Subject: Chameleon, graphic technology
Posted by riri on Wed, 12 Apr 2006 10:43:54 GMT
View Forum Message <> Reply to Message

Hi

I were trying to find any information about graphic improvement, like better integrated look & feel and so on on Ultimate++ (because I can't stand the pseudo WinXP L&F on GNU/Linux), and saw chameleon support in the roadmap. Not finding any revelant info with Google (seems this's a third party techno ?), so I tried to find the info in upp forums, but unsuccesfully.

I understood that this chameleon techno will be the solution for look (and even skins) in the future, but what about this techno ? do you have any information about it ? Is the developement started (in this's not a third party solution) ? when do you plan the Big Jump ?

Thanks in advance

Subject: Re: Chameleon, graphic technology
Posted by mirek on Wed, 12 Apr 2006 11:28:17 GMT
View Forum Message <> Reply to Message

riri wrote on Wed, 12 April 2006 06:43Hi

I were trying to find any information about graphic improvement, like better integrated look & feel and so on on Ultimate++ (because I can't stand the pseudo WinXP L&F on GNU/Linux), and saw chameleon support in the roadmap. Not finding any revelant info with Google (seems this's a third party techno ?), so I tried to find the info in upp forums, but unsuccesfully.

I understood that this chameleon techno will be the solution for look (and even skins) in the future, but what about this techno ? do you have any information about it ? Is the developement started (in this's not a third party solution) ? when do you plan the Big Jump ?

Thanks in advance

Well, right now we are in process of refactoring "rendering infrastructure". This is mostly about raster image processing capabilities, which took more time to do than anticipated, OTOH gives unexpectly nice framework for manipulating images. (One of nice features is that the same code can be used to e.g. sharpen in memory icon Image and sharpen 10GB raster file, without any performance compromises on either side).

When that is done, we will most likely have to write new Image designer (with alpha support) and then will be time for Chameleon.

Mirek

Subject: Re: Chameleon, graphic technology
Posted by mr_ped on Wed, 12 Apr 2006 11:42:29 GMT
View Forum Message <> Reply to Message

Mirek, you did not answer him what Chameleon is and who is/will develop it. You just told him the U++ project is not ready yet for the "big jump".

AFAIK Chameleon is in-house code name of the U++ development team, not a third party library. Than again I'm not the right person to answer this.

Subject: Re: Chameleon, graphic technology
Posted by mirek on Wed, 12 Apr 2006 11:59:30 GMT
View Forum Message <> Reply to Message

OK then, Chameleon is the code-name for U++ technology that will (if things work as anticipated)

- provide _flexible_ skinning support for U++

- provide the OS/Platform/Look&Feel detection that shall adjust current skin accordingly.

Mirek

Subject: Re: Chameleon, graphic technology
Posted by riri on Wed, 12 Apr 2006 12:11:19 GMT
View Forum Message <> Reply to Message

Right, thank you both for answer

Is there any roadmap or information on what as to be done, development direction, premilinary ideas and so on ?

That's a kind of coding I'd like to participate on (with ideas or if the time permits to me, in coding), but I've no idea on goals, I guess Mirek's mind is already full of them ?

Maybe it's to early as Mirek said ?

Richard

Subject: Re: Chameleon, graphic technology
Posted by mirek on Wed, 12 Apr 2006 12:27:39 GMT
View Forum Message <> Reply to Message

Well, the basic idea is to introduce "skinning variables" and "skinning expression".

E.g. say HighlightColor can be "skinning variable". It can have default definition "blend(Blue, White)", where Blue and White are another two skinning variables.

Now the idea is that all defualt skinning will be build on several basic variables, heavily reusing former variables in further definitions, e.g.

MenuHighlightColor = "HighlightColor"

That way you should be able to adjust only a couple of expression to get completely different look - changes will propagate through the whole skin definition.

Well, that is the basic idea. None real work have been done yet...

Mirek

Subject: Re: Chameleon, graphic technology
Posted by unodgs on Wed, 12 Apr 2006 16:18:02 GMT
View Forum Message <> Reply to Message

We could also add cairo as a render engine in linux, because now it uses opengl to paint everything (if xgl is run)... (we could be faster than trolltech which plans to use opengl since 4.1 version of qt)

Subject: Re: Chameleon, graphic technology
Posted by mirek on Wed, 12 Apr 2006 16:40:01 GMT
View Forum Message <> Reply to Message

Well, I disagree. After one year of searching, I think that the basic drawing model should be left as it is.

There are too many rendering options now and it will be better to leave them as extensions. Adding cairo (or AGG or GDI+ or QuickDraw) as the only model would IMO have serious impact on portability and performance.

To my knowledge, far not everything is accelerated in cairo. In the end, cairo is 2D memory surface painter for many ops, just like AGG etc... In fact, using OpenGL might work as nice decceleration in many cases (because you are forced to swap memory surfaces there and back all the time).

