* if(GetKind() == IMAGE_OPAQUE) {
  if(!hbmp) {
   LTIMING("Image Opaque create");
   CreateHBMP(dc, buffer);
  }
  LTIMING("Image Opaque blit");
  HDC dcMem = ::CreateCompatibleDC(dc);
  HBITMAP o = (HBITMAP)::SelectObject(dcMem, hbmp);
  ::BitBlt(dc, x, y, ssz.cx, ssz.cy, dcMem, sr.left, sr.top, SRCCOPY);
  ::SelectObject(dcMem, o);
  ::DeleteDC(dcMem);
  PaintOnlyShrink();
  return;
 }*/
 if(GetKind() == IMAGE_MASK || GetKind() == IMAGE_OPAQUE) { // THIS NOW HANDLES
BOTH CASES
  HDC dcMem = ::CreateCompatibleDC(dc);
  if(!hmask) {
   LTIMING("Image Mask create");
   Buffer<RGBA> bmp(len);
   hmask = CreateBitMask(buffer, sz, sz, sz, bmp);
   ResCount++;
   if(!hbmp)
    CreateHBMP(dc, bmp);
  }
  LTIMING("Image Mask blt");
  HBITMAP o = (HBITMAP)::SelectObject(dcMem, ::CreateCompatibleBitmap(dc, sz.cx, sz.cy));
  ::BitBlt(dcMem, 0, 0, ssz.cx, ssz.cy, dc, x, y, SRCCOPY);
  HDC dcMem2 = ::CreateCompatibleDC(dc);
  ::SelectObject(dcMem2, hmask);
  ::BitBlt(dcMem, 0, 0, ssz.cx, ssz.cy, dcMem2, sr.left, sr.top, SRCAND);
  if(IsNull(c)) {
   ::SelectObject(dcMem2, hbmp);
   ::BitBlt(dcMem, 0, 0, ssz.cx, ssz.cy, dcMem2, sr.left, sr.top, SRCPAINT);
  }
```

```
 else {
  HBRUSH ho = (HBRUSH) SelectObject(dcMem, CreateSolidBrush(c));
  ::BitBlt(dcMem, 0, 0, ssz.cx, ssz.cy, dcMem2, sr.left, sr.top, 0xba0b09);
  ::DeleteObject(::SelectObject(dcMem, ho));
 }
 ::BitBlt(dc, x, y, ssz.cx, ssz.cy, dcMem, 0, 0, SRCCOPY);
 ::DeleteObject(::SelectObject(dcMem, o));
 ::DeleteDC(dcMem2);
 ::DeleteDC(dcMem);
 PaintOnlyShrink();
 return;
 }
```

To explain - now commented part was dealing with the "simple" case of completely opaque Image being sent to display, while now used part deals with this opaque image is if it was complex one.

Now this is weirdest thing I have ever met - much more complex GDI code is faster? WTF?!

Any suggestions why it behaves this way?

Mirek

Subject: Re: Optimization Mystery...
Posted by mirek on Sun, 30 Sep 2007 08:41:02 GMT
View Forum Message <> Reply to Message

Funny, I believe I have just found the reason....

This behaviour prevailed when the display was in 256 colors mode.

Setting to TrueColor immediately fixes everything.

Looks like poor Win98 was forced to perform the conversion of Image format each time for that particular code.

Mirek

Subject: Re: Optimization Mystery...
Posted by mirek on Sun, 30 Sep 2007 08:56:17 GMT
View Forum Message <> Reply to Message

OK, fixed

Subject: Re: Optimization Mystery...
Posted by mr_ped on Sun, 30 Sep 2007 10:16:20 GMT
View Forum Message <> Reply to Message

Any chance to try some linux distro too and look if there's anything to optimize?
(but I think the KDE or GTK will be too slow on such machine, maybe some lightweight X would make more sense. Than again making sure UPP is fast under GTK and KDE is more important, as those two are major players right now).

Subject: Re: Optimization Mystery...
Posted by mirek on Sun, 30 Sep 2007 12:25:28 GMT
View Forum Message <> Reply to Message

Seriously, this issue was very Win32 specific...

Also GTK != Gnome.

Anyway, does this mean you have found performance problem with U++/X11?

Mirek

Subject: Re: Optimization Mystery...
Posted by mr_ped on Sun, 30 Sep 2007 12:40:59 GMT
View Forum Message <> Reply to Message

not really .. on my old 1GHz Athlon the whole KDE is quite slow (when compared to win2k). U++ performs so far just as good like anything else.

Subject: Re: Optimization Mystery...
Posted by mirek on Sun, 30 Sep 2007 14:22:35 GMT
View Forum Message <> Reply to Message

mr_ped wrote on Sun, 30 September 2007 08:40not really .. on my old 1GHz Athlon the whole KDE is quite slow (when compared to win2k). U++ performs so far just as good like anything else.

BTW, usually the problem is with video drivers...

Mirek