Subject: Painter
Posted by mirek on Wed, 14 Jan 2009 12:52:40 GMT
View Forum Message <> Reply to Message

Ok,

so I have renamed SDraw to Painter and officially moved it to uppsrc.

Means, since today, we have high-quality, high-peformance, easy to use software renderer in U++.

There is still a lot of work to be done, especially we need Painting and PaintingPainter classes (to store paintings) and printing support, then printing support in Draw.

Later, I plan for some reingeneering of rendering internal, some AGG design choices are IMO a little bit long in teeth (plus, there is already a couple of bugs I had to fix). OTOH, as source of nice geometrick algorithms, AGG is invaluable

Even later, we need to finally split Draw into (system-GUI-)Draw and CoreDraw (depending only on Core) and make Painter dependent only on CoreDraw (plus font libraries, Fc and Ft).

Mirek

Subject: Re: Painter
Posted by Mindtraveller on Wed, 14 Jan 2009 22:26:01 GMT
View Forum Message <> Reply to Message

I hope this won`t make GUI binaries sizes something like 15Mb, ecpecially with gcc

Subject: Re: Painter
Posted by mirek on Thu, 15 Jan 2009 00:52:27 GMT
View Forum Message <> Reply to Message

Of course, it is not mandatory.

OTOH, so far it is surprisingly small, something like 100KB, and IMO it can be pushed to less.

Mirek

Subject: Re: Painter
Posted by zsolt on Thu, 15 Jan 2009 18:34:48 GMT
View Forum Message <> Reply to Message

Thanks for your work, Mirek!

Subject: Re: Painter
Posted by mirek on Thu, 15 Jan 2009 22:57:06 GMT
View Forum Message <> Reply to Message

examples/PainterExamples contains all you need to use new Painter class.

(Meanwhile, its "Stroker bug" shows one current AGG problem we are about to solve....)

Mirek

Subject: Re: Painter
Posted by chickenk on Fri, 16 Jan 2009 13:21:07 GMT
View Forum Message <> Reply to Message

C:\upp\uppsrc\Painter\/agg_path_storage.h:967: error: 'bezier_arc_svg' was not declared in this scopeI also see in the same file that #include "agg_bezier_arc.h" is commented out, and the header is not present. What was that for ?

Lionel

Subject: Re: Painter
Posted by mirek on Fri, 16 Jan 2009 16:34:54 GMT
View Forum Message <> Reply to Message

Should be now fixed for Linux and GCC.

Also, in Ubuntu64, the size of Painter is now 122KB - measured by comparing reference/Reports size compiled in optimal mode with and without Painter.

Subject: Re: Painter
Posted by chickenk on Fri, 16 Jan 2009 16:49:26 GMT
View Forum Message <> Reply to Message

sorry, I was trying on Windows+MinGW... I got it working by nastily hacking the code (uncommenting the header inclusion, adding it to the sources plus other few fixes).

Speaking of the few fixes, it seems that PrintDlg is #define'd by windows headers and replaced in

Print.cpp by PrintDlgA, thus a compiling error follows... I don't know why it did not happen before. Maybe the reorganization makes the windows header appear earlier.

I added the following :

#ifdef PrintDlg
undef PrintDlg
#endif

and it worked then.

I also had problems with the call to function abs(). I don't know if there is one that exists on windows for double types... So I also added a #define ABS() doing quite the same, and replaced the calls to abs() by ABS().

Just my 2 cents to keep U++ compilable on every platforms... Don't follow my fixes they are ugly, but can give some hints to speed up the real correction.

Lionel

Subject: Re: Painter
Posted by Tom1 on Thu, 22 Jan 2009 13:34:48 GMT
View Forum Message <> Reply to Message

Hi,

Good work with the Painter, Mirek! The interface to various features is very nice once one gets to know it.

Now I would like to know the road-map of the Painter development. (Maybe I have missed it somewhere else in the forum??) Is this new interface something that will be introduced for hardware based rendering too (entirely or partly) or is this only for the software based Painter?

