Subject: Compilation on Mac
Posted by Novo on Fri, 14 Oct 2022 18:33:17 GMT
View Forum Message <> Reply to Message

```
./umk uppsrc ide CLANG -bus
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CocoCtrl.h:5:29: error: duplicate member
'WndCaretTime'
	static int		WndCaretTime;
				^
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CtrlCore.h:664:19: note: previous declaration is
here
	static int	WndCaretTime;
			^
...
In file included from /Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CtrlCore.h:802:
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CocoCtrl.h:6:29: error: duplicate member
'WndCaretVisible'
	static bool		WndCaretVisible;
				^
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CtrlCore.h:665:19: note: previous declaration is
here
	static bool	WndCaretVisible;
			^
...
In file included from /Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CtrlCore.h:802:
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CocoCtrl.h:8:14: error: class member cannot be
redeclared
	static void AnimateCaret();
		    ^
/Users/ssg/worker0/m-upp/build/uppsrc/CtrlCore/CtrlCore.h:667:19: note: previous declaration is
here
	static void	AnimateCaret();
			^
...
```

Could you fix that please?
TIA

Subject: Re: Compilation on Mac
Posted by brown on Sun, 01 Jan 2023 01:25:19 GMT
View Forum Message <> Reply to Message

Hi,
there are multiple declarations for several lines in case of X11 and MAC.
A possible workaround could be, if you just simply add an #ifdef block to CtrlCore/CtrlCore.h around line 664-667:

```
#ifndef  GUIPLATFORM_CTRL_DECLS_INCLUDE
static int       WndCaretTime;
static bool      WndCaretVisible;

static void      AnimateCaret();
#endif
```

Subject: Re: Compilation on Mac
Posted by fudadmin on Sun, 01 Jan 2023 15:54:20 GMT
View Forum Message <> Reply to Message

brown wrote on Sun, 01 January 2023 01:25Hi,
there are multiple declarations for several lines in case of X11 and MAC.
A possible workaround could be, if you just simply add an #ifdef block to CtrlCore/CtrlCore.h around line 664-667:

```
#ifndef  GUIPLATFORM_CTRL_DECLS_INCLUDE
static int       WndCaretTime;
static bool      WndCaretVisible;

static void      AnimateCaret();
#endif
```

We are working on it. Would you like to join on github?
https://github.com/ultimatepp/ultimatepp/pull/127

Subject: Re: Compilation on Mac
Posted by fudadmin on Mon, 02 Jan 2023 05:50:13 GMT
View Forum Message <> Reply to Message

I have committed working cocoa changes to my fork repo and created a pull request. You can follow progress here. https://github.com/ultimatepp/ultimatepp/pull/129

Subject: Re: Compilation on Mac
Posted by brown on Mon, 02 Jan 2023 09:24:05 GMT
View Forum Message <> Reply to Message

thx. I forked the master, and I will try to integrate your branch too mine as well. I did focus the configuration and AppKit/objc too. Slightly offtopic here, but:
I have strategic questions mainly.
- are the .mm files objc files ? Is it documented what are the goals there? I wrote a test.cpp and

