Subject: Offset and the clipping stack. How to use?
Posted by mrjt on Tue, 12 Jun 2007 13:47:04 GMT
View Forum Message <> Reply to Message

I'd like to be able to use the following function (from Draw::):
Quote:void Offset(int x, int y)
Creates a new coordinate / clipping entry on top of the stack equal to the previous entry with coordinate origin shifted by (x, y) pixels.

But if this always adds a new entry to the stack, and the stack must end at with the same length as it started, how do you do cumulative offsets?

for example, the code below works because we know how many offsets were pushed:

```
draw.Offset(5, 20);
draw.Offset(5, 20);
draw.Offset(5, 20);
draw.End();
draw.End();
draw.End();
```

But this is not possible if the number of stack pushes is uncertain. Is there a way to either: automatically clear the stack, or add to an offset without pushing on a new stack op?

Cheers,
James

Subject: Re: Offset and the clipping stack. How to use?
Posted by mirek on Tue, 12 Jun 2007 15:17:32 GMT
View Forum Message <> Reply to Message

Well, not really. But you can always do something like:

```
w.Offset(a, b);

.......

w.End(); w.Offset(c, d);

.......

w.End();
```

Some small helper class can make it even better...

Mirek

Subject: Re: Offset and the clipping stack. How to use?
Posted by mrjt on Wed, 13 Jun 2007 10:32:52 GMT
View Forum Message <> Reply to Message

Okay, I see how I can use GetClloffCount to record the stack depth and then trim it to the same level when I'm done, but I'm not sure why all this stack manipulation should be necessary for something so simple. What I realy want is something like the ModelView matrix stack in in OpenGL.

The following methods would do exactly what I want:
Win32:
```
void Draw::AddOffsetOp(Point p)
{
 DrawLock __;
 actual_offset += p;
 LTIMING("Offset");
 SetOrg();
}
```
X11 (not tested):
```
void Draw::AddOffsetOp(Point p)
{
 Cloff &f = cloff.Top();
 actual_offset += p;
 offset[f.offseti] = actual_offset;
}
```
Or does this break something else (Drawing)?

Subject: Re: Offset and the clipping stack. How to use?
Posted by mirek on Wed, 13 Jun 2007 11:09:10 GMT
View Forum Message <> Reply to Message

mrjt wrote on Wed, 13 June 2007 06:32Okay, I see how I can use GetClloffCount to record the stack depth and then trim it to the same level when I'm done, but I'm not sure why all this stack manipulation should be necessary for something so simple.

The main reason is to deliberately enforce that any painting (e.g. Paint) returns the offset and clipping to the state it started with.

You know, e.g., all widgets in a U++ window are painted using the single same Draw (there is just single WM_PAINT/Expose for all widgets). If any Paint routine would move offset out and not restored it, the painting result would be a mess...

Surely, there is an alternative solution that would use helper classes to represent "offseted" or "clipped" Draw, hard to say what is really better. In any case, stack in Draw is good enough to do the task.

Subject: Re: Offset and the clipping stack. How to use?
Posted by mrjt on Wed, 13 Jun 2007 11:27:42 GMT
View Forum Message <> Reply to Message

Fair enough, thanks for the explanation. As you say there are other ways of doing it, I just didn't see why I couldn't short-cut around all of stack manipulation.

Cheers,
James.