I did some quick benchmarking of the Painter versus Draw and found that the cost of using the Painter instead of Draw is about 10x..20x slowdown for rendering strokes. The fills are not that slow, but they still do not match the speed of Draw. I may well be dead wrong, but I guess the main reason for the huge slowdown is the anti-aliasing. Is there a way to disable anti-aliasing to speed up the rendering at the cost of quality? (I wish there was. Actually, I wish it could be turned on and off on-the-fly.)

My application is very sensitive to speed and anti-aliasing is usually not required. However, the dashed line styles (with great flexibility in Painter, I might add) is what I have been waiting to get for Draw for quite some time now.

Anyway, great stuff!

Best regards,

Tom

Subject: Re: Painter
Posted by mirek on Thu, 22 Jan 2009 14:40:24 GMT
View Forum Message <> Reply to Message

Tom1 wrote on Thu, 22 January 2009 08:34
Now I would like to know the road-map of the Painter development. (Maybe I have missed it somewhere else in the forum??)

Not really, I have to admit that I sort of got interested first and captured later doing this. But it was really missing for a long time.

Quote:
 Is this new interface something that will be introduced for hardware based rendering too (entirely or partly) or is this only for the software based Painter?

It is meant to be solely sw based, at least for now.

Quote:
I did some quick benchmarking of the Painter versus Draw and found that the cost of using the Painter instead of Draw is about 10x..20x slowdown for rendering strokes. The fills are not that slow, but they still do not match the speed of Draw. I may well be dead wrong, but I guess the main reason for the huge slowdown is the anti-aliasing.

IMO, the main difference is the way how strokes are drawn. GDI simply draws lines, at least as long as line width is 1 - and in that case, it is often HW accelerated.

Painter first converts strokes to polygons and then draws them.

In future, we might consider optimizing this and draw strokes directly in important special cases.

BTW, you might also try to benchmark different line widths...

Quote:
Is there a way to disable anti-aliasing to speed up the rendering at the cost of quality? (I wish there was. Actually, I wish it could be turned on and off on-the-fly.)

Yes, there is "NoAA" attribute. But do not expect huge performance gains, it was rather added for printing where antialiasing does not look well.

