Subject: problem with compilation/linking - Duplicate .cpp file in single package
Posted by mr_ped on Fri, 14 Sep 2007 00:16:32 GMT
View Forum Message <> Reply to Message

Hello, I'm trying to use UnitTest++ in my project (http://unittest-cpp.sourceforge.net/) and to create a UPP package from it.

But I have run into the limited UPP builder ...

The source contain two files with the same name:
src/Posix/TimeHelpers.cpp
src/Win32/TimeHelpers.cpp

Only one of them should be used during compile, chosen by the target platform.
There's no problem to add #ifdef PLATFORM_WIN32 into that cpp file, so the cpp will compile without problem.
But during linking the linker gets the same .o twice, thus reporting multiple definitions of some code.

I think the .o file names should reflect that those are two different files, and produce two different .o files (like TimeHelpers.o and TimeHelpers(2).o ?? or srcPosixTimeHelpers.o srcWin32TimeHelpers.o (directory path added as prefix to .o file name? This can still lead to duplicity, if somebody has directory structure .../a/a and .../aa .. so this one will not work, the UPP must cleanly recognize all duplicity and resolve it with generated names which don't conflict))

Anyway, IMO another correct solution is to add "exclude from build WHEN" to package settings, so I can exclude the Win32 files on !PLATFORM_WIN32 and exclude the Posix files when on PLATFORM_WIN32.
This still does not solve cpp files with same name in different directories, but it would solve my problem and IMHO it makes sense so You don't need those ugly #ifdef in code, but you can omit whole cpp from building process by flags.

Another problem I run into is that I can't enter "+" into package name when creating new package. I had to edit the dir and .upp files manually on disk to get "UnitTest++" package name.

Subject: Re: problem with compilation/linking - Duplicate .cpp file in single package
Posted by mirek on Tue, 18 Sep 2007 18:31:01 GMT
View Forum Message <> Reply to Message

Well, that really is a trouble...

Anyway, I am not sure I want to further complicate the build process here... I think that workaround are possible, e.g. what about not to put these files directly to the package, but use helper file that includes one of them using #ifdef ?

Mirek

Subject: Re: problem with compilation/linking - Duplicate .cpp file in single package
Posted by mr_ped on Tue, 18 Sep 2007 18:42:49 GMT
View Forum Message <> Reply to Message

So the file will remain in package directory, but it will be not included into .upp file, if I understand you correctly?

I find this solution a bit unfortunate, I prefer to see all relevant files in IDE.

Ah, you gave me some "Ultimate" idea how to solve it properly with current tools.. I'm going to try it... Later.

Subject: Re: problem with compilation/linking - Duplicate .cpp file in single package
Posted by mr_ped on Tue, 18 Sep 2007 19:05:09 GMT
View Forum Message <> Reply to Message

Ok, I tried to split the package into main UnitTest++ and two small UnitTest++[Posix|Win32].
Than I added them in package manager to UnitTest++ main package with "WHEN" on !WIN32 and WIN32.
And finally I changed the #include in sources to point to correct directories.

The result does work as expected, everything is ok.

The problem is it still does not *feel* right.
I will rather stick to renamed files to avoid duplicity, and have them all in one package.

I still think there should be a way to exclude/include files into build process, it feels very weak to have a developing tool where you can't easily force for example file named "generated_templates_with_wrong.extension" to compile as a .cpp.
Including the feature of excluding any .cpp file with "when" just like you can include/exclude whole package at this moment.

Or maybe I'm just too much used to power of Makefiles.

P.S.
If I will ever look into adding "exclude when" for files ... will you use such patch, or you explicitly don't like the idea to have such feature?

Subject: Re: problem with compilation/linking - Duplicate .cpp file in single package
Posted by Zardos on Tue, 18 Sep 2007 19:40:49 GMT
View Forum Message <> Reply to Message

Well, if it is all about a Unittest package you can probably write one which would fit perfectly into U++.

I have attached a zip-file containing 3 files. UTest.cpp, UTest.h and _Test_.cpp

Test.cpp shows an example how to use it.

Basically there are:

```
UTest(testname) {
}

UModule(modulename) {

  int var1;
  XY *xy;

  UInit(modulename) { // Initialize a module
    xy = new XY;
  }

  UTerm(modulename) { // Terminate a module
    delete xy;
  }

  ... following UTest(..) "functions" probably using var1 and xy
}

// to be used inside UTest(...)
UCheck(expression) // checks if an expression evaluates to true (ASSERT for UTest)

UExcept(ExectionClass, functioncall) // checks if an exception is thrown

UTiming(seconds) {// checks if a function executes in a given time
  // Block which gets messured
}
```

These files work not out of the box with U++!!!!

You have to rewrite some functions with the U++ classes and functions. But all in all it's just 160 lines... So it can probably done in 20 minutes.

Basically replace Vec with Vector. replace the foreach(e, c) loops with the U++ style iteration: for(int i = 0, i < ...). Change the Logging functions: Err/Inf to the U++ logging functions and finally replace the Timing class Timer with the U++ version.

Ahh and you have to call RunUTests() somewhere

If I have a little bit time tomorrow I can do it for you. I'm currently just playing around with some other stuff, so it's not in U++ this time.

File Attachments

1) UTest.zip, downloaded 412 times

Subject: Re: problem with compilation/linking - Duplicate .cpp file in single package
Posted by mr_ped on Tue, 18 Sep 2007 22:06:00 GMT
View Forum Message <> Reply to Message

It's not about UnitTest package (I hacked it "to work" within minutes, see http://www.ultimatepp.org/forum/index.php?t=msg&th=1156& amp;start=0& for ready-to-use package).
It's about how to do it in "right way" (and with minimal changes to the 3rd party source), and about my wishes what I miss in Ultimate tool chain.

My current version can't be simply updated to newer one just by replacing sources, all those little changes must be preserved whenever I will upgrade it to newer version. So adding such feature into TheIDE would both bring down the amount of changes needed, and it would also add a feature which I feel is missing (yet I don't have any other source which would take advantage of it, that's the reason why I say "feel").

I think the Core itself would benefit a little bit from it too, so some .cpp files (like Core/Win32Com.cpp) can be without #ifdef on the beginning and end, simply excluded completely from build process on different platforms.