Subject: Pre processor and macro error
Posted by Xemuth on Mon, 30 Nov 2020 14:22:52 GMT
View Forum Message <> Reply to Message

I am working on GLFW package, I want it to be cross platforme, so as plugin/png I did this kind of trick (only pay attention to Windows part):

According to this image, my "glfw3.h" is interpreted after having defined the target (_GLFW_WIN32)

In "glfw3.h" the file internal.h get included and have this following block of code :

```
#if defined(_GLFW_COCOA)
 #include "cocoa_platform.h"
#elif defined(_GLFW_WIN32)
 #include "win32_platform.h"
#elif defined(_GLFW_X11)
 #include "x11_platform.h"
#elif defined(_GLFW_WAYLAND)
 #include "wl_platform.h"
#elif defined(_GLFW_OSMESA)
 #include "null_platform.h"
#else
 #error "No supported window creation API selected"
#endif
```

A this point, on my computer, the flag _GLFW_WIN32 should have been set but it's not the case. I don't get why

so here come my question, why when having this block :

```
#if defined(_WIN32)
 #define _GLFW_WIN32 1
#elif defined(_APPLE)
 #define _GLFW_COCOA 1
#elif defined(_LINUX)
 #define _GLFW_X11 1
#elif defined(_WAYLAND)
 #define _GLFW_WAYLAND 1
#elif define(_OMESA)
 #define _GLFW_OMESA 1
#endif
```

and being on W10 compiling with CLANG or MSVS don't result in _GLFW_WIN32 being set ?
What I am missing ?

Thanks in advance
Xemuth

Subject: Re: Pre processor and macro error
Posted by Klugier on Mon, 30 Nov 2020 20:54:05 GMT
View Forum Message <> Reply to Message

Hello Xemuth,

Please noticed that GLFW uses CMake as a build system. One of the initial step of this build to is to generate platform specific build basing on CMakeList.txt files. So, I think the flags are generated in this steps, so in our case we need to workaround it if we would like to port it our build system.

So, in your case, all you need to do is to add dependency to upp/Core and in file when flags are defined add something like that:

```
#include <Core/config.h>

#if defined(PLATFORM_WIN32)
 #define _GLFW_WIN32 1
#elif defined(PLATFORM_COCOA)
 #define _GLFW_COCOA 1
#elif defined(PLATFORM_POSIX)
 #define _GLFW_X11 1
#elif defined(flagWAYLAND) // <- Not supported yet, however you could still pass it as package
specific flag
 #define _GLFW_WAYLAND 1
#elif define(flagOMESA) // <- Not supported yet - I don't know what OMESA is? Is it some kind of
software rendering?
 #define _GLFW_OMESA 1
#endif
```

Backing to Wayland here is the documentation how to distinguish it on GTK side. Would be good to have it defined somewhere.

Also, if everything will work as expected please consider make the glfw package available in UppHub :)

If you do not want to use Core/config.h. You could port it like this:

```
#if defined(flagWIN32)
 #define _GLFW_WIN32 1
#elif defined(flagCOCOA)
 #define _GLFW_COCOA 1
```

```
#elif defined(flagX11)
 #define _GLFW_X11 1
#elif defined(flagWAYLAND) // <- Not supported yet, however you could still pass it as package specific flag
 #define _GLFW_WAYLAND 1
#elif define(flagOMESA) // <- Not supported yet - I don't know what OMESA is? Is it some kind of software rendering?
 #define _GLFW_OMESA 1
#endif
```

And then GLFW should accepts WIN32 COCOA X11 WAYLAND OMESA and in the package (application) that use that package all you need to do is manually specify flag in "Main package configuration(s)" dialog.

Klugier

Subject: Re: Pre processor and macro error
Posted by Xemuth on Tue, 01 Dec 2020 10:49:22 GMT
View Forum Message <> Reply to Message

Hello Klugier,

Thanks for your awnser !

I will dig use of <Core/config.h> and try to reproduce all flag Cmake generate depending on configuration.

-WAYLAND, I had never meet any computer using it yet(according to Ubuntu wiki Wayland is a new protocol that enables 3D compositors to be used as primary display servers, instead of running the 3D compositor as an extension under the (2D) X.org display server.) I guess I just have to provide file to GLFW when the system is compatible Wayland, I will try to implement it

-MESA and not 'OMESA' I misspelled it :d (according to MESA home site The Mesa project began as an open-source implementation of the OpenGL specification - a system for rendering interactive 3D graphics.) Same as Wayland, I will implement it.

Quote:please consider make the glfw package available in UppHub
I decided to make my GLFW package (which only worked on Windows) cross-platform when I saw the UppHub announcement in order to share it ! (I also want to share my custom 'game engine' but it's another story :p)