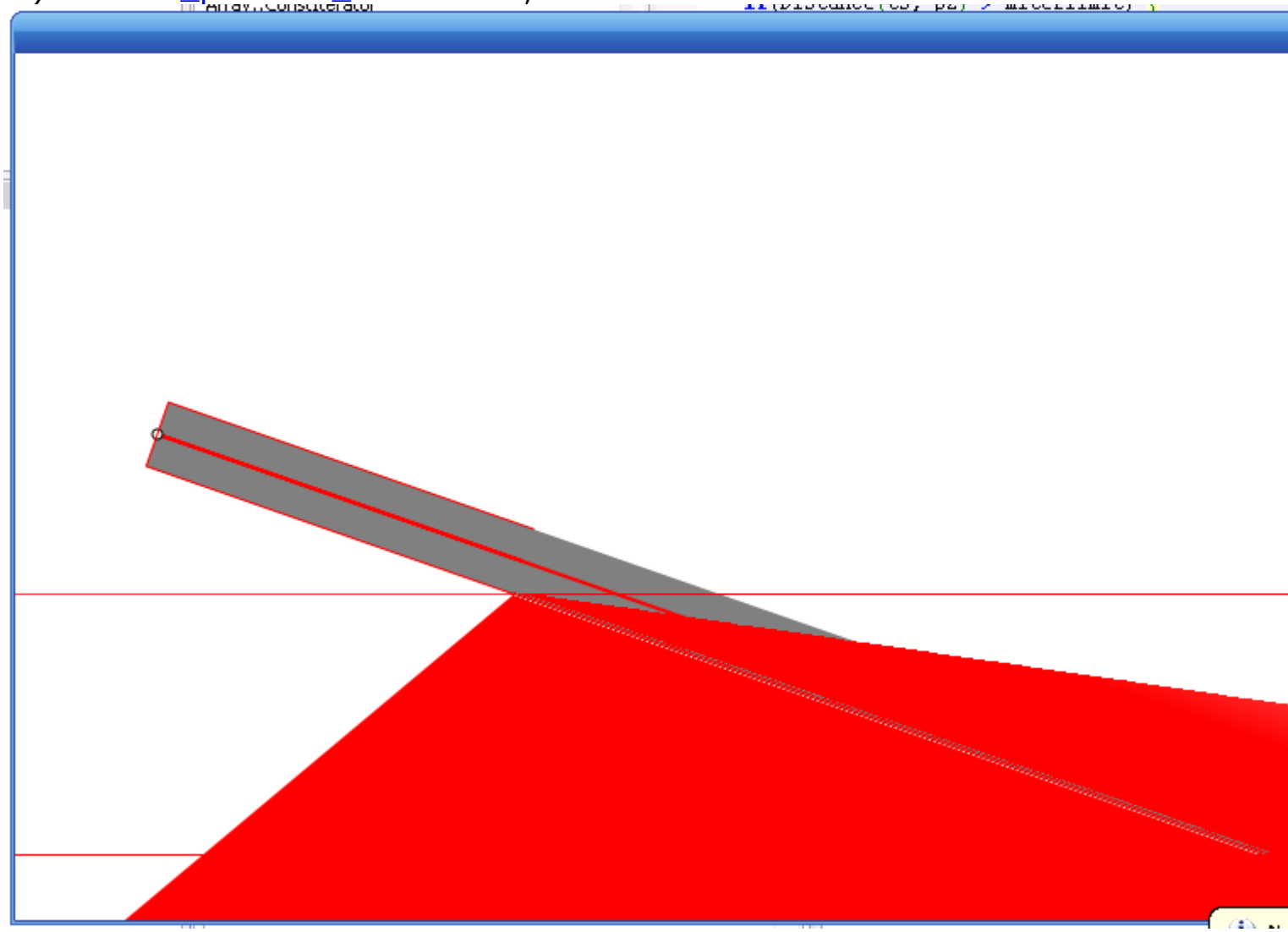
Mirek

Subject: Re: Painter
Posted by tojocky on Thu, 05 Feb 2009 13:14:07 GMT
View Forum Message <> Reply to Message

I tried Stroke example from uppdev.
I have interesting situation in this image:

Is this a normal situation?

File Attachments
1) Stroke_painter_error.PNG, downloaded 1397 times



Subject: Re: Painter
Posted by koldo on Fri, 06 Feb 2009 22:40:54 GMT
View Forum Message <> Reply to Message

Hello Mirek

I am also happy with Painter. Thank you.

As Tom said:
Quote:Now I would like to know the road-map of the Painter development

Now I am doing a very basic library for interactively doing flowcharts. Should I have to port and develop it using Painter?. No problem if the answer is yes.

Best regards
Koldo

Subject: Re: Painter
Posted by mirek on Sat, 07 Feb 2009 10:43:34 GMT
View Forum Message <> Reply to Message

tojocky wrote on Thu, 05 February 2009 08:14I tried Stroke example from uppdev.
I have interesting situation in this image:

Is this a normal situation?

Nope.

Well, "Stroke" was my development package for stroking the path.

For that reason, I was using exisitng AGG based painter for comparison. I believe the bug you see is in AGG

Anyway, that phase is now over, my stroker is working.

Funny thing is that AGG stroker is 800 lines while my new stroker is 150 lines long (and 2KB in compiled code).

I guess that my "AGG-free" painter will squeeze to 20KB of code - something that could be easily added to Draw (or future DrawCore).

Mirek

Subject: Re: Painter

Posted by mirek on Sat, 07 Feb 2009 10:47:35 GMT
View Forum Message <> Reply to Message

koldo wrote on Fri, 06 February 2009 17:40Hello Mirek

I am also happy with Painter. Thank you.

As Tom said:
Quote:Now I would like to know the road-map of the Painter development

Well, I got bored with AGG so started again from the scratch. There is "Painter 2.0" coming - hard parts are finished (rasterizer and stroker), the rest should be quite simple.

Anyway, from the interface perspective, existing methods will stay. The interface will only be enchanced with Pointf based variants (in addition to x, y pars).

Quote:
Now I am doing a very basic library for interactively doing flowcharts. Should I have to port and develop it using Painter?. No problem if the answer is yes.

Well, Painter is here to stay....

Mirek