Subject: setting include-path
Posted by amando1957 on Mon, 05 Nov 2007 20:24:47 GMT
View Forum Message <> Reply to Message

Hello folks,

In my first posting, i want say a THANKS SO MUCH to the contributors of U++! Anyone will encounter that there are true prof's at work here, which have obvious pushed the gas pedal heavily. That's it what makes us happy .

Now my question. I have set an include-path for my own headers, but got the error (translated from German):
fatal error C1083: Include-file cannot be opened: 'mystuff.h': No such file or directory
Is that not enough to set the include-path at "setup / build methods"?

with kind greets
Martin

Subject: Re: setting include-path
Posted by mr_ped on Mon, 05 Nov 2007 20:58:16 GMT
View Forum Message <> Reply to Message

Try to check "Setup / Be verbose" and examine the command line used to spawn compiler, this will give you answer if (and how) your settings are used.

Subject: Re: setting include-path
Posted by amando1957 on Thu, 08 Nov 2007 19:35:19 GMT
View Forum Message <> Reply to Message

Hi,

Thanks for the hint. I have found the "be verbose", but i don't really understand: do i need to compile over the command-line, to get the info mentioned? I am not very fit with "cmd".

amando

Subject: Re: setting include-path
Posted by mr_ped on Thu, 08 Nov 2007 20:32:24 GMT
View Forum Message <> Reply to Message

No, just check the "be verbose" ON, and than recompile the package with your source.

In the status window where normally the compilation progress is shown you will see not just which

files and packages are build, but also the command line used to invoke compiler will be printed.

Like this for example:

```
cd /home/ped/UppVA/VATestsAll
VATestsAll.cpp
c++ -c  -I"/home/ped/UppVA" -I"/home/ped/upp/uppsrc" -I"/usr/include/freetype2"
-I"/usr/include/gtk-2.0" -I"/usr/include/glib-2.0" -I"/usr/lib/glib-2.0/include" -I"/usr/lib/gtk-2.0/include"
-I"/usr/include/cairo" -I"/usr/include/pango-1.0" -I"/usr/include/atk-1.0" -DflagMAIN -DflagGCC
-DflagDEBUG -DflagSHARED -DflagDEBUG_FULL -DflagBLITZ -DflagLINUX -DflagNOLIB