test.mm and tried to compile it, and got errors with MacOs13.1 Sdk. So, I am wondering how you could compile these nowadays...
-this project doesn't use autoconf, and unused libs are not disabled automatically or command line way, rather it tries to remove the pkg-config --libs part with sed. So, is it a strategy here to manage library depandancies , in example if someone installed libpng or not?
- Finding1: only the lib part is removed by sed, but --cflag part is not consistent with those.
- Finding2: the text is not matching at --cflag replacement line, because of the makefiles content has been changed meanwhile.
My version of configure_makefile contains this now (but this is not the final version, probably it could be a base for further discussions):

```
if [[ "$uname" == 'Darwin' ]]; then
 echo Configuring $1 for MacOS
 # sed -i.bak 's/\`pkg-config --cflags libpng\` \`pkg-config --cflags freetype2\` \`pkg-config --cflags
x11\` \`pkg-config --cflags fontconfig\` \`pkg-config --cflags xcb\` \`pkg-config --cflags expat\`/ /' $1
 `pkg-config --exists libpng` || sed -i.bak 's/\`pkg-config --cflags libpng\`/ /' $1
 `pkg-config --exists freetype2` || sed -i.bak 's/\`pkg-config --cflags freetype2\`/ /' $1
 `pkg-config --exists x11` || sed -i.bak 's/\`pkg-config --cflags x11\`/ /' $1
 `pkg-config --exists xinerama` || sed -i.bak 's/\`pkg-config --cflags xinerama\`/ /' $1
 `pkg-config --exists xrender` || sed -i.bak 's/\`pkg-config --cflags xrender\`/ /' $1
 `pkg-config --exists xft` || sed -i.bak 's/\`pkg-config --cflags xft\`/ /' $1
 `pkg-config --exists xdmcp` || sed -i.bak 's/\`pkg-config --cflags xdmcp\`/ /' $1
 `pkg-config --exists xext` || sed -i.bak 's/\`pkg-config --cflags xext\`/ /' $1
 `pkg-config --exists gtk+3.0` || sed -i.bak 's/\`pkg-config --cflags gtk+-3.0\`/ /' $1
 `pkg-config --exists libnotify` || sed -i.bak 's/\`pkg-config --cflags libnotify\`/ /' $1
 `pkg-config --exists fontconfig` || sed -i.bak 's/\`pkg-config --cflags fontconfig\`/ /' $1
 `pkg-config --exists xcb` || sed -i.bak 's/\`pkg-config --cflags xcb\`/ /' $1
 `pkg-config --exists expat` || sed -i.bak 's/\`pkg-config --cflags expat\`/ /' $1
 sed -i.bak 's/-Wl,--gc-sections $(LINKOPTIONS)/$(LINKOPTIONS)/' $1
 sed -i.bak 's/$(LINKER) -o "$(OutFile)" -Wl,-s $(LIBPATH) -Wl,-O,2 $(LDFLAGS)
-Wl,--start-group/$(LINKER) -o "$(OutFile)" $(LIBPATH) $(LDFLAGS)/' $1
 `pkg-config --exists libpng` || sed -i.bak 's/\`pkg-config --libs libpng\`/ /' $1
 `pkg-config --exists freetype2` || sed -i.bak 's/\`pkg-config --libs freetype2\`/ /' $1
 `pkg-config --exists x11` || sed -i.bak 's/\`pkg-config --libs x11\`/ /' $1
 `pkg-config --exists fontconfig` || sed -i.bak 's/\`pkg-config --libs fontconfig\`/ /' $1
 `pkg-config --exists xcb` || sed -i.bak 's/\`pkg-config --libs xcb\`/ /' $1
 `pkg-config --exists expat` || sed -i.bak 's/\`pkg-config --libs expat\`/ /' $1
 sed -i.bak 's/-lrt / /' $1
 sed -i.bak 's/-Wl,--end-group/ /' $1
fi
```

My sandbox is mainly for learn how to help you the best. For example, how can I get your commits before it's merged, etc. Also, even if was able to eliminate all of the compilation errors, I got linker erros. So, first I have to understand how the AppKit c++/objc is working in u++.
Therefore I have not plan to provide pr now, but at least I would like to learn what is missing at my sandbox.
(the last released mac binaries are working on my mac, but I am not sure, if it is M2 arch rather it is intel and using some rosetta virtualization on my cpu.. So, before I can help anything, I would

like to go through several integration related questions first.)
my fork/sandbox is here: https://github.com/bfarago/ultimatepp/tree/IntegrationTest_o n_Mac_M2
Compare to the master: https://github.com/ultimatepp/ultimatepp/compare/master...bf arago:ultimatepp:IntegrationTest_on_Mac_M2
I did not committed all of my other local changes here, because I've plan to rebase to yours or master branch solution rather.
Happy new year!

Subject: Re: Compilation on Mac
Posted by fudadmin on Mon, 02 Jan 2023 15:21:18 GMT
View Forum Message <> Reply to Message

1.
brown wrote on Mon, 02 January 2023 09:24
- are the .mm files objc files ? Is it documented what are the goals there? I wrote a test.cpp and test.mm and tried to compile it, and got errors with MacOs13.1 Sdk. So, I am wondering how you could compile these nowadays...

*.mm files are ObjectiveC++.
*.m files are ObjectiveC.
*.mm is C++ plus ObjectiveC.

There are different methods if you want to call C++ from ObjC and to call ObjC from C++.

Please post your test files and the method of building(compiling+linking)(ide, umk, xcode, or any other) if you want to know the reasons.
Upp *.mm files serves the goals to call ObjC macos Cocoa GUI framework from upp C++.

2.
brown wrote on Mon, 02 January 2023 09:24
this project doesn't use autoconf, and unused libs are not disabled automatically or command line way, rather it tries to remove the pkg-config --libs part with sed. So, is it a strategy here to manage library depandancies , in example if someone installed libpng or not?

Upp project main building tool is theide.
Umk is more or less auxiliary but it is the main thing if you need bootstrapping to a new platform and/or theide is not working. Or for remote builds.
Umk needs improvement and there are efforts and pull requests on github. You could cooperate and get more info from @Klugier. https:// www.ultimatepp.org/forums/index.php?t=usrinfo&id=1517&am p;
And/or by joining our Discord channel.

3.
brown wrote on Mon, 02 January 2023 09:24
but I am not sure, if it is M2 arch rather it is intel and using some rosetta virtualization on my cpu

Rosetta at the moment. It runs well enough on my M1 MacBookAir.

Subject: Re: Compilation on Mac
Posted by fudadmin on Mon, 02 Jan 2023 16:04:51 GMT
View Forum Message <> Reply to Message

brown wrote on Mon, 02 January 2023 09:24
my fork/sandbox is here: https://github.com/bfarago/ultimatepp/tree/IntegrationTest_o
n_Mac_M2
Compare to the master: https://github.com/ultimatepp/ultimatepp/compare/master...bf
arago:ultimatepp:IntegrationTest_on_Mac_M2
I did not committed all of my other local changes here, because I've plan to rebase to yours or master branch solution rather.
Happy new year!

In your fork I can see you introduced

```
#include <sys/sysctl.h>
```

Are you trying to get cpu type? For what reasons? Building?

If that is the case then this should be sufficient? Eg

```
clang++ main.cpp -target arm64-apple-macos11
```

Starting point would be to create theide build method (*.bm) and test. I will try that later.

More info on (cross)building for arm cpu:
https:// stackoverflow.com/questions/65361672/build-apple-silicon-bin ary-on-intel-machine

---

Subject: Re: Compilation on Mac
Posted by brown on Mon, 02 Jan 2023 17:03:51 GMT
View Forum Message <> Reply to Message

In case of the sysctl.h include is missing from Cpu.cpp, I got 3 compilation errors on Cpu.cpp lines 168,169,173 like undeclared identifier 'CTL_HW', 'HW_MEMSIZE' and sysctl() fn prototype. Perhaps, this is just a side effect of something else, which I didn't find yet... But there is a GetSystemMemoryStatus(r,r) fn implementation in this file, which have multiple "body" regarding to the PLATFORM_MACOS "switch", the referenced fn and sys ctrl codes are came from the sys/sysctl.h include file according to man 3 sysctl in this platform...

I compiled the umk here (because the normal make was getting failed earlier - known fixed reason). So, if I run the
make -f umkMakefile
command, this was the remaining issue, which can be fixed by adding the specific include to the file.
Yeah, probably it should be included from somewere else, I can imagine...

Back to the previous thread (compile mm files), I just wanted to build the umk and theide from its source on mac. There are .mm files in example: Draw/FontCoco.mm.
Some reason, my make still not able to link, because of the Makefile doesn't contains any *.mm files at all... I guess those are missing from my Makefile .
How could you compile the upp itself on Mac with the actual Makefile.in content?

Subject: Re: Compilation on Mac
Posted by fudadmin on Mon, 02 Jan 2023 17:51:22 GMT
View Forum Message <> Reply to Message

brown wrote on Mon, 02 January 2023 17:03In case of the sysctl.h include is missing from Cpu.cpp, I got 3 compilation errors on Cpu.cpp lines 168,169,173 like undeclared identifier 'CTL_HW', 'HW_MEMSIZE' and sysctl() fn prototype.
Perhaps, this is just a side effect of something else, which I didn't find yet... But there is a GetSystemMemoryStatus(r,r) fn implementation in this file, which have multiple "body" regarding to the PLATFORM_MACOS "switch", the referenced fn and sys ctrl codes are came from the sys/sysctl.h include file according to man 3 sysctl in this platform...
I compiled the umk here (because the normal make was getting failed earlier - known fixed reason). So, if I run the
make -f umkMakefile
command, this was the remaining issue, which can be fixed by adding the specific include to the file.
Yeah, probably it should be included from somewere else, I can imagine...

Back to the previous thread (compile mm files), I just wanted to build the umk and theide from its source on mac. There are .mm files in example: Draw/FontCoco.mm.
Some reason, my make still not able to link, because of the Makefile doesn't contains any *.mm files at all... I guess those are missing from my Makefile .
How could you compile the upp itself on Mac with the actual Makefile.in content?

You are trying a very very complicated way... Try 3 simple steps:
1. Please download working macos version from here:
https://sourceforge.net/projects/upp/files/upp/2022.2/
2. It should contain an older working ide. Then, with the ide, try to build at least one reference package. (Report errors here on forums, if any)
3. If that goes ok, create a new uppsrc assembly and point to a folder from my fork branch. Build/execute.

P.S. Yes. You are right . There are problems for the macos version 13 because of sysctl.h. And that include should contain version checking.

Then the problems arise with linking upp x86_64 with macos arm libs...
To solve it would be needed to find how to switch compiling upp into correct CPU_ARM or __aarch64__ or just __arm__ for macos? Or some changes in Core.h would be needed?

P.P.S Putting into *.bm (Build methods) -target arm64-apple-darwin. or arm64-apple-macos* switches upp core/config.h aarch64 correctly. But the compiled *.o files are x86_64. I suspect theide internal builder but need more investigation. Commmand line check clang --version gives arm64. I think, Mirek already has the answers.

Subject: Re: Compilation on Mac
Posted by brown on Tue, 03 Jan 2023 00:56:07 GMT
View Forum Message <> Reply to Message

Thank you, it works with an earlier theide...
I got the point, added your branch as an assembly, using the default arch settings, then I got an error for crypto due to the different arch:
 (): ld: warning: ignoring file /opt/homebrew/Cellar/openssl@3/3.0.7/lib/libcrypto.a,
 building for macOS-x86_64 but attempting to link with file built for macOS-arm64

When I change the build config to add -target arm64-apple-macos11, then I get again the previous problem (sysctl related compiler error).
Therefore I add again the #include <sys/sysctl.h> onto your source tree, and the compilation + link gets success then. (no error, 32 warning)
Furtheremore the newly built "ide" app is running on m2:
% file ./ide
./ide: Mach-O 64-bit executable arm64
while the downloaded previous upp theide is x86
file ./theide
./theide: Mach-O 64-bit executable x86_64

There was an .so / lib problem popup regarding libClang with the new ide, also only an external debugger can be run (lldb) but internal gdb method is not working.
Obviously, I have to set the CLANG.bm correctly. by adding openssl include and lib as well.
I will soon retest my projects based on arm64 and the latest possible sources...

Subject: Re: Compilation on Mac
Posted by fudadmin on Tue, 03 Jan 2023 01:27:35 GMT
View Forum Message <> Reply to Message

brown wrote on Tue, 03 January 2023 00:56Thank you, it works with an earlier theide...
I got the point, added your branch as an assembly, using the default arch settings, then I got an error for crypto due to the different arch:
 (): ld: warning: ignoring file /opt/homebrew/Cellar/openssl@3/3.0.7/lib/libcrypto.a,
 building for macOS-x86_64 but attempting to link with file built for macOS-arm64

When I change the build config to add -target arm64-apple-macos11, then I get again the previous problem (sysctl related compiler error).
Therefore I add again the #include <sys/sysctl.h> onto your source tree, and the compilation + link gets success then. (no error, 32 warning)
Furtheremore the newly built "ide" app is running on m2:
% file ./ide
./ide: Mach-O 64-bit executable arm64
while the downloaded previous upp theide is x86
file ./theide
./theide: Mach-O 64-bit executable x86_64

There was an .so / lib problem popup regarding libClang with the new ide, also only an external debugger can be run (lldb) but internal gdb method is not working.
Obviously, I have to set the CLANG.bm correctly. by adding openssl include and lib as well.
I will soon retest my projects based on arm64 and the latest possible sources...

Fantastic news! Congratulations! I suspect we have a VERY promising member... Would you mind to share your *bm file. And/or test if -target arm64-apple-macos13 works?

P.S. Tested. Mine working CLANG_ARM.bm is as follows (might need to clean some surplus -target...):

```
BUILDER = "CLANG";
COMPILER = "clang++";
COMMON_OPTIONS = "-mmacosx-version-min=13 -DTARGET_CPU_ARM64 -target
arm64-apple-macos13";
COMMON_CPP_OPTIONS = "-std=c++17 -target arm64-apple-macos13 -Wall
-Wno-logical-op-parentheses -Wno-deprecated-anon-enum-enum-conversion
-Wno-deprecated-declarations";
COMMON_C_OPTIONS = "-target arm64-apple-macos13";
COMMON_LINK = "-target arm64-apple-macos13";
COMMON_FLAGS = "";
DEBUG_INFO = "2";
DEBUG_BLITZ = "0";
DEBUG_LINKMODE = "1";
DEBUG_OPTIONS = "-O0";
DEBUG_FLAGS = "";
DEBUG_LINK = "";
RELEASE_BLITZ = "0";
RELEASE_LINKMODE = "1";
RELEASE_OPTIONS = "-O3 -ffunction-sections -fdata-sections";
RELEASE_FLAGS = "";
RELEASE_LINK = "";
DEBUGGER = "gdb";
ALLOW_PRECOMPILED_HEADERS = "0";
DISABLE_BLITZ = "0";
```

PATH = "";
INCLUDE = "/opt/X11/include;/opt/X11/include/freetype2";
LIB = "/opt/X11/lib";
LINKMODE_LOCK = "0";

P.S. It is better to use -arch arm64 flags for compiling and linking than the mentioned above if you do not want restrict your users.

Subject: Re: Compilation on Mac
Posted by fudadmin on Tue, 03 Jan 2023 04:59:20 GMT
View Forum Message <> Reply to Message

If someone wants to try arm version 2022.3 with my fixes you might download it from here
https://github.com/arilect/ultimatepp/releases
Next step would be to tweak the ide to load libclang.dylib from
/Applications/Xcode.app/Contents/Developer/Toolchains/XcodeD efault.xctoolchain/usr/lib

in ide/clang package... or do we have other options to install libclang on macos?
