Wt - Feature #3606

Make WGLWidget interface as identical as possible to Qt's QGLWidget

09/25/2014 08:37 PM - Prasad Dixit

Status:	InProgress	**Start date:**	09/25/2014
Priority:	Normal	**Due date:**	
Assignee:	Wim Dumon	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

In Qt's QGLWidget, I can use native OpenGL calls such as glClearColor(...) in addition to qglClearColor. I can always use native OpenGL code from other projects and paste it in my sub class of QGLWidget and it works like a charm.

Not the case with Wt's WGLWidget. I need to use enable() instead of glEnable(), depthClamp() instead of glDepthClamp(). For a small project makin gthis changes is fine. But for a large project it can be a colossal waste of time.

I am working on porting a desktop application written in C/Qt/OpenGL to an equivalent web application using Wt. The changes I need to make to code has been pain point in porting to Wt.

I request you to keep interface of WGLWidget identical to Qt's QGLWidget and there should be no need to rename functions from glxxxXXX to xxxXXX for it to compile with Qt. In ideal case a simple search and replace of Qt with Wt and say qxxx with wxxxx should do the 95% job.

BTW, I love what you are doing with Wt. It is a great tool, especially if you have large Qt based code which you need to port to a web application.

History

#1 - 09/26/2014 08:04 PM - Wim Dumon

- Status changed from New to Resolved

Hello,

I just checked in the QGLWidget documentation: there are no members called glEnable(), glDepthClamp(), ... Those functions are OpenGL functions that you call directly.

In Wt, you're not using the server side OpenGL implementation, but a client side version. We've considered prefixing the gl-mapped functions with gl, but decided not to do it in the end to avoid name clashes and wrong user expectations. Our functions are really different, some take other parameter types, and are absolutely not simply the 'normal' opengl functions.

If nevertheless you want to rename these functions, you can do this by implementing a class that inherits from WGLWidget, and add methods with the gl prefix that simply call the methods without the gl prefix. You will then indeed likely have to change less of the code to be ported.

Best regards,

Wim.

#2 - 09/26/2014 08:34 PM - Prasad Dixit

Just saw that status changed from new to resolved. Does that mean it will be available in Wt 3.3.4? Thanks!

Wim Dumon wrote:

> Hello,
>
> I just checked in the QGLWidget documentation: there are no members called glEnable(), glDepthClamp(), ... Those functions are OpenGL functions that you call directly.
>
> In Wt, you're not using the server side OpenGL implementation, but a client side version. We've considered prefixing the gl-mapped functions with gl, but decided not to do it in the end to avoid name clashes and wrong user expectations. Our functions are really different, some take other parameter types, and are absolutely not simply the 'normal' opengl functions.
>
> If nevertheless you want to rename these functions, you can do this by implementing a class that inherits from WGLWidget, and add methods with the gl prefix that simply call the methods without the gl prefix. You will then indeed likely have to change less of the code to be ported.

Best regards,

Wim.

#3 - 09/26/2014 08:53 PM - Prasad Dixit

Oops...didn't read your description. Thanks for the explanation.

#4 - 10/10/2014 05:22 PM - Koen Deforche

- Assignee set to Wim Dumon

#5 - 10/22/2014 02:28 AM - Koen Deforche

- Status changed from Resolved to Closed

#6 - 11/11/2014 12:40 AM - Prasad Dixit

- File WGLWidgetWrapper.h added

As per Wim's suggestion: *"If nevertheless you want to rename these functions, you can do this by implementing a class that inherits from WGLWidget, and add methods with the gl prefix that simply call the methods without the gl prefix. You will then indeed likely have to change less of the code to be ported."*, I am implementing a wrapper class for WGLWidget named WGLWidgetWrapper in order to make OpenGL ES code base portable to Wt with minimal effort. You can see the implementation in attached file. It is not complete yet but you can get a very good idea of what I am trying to accomplish.

Do you see any potential downfall to this approach? Any other issues I should be mindful of (proper static casts, enum definitions etc.)?

@ We've considered prefixing the gl-mapped functions with gl, but decided not to do it in the end to avoid name clashes and wrong user expectations. Our functions are really different, some take other parameter types, and are absolutely not simply the 'normal' opengl functions.

I still believe that it is in Wt's best interest to name WGLWidget's GL specific functions and enums with 'gl' prefix such that there is 1 to 1 mapping to OpenGL ES calls. Considering they are in Wt's namespace (that is why we have namespaces, right?) and and that they are defined in WGLWidget class, I don't see why name clashes would be an issue. As an user of Wt, I don't like the idea of writing and maintaining my own wrapper class of WGLWidget. It is better for everyone if it is part of the Wt library out of the box.

Thank you!

#7 - 11/11/2014 08:38 PM - Koen Deforche

- Status changed from Closed to InProgress

- Priority changed from High to Normal

#8 - 11/12/2014 04:32 PM - Wim Dumon

Hey Prashad,

Skimming through your implementation - did you test if this approach actually works? I'm worried about how you handle program/shader/... IDs in your interface. Server side Wt doesn't know the numeric representation assigned client-side. How will you handle that?

I haven't found a way to stick to the exact OpenGL ES implementation server-side. I'm interested to see how you handle this.

Best regards,

Wim.

#9 - 11/12/2014 05:45 PM - Prasad Dixit

Hi Wim,

I haven't tested it. It was just a conceptual implementation and wanted your feedback to see if it possible.

Equivalent of Program/Shader/AttributeLocation/UniformLocation are GLuint (or unsigned int) in OpenGL ES. In Wt's implementation they inherit from GLObject which has only one data member of type 'int' named _id they and don't have any data members of there own. So I thought it would be possible to cast program/shader/attribute location/uniform location ids passed as GLuint to corresponding Wt's Program/Shader/AttributeLocation/UniformLocation objects. Probably I am wrong as I haven't dug a lot into it.

Do you think there is way to convert say Program ID passed to wrapper function as GLuint to equivalent Program object?

@ Server side Wt doesn't know the numeric representation assigned client-side. How will you handle that?

My knowledge of web technologies is limited (That is why I started to look into Wt so that I could develop Web apps with minimal barrier to entry by using my existing knowledge of C). Can you please explain what do you mean by that?

@ I haven't found a way to stick to the exact OpenGL ES implementation server-side.

I don't expect exact OpenGL ES interface. Considering that some work needs to be done on server side and some on client side, certain differences are inevitable (like JavsScriptMatrix and JavaScriptVector to avoid round trip to server). All I want is that such differences should be kept to minimal for maximum portability. Sticking to the same names for functions and enums as in OpenGL ES will certainly be a step in the right direction.

Thank you!

-Prasad

Files

WGLWidgetWrapper.h	26.7 KB	11/10/2014	Prasad Dixit