That said, my current intentions are like this:

Current Draw model being left just where it is. Then add "extensions", like cairo. In the code it should look like this:

void MyCtrl::Paint(Draw& w)

```
{
  Cairo cairo(w);
  cairo.DoStuff()....
}
```

Another reason is the printing problem. I have not checked that completely, but I am afraid then when you are going to print Cairo or AGG produced output, it simply gets rendered as large bitmaps -> slow. My customers would not be pleased when waiting for large reports.

Another things to consider, quite similar, is terminal access and X11 over the NET...

By using "extended" model for parts where it is really needed only, we can (I hope) have best of all worlds...

Mirek

Subject: Re: Chameleon, graphic technology
Posted by unodgs on Wed, 12 Apr 2006 17:12:30 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 12 April 2006 12:40Well, I disagree. After one year of searching, I think that the basic drawing model should be left as it is.

There are too many rendering options now and it will be better to leave them as extensions. Adding cairo (or AGG or GDI+ or QuickDraw) as the only model would IMO have serious impact on portability and performance.

To my knowledge, far not everything is accelerated in cairo. In the end, cairo is 2D memory surface painter for many ops, just like AGG etc... In fact, using OpenGL might work as nice decceleration in many cases (because you are forced to swap memory surfaces there and back all the time).

That said, my current intentions are like this:

Current Draw model being left just where it is. Then add "extensions", like cairo. In the code it should look like this:

```
void MyCtrl::Paint(Draw& w)
{
  Cairo cairo(w);
  cairo.DoStuff()....
}
```

Another reason is the printing problem. I have not checked that completely, but I am afraid then when you are going to print Cairo or AGG produced output, it simply gets rendered as large bitmaps -> slow. My customers would not be pleased when waiting for large reports.

Another things to consider, quite similar, is terminal access and X11 over the NET...

By using "extended" model for parts where it is really needed only, we can (I hope) have best of all worlds...

Mirek

I didn't make myself clear. I wanted cairo to be "under Draw" just like gdi or xlib is now. Just as a next rendering engine (used automaticaly if opengled env is detected). Of course if there is a better lib that use opengl we should go for it. I mentioned cairo because this is the only lib using opengl I know (and which really works (newest gtk use it)).

Subject: Re: Chameleon, graphic technology
Posted by fudadmin on Wed, 12 Apr 2006 17:54:10 GMT
View Forum Message <> Reply to Message

luzr wrote on Wed, 12 April 2006 17:40Well, I disagree. After one year of searching, I think that the basic drawing model should be left as it is.

There are too many rendering options now and it will be better to leave them as extensions. Adding cairo (or AGG or GDI+ or QuickDraw) as the only model would IMO have serious impact on portability and performance.

To my knowledge, far not everything is accelerated in cairo. In the end, cairo is 2D memory surface painter for many ops, just like AGG etc... In fact, using OpenGL might work as nice decceleration in many cases (because you are forced to swap memory surfaces there and back all the time).

That said, my current intentions are like this:

Current Draw model being left just where it is. Then add "extensions", like cairo. In the code it should look like this:

```
void MyCtrl::Paint(Draw& w)
{
  Cairo cairo(w);
  cairo.DoStuff()....
}
```

Another reason is the printing problem. I have not checked that completely, but I am afraid then when you are going to print Cairo or AGG produced output, it simply gets rendered as large bitmaps -> slow. My customers would not be pleased when waiting for large reports.

Another things to consider, quite similar, is terminal access and X11 over the NET...

By using "extended" model for parts where it is really needed only, we can (I hope) have best of all worlds...

Mirek

Yes. I agree with your approach. Even more after I studied Ultimate's Draw. Very nice...!

Subject: Re: Chameleon, graphic technology
Posted by mirek on Wed, 12 Apr 2006 19:00:25 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Wed, 12 April 2006 13:54
Yes. I agree with your approach. Even more after I studied Ultimate's Draw. Very nice...!

Speaking about it, I would like to ask you to start investigating how U++ encapsulation of AGG should look like....

There are some things to consider:

"Draw extensions", as described above, should work with most likely two targets: raw memory surfaces and Drawing.

How interfaces to achieve this will work is much to be decided, however, memory surfaces will be used for screen rendering and also for printing, which should be in general achieved (just like it is now in most cases) by using Drawing interface first, and then some sort of optimalized banding technique - real implementation will depend, but I think that Drawing as intermediate storage media for printing will be very helpful.

I believe that current basic design principle of mostly stateless Draw is sound - at least for kind of applications we usually develop, so please take that into consideration (means, it would be nice to reproduce this on another level with advanced rendering).

Mirek

Subject: Re: Chameleon, graphic technology
Posted by fudadmin on Thu, 13 Apr 2006 03:09:38 GMT
View Forum Message <> Reply to Message

Started...
