Subject: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Tue, 15 Sep 2020 13:46:10 GMT
View Forum Message <> Reply to Message

Hello,

I want to share with you a package I have made to allow visualisation of 3D files and Surface objects (Surface is a class which represent a 3D surface. It can be found in Bazaar)

At the time, SurfaceCtrl allow loading of multiple object at one time, It support several file format of 3D Object (You can find the complete list of supported file-formats here)

SurfaceCtrl include a Arcball camera and a "smart" way to move and rotate arround object via is point and click feature. It also provide an easy way to apply texture on each object loaded and it allow you to see Draw line and normal of each object !

A good way to experiment everythings it do is by compiling the package SurfaceCtrl_Demo located into Bazaar assembly. it contain three differents 3D objects, and show how to apply modification on them !

To rotate camera arround object/ axis, press MouseWheel and move the cursor arround the Ctrl.
To translate camera freely, press MouseWheel + shift and move the cursor arround the Ctrl.
To move an object, Left click it and move arround, To move multiple objects at once, select them by holding shift and move cursor arround without releasing shift. To select all Object, simply Ctrl + A.

If you are aware of bug or want to see some features added, feel free to propose anythings !

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by koldo on Tue, 15 Sep 2020 15:07:08 GMT
View Forum Message <> Reply to Message

Thank you for your generous contribution. I know first hand that it has taken you countless hours to get it.

U++ needed a control for 3D object visualization and this is the answer.
And, although it is certainly very upgradeable, it can be an excellent basis to, with everyone's help, get it integrated into main U++.

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Didier on Fri, 18 Sep 2020 07:55:10 GMT
View Forum Message <> Reply to Message

I join myself to Koldo to thank you,

This control seems very promising and it will be a great contribution to Upp

2 remarks:
  * Eigen is used and the package is not included
  * Eigen has license issues : so could you force the use of only BSD parts of Eigen, otherwise many users won't be able to use you'r Ctrl

Note for all : maybe Eigen should have 2 separate packages :
  * one BSD only (Eigen_BSD)
  * and another one (Eigen_GPL) with all Eigen available (just to be sure no GPL code get's in a commercial APP)
This would also enable having a valid "Copying" file

Continue good work :)

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by koldo on Fri, 18 Sep 2020 16:52:50 GMT
View Forum Message <> Reply to Message

Didier wrote on Fri, 18 September 2020 09:55I join myself to Koldo to thank you,

This control seems very promising and it will be a great contribution to Upp

2 remarks:
  * Eigen is used and the package is not included
  * Eigen has license issues : so could you force the use of only BSD parts of Eigen, otherwise many users won't be able to use you'r Ctrl

Note for all : maybe Eigen should have 2 separate packages :
  * one BSD only (Eigen_BSD)
  * and another one (Eigen_GPL) with all Eigen available (just to be sure no GPL code get's in a commercial APP)
This would also enable having a valid "Copying" file

Continue good work :)
Dear Didier

As Eigen is a header only library, and the Eigen U++ package functions are not used by SurfaceCtrl. it is not explicitly necessary to include it in this case.
However, maybe it would be interesting to include it anyway in SurfaceCtrl.

Eigen package included in U++ has MPL2 license, that is compatible with U++ BSD, and no GPL/LGPL code is included.
See here. Anyway, to avoid any error, the EIGEN_MPL2_ONLY is now defined.

Maybe the reason of the misunderstanding is that the GPL license file is in Eigen... It will be removed immediately.

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by koldo on Fri, 18 Sep 2020 17:12:05 GMT
View Forum Message <> Reply to Message

Well, I only found it in "ToStringPlugin.h" ------> "ToString"GPL"ugin.h", but it's a bit far-fetched, isn't it?

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Fri, 18 Sep 2020 20:57:09 GMT
View Forum Message <> Reply to Message

I have added a dialog box when right clicking the SurfaceCtrl :

Cursor icon is now update depending of action on SurfaceCtrl :
-WheelClicking now show a rotation icon
-Drag and move now show a grabbing hand
-Translation move (shift + WheelClick) now show a quadArrow cursor
-Default cursor is now a black cross

License file have been added

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Fri, 18 Sep 2020 21:10:45 GMT
View Forum Message <> Reply to Message

Didier wrote on Fri, 18 September 2020 09:55I join myself to Koldo to thank you,

This control seems very promising and it will be a great contribution to Upp

Continue good work :)

Thanks !

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Fri, 18 Sep 2020 21:15:05 GMT
View Forum Message <> Reply to Message

koldo wrote on Tue, 15 September 2020 17:07
Thank you for your generous contribution. I know first hand that it has taken you countless hours to get it.

U++ needed a control for 3D object visualization and this is the answer.

Thanks, it have, and it's still a pleasure to work in order to extends Upp functionalities !

koldo wrote on Tue, 15 September 2020 17:07
And, although it is certainly very upgradeable, it can be an excellent basis to, with everyone's help, get it integrated into main U++.

Indeed, if some of you looked at the code and found somethings weird about the way I architected it, please tell me ! a better architecture will increase the upgrade potential

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Klugier on Fri, 18 Sep 2020 21:17:42 GMT
View Forum Message <> Reply to Message

Hello Xemuth,

License for the SurfaceCtrl should be added by adding only Copying file. The reason for that is simply it integrates well with TheIDE. When you click on "File" -> ""License" your license should be there. So, I suggest to remove/rename other files. You do not also need to past there license for other libraries like assip. It should be added automatically by Copying file in this package. BTW, Koldo can you do the same with plugin/assimp I see that there is lack of Copying file and there is LICENSE file. Also, it would be great if Copying file will be attached to package.

If think in your case the license file (Copying) should look as follow:

Copyright (c) 1998, 2020, The U++ Project
All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of

conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR
IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY
AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR
OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

You could place your name there, however I suggest to mention it in documentation and thanks Koldo there. This is general approach for the packages that has aspiration to join uppsrc family :)

My first suggestion is to make as less dependency to other packages as you can.

Klugier

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Sat, 19 Sep 2020 01:37:16 GMT
View Forum Message <> Reply to Message

Hello Klugier,

I have fixed the license, feel free to tell me if something is wrong

---

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Didier on Sat, 19 Sep 2020 10:46:23 GMT
View Forum Message <> Reply to Message

koldo wrote on Fri, 18 September 2020 18:52Dear Didier

As Eigen is a header only library, and the Eigen U++ package functions are not used by SurfaceCtrl. it is not explicitly necessary to include it in this case.
However, maybe it would be interesting to include it anyway in SurfaceCtrl.

Eigen package included in U++ has MPL2 license, that is compatible with U++ BSD, and no GPL/LGPL code is included.
See here. Anyway, to avoid any error, the EIGEN_MPL2_ONLY is now defined.

Maybe the reason of the misunderstanding is that the GPL license file is in Eigen... It will be removed immediately.

Hello Koldo,

You are wright, I saw the COPYING.GPL and COPYING.LGPL files in EIGEN so I supposed there was really GPL code inside ... Happy to hear it isn't the case :)
Eigen header is included in Surface/Surface.h : if it isn't used, it shoudn't be included ==> this was the origin of my remark

As for including it in SurfaceCtrl by default : I think this decision should be left to final user: all packages should be kept as light as possible : this saves some trouble from time to time

Quote:As Eigen is a header only library, and the Eigen U++ package functions are not used by SurfaceCtrl. it is not explicitly necessary to include it in this case.
I am not sure I understand correctly, but eaven if a package is "header only" it should be normally managed with package dependencies if used : this prevents the compiler from using other includes that may be available on you're OS but aren't the same version (This easily happens with boost)

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Klugier on Sat, 19 Sep 2020 22:11:18 GMT
View Forum Message <> Reply to Message

Hello Xmeuth,

Seems that the license problem is handled correctly. Thanks!

I have few remarks to the code:
- In Definition.h "#define GL GLCtrl" this is not needed explicit GLCtrl is fine and additional define can be pass to the user.
- "enum Camera_Movement {CM_FORWARD,CM_BACKWARD,CM_LEFT,CM_RIGHT};" modernized to enum class you will much better type control.
- Camera_Movement CamerMovementDirection shold be enough. I think _ is not needed in public API.
- noexcept - why it is used in so many places. I think it obscures readability and can be ommited. I do not see noexcept such common in uppsrc.

- "void MoveAllSelectedObjects(glm::vec3 move)noexcept; //Move all selected object" - this comment is redurdant to the method signature (it occures in many places). If you want documentation just use Topic++.
- Long functions are implemented in the header file. If function is not one liner then it should be moved to .cpp file.

For example: "OpenGLProgram& SetBool(Upp::String name, bool value)noexcept{if(linked)glUniform1i(GetUniformLocation(name), (int)value);return *this;}" should be moved to .cpp file.

- Shader managment can be placed to separate package. I know we have some shader related stuff in GLDraw, but would be nice if we can unify this. Please check the code behind it
- public method/members should go first - then you should place private variables. The first thing I want to examin is the class public API. I do not want to analyze implementation
- French comments "//Gestion of alpha" - anyway in the modern programming comments should be avoided - your code should be clear enough to exist without comments. Of course, sometimes this rule may be deviated from, but in very rare cases.
- Magic numbers - what is 20000. For more info please visit https://clang.llvm.org/extra/clang-tidy/checks/readability-m agic-numbers.html. Optimally you should define constexpr variables.
- Probably more, but let's start from addressing above issues...

Klugier

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Xemuth on Sun, 20 Sep 2020 14:45:55 GMT
View Forum Message <> Reply to Message

Hello Klugier, Thanks for this code analysis, all the think you said help me to become better ! Thanks a lot !

Klugier wrote on Sun, 20 September 2020 00:11
- In Definition.h "#define GL GLCtrl" this is not needed explicit GLCtrl is fine and additional define can be pass to the user.
- "enum Camera_Movement {CM_FORWARD,CM_BACKWARD,CM_LEFT,CM_RIGHT};" modernized to enum class you will much better type control.
- Camera_Movement CamerMovementDirection shold be enough. I think _ is not needed in public API.

fixed / applied

Klugier wrote on Sun, 20 September 2020 00:11
- noexcept - why it is used in so many places. I think it obscures readability and can be ommited. I do not see noexcept such common in uppsrc.

I have read in a book about advanced C++ that compiler will remove all stuff about exception when compiling a function if it is marked as noexcept, so it optimize the code to run faster. May

the result is not high enought to obstruct readability ?

Klugier wrote on Sun, 20 September 2020 00:11
- "void MoveAllSelectedObjects(glm::vec3 move)noexcept; //Move all selected object" - this comment is redurdant to the method signature (it occures in many places). If you want documentation just use Topic++.

indeed
Klugier wrote on Sun, 20 September 2020 00:11
- Long functions are implemented in the header file. If function is not one liner then it should be moved to .cpp file.

What about header only file ? should I create A cpp file even if in my header only class there is somethings like 3 function which take more than one line ?

Klugier wrote on Sun, 20 September 2020 00:11
- Shader managment can be placed to separate package. I know we have some shader related stuff in GLDraw, but would be nice if we can unify this. Please check the code behind it

I will take a look, since modern OpenGL use (90% of time (because fixed pipeline is strongly deprecated)) shaders are the good way to work .Maybe it would be good to create a GLShader package which goes with GLCtrl package ?
Maybe I can developpe something in that way ?

Klugier wrote on Sun, 20 September 2020 00:11
- public method/members should go first - then you should place private variables. The first thing I want to examin is the class public API. I do not want to analyze implementation

Applied

Klugier wrote on Sun, 20 September 2020 00:11
- French comments "//Gestion of alpha" - anyway in the modern programming comments should be avoided - your code should be clear enough to exist without comments. Of course, sometimes this rule may be deviated from, but in very rare cases.

I guess you are right but atm my code is full of comments so, to remove them I must need documentation but it take time. I will try to remove them in futurs update

Klugier wrote on Sun, 20 September 2020 00:11
- Magic numbers - what is 20000. For more info please visit https://clang.llvm.org/extra/clang-tidy/checks/readability-m agic-numbers.html. Optimally you should define constexpr variables.

Indeed, the readability is better, I will update my code in this sense

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by Klugier on Thu, 24 Sep 2020 21:09:54 GMT
View Forum Message <> Reply to Message

Hello Xemuth,

Quote:
I have read in a book about advanced C++ that compiler will remove all stuff about exception when compiling a function if it is marked as noexcept, so it optimize the code to run faster. May the result is not high enought to obstruct readability ?
This is the micro optimization and it influences the overall code readability. In this case I would suggest to remove all "noexcept". I do not imagine situation when programmers marks almost all of functions/methods with noexpect only for better performance. Please noticed that it was introduced in c++11 and I see massive noexcept for the first time :)

Let's read this article for more info.

"The standard's policy on noexcept is to only mark functions that cannot or must not fail, but not those that simply are specified not to throw exceptions. In other words, all functions that have a limited domain (pass the wrong arguments and you get undefined behavior) are not noexcept, even when they are not specified to throw.".

Quote:
What about header only file ? should I create A cpp file even if in my header only class there is somethings like 3 function which take more than one line ?

Yes this is good practice in cpp world to do not implement in header file. Even if you have one method that requires 3 lines of code it should be placed in .cpp file. For your code the cost of this change is small - all you need to do is just create corresponding .cpp file and include .h add implementation. That's all :) This practice is used in uppsrc. Only methods that maximally execute two commands are placed in header.

Please notice that if you modify header all files that includes that header needs to be recompile. If you change .cpp file compilers needs to recompile only that file. Let's imagine we have several methods implemented in Core.h then one change there will require recompilation of all application.

Quote:
I guess you are right but atm my code is full of comments so, to remove them I must need documentation but it take time. I will try to remove them in futurs update

You can make your code self documented - not need for documentation. However refactoring is required :) From my experience whenever you need to add new comment you could create method with the comment content. Simply, but powerful. If you are interested in this topic you could read Clean Code by Robert C. Martin.

Klugier

Subject: Re: SurfaceCtrl, 3D viewer of Multiple file format and Surface class
Posted by mirek on Wed, 17 Feb 2021 16:57:54 GMT
View Forum Message <> Reply to Message

There are problems with UppHub conversion - this has to be changed:

#define LAYOUTFILE <exemples/SurfaceCtrl_Demo/SurfaceCtrl_Demo.lay>

and being there, please fix the ex_a_mples typo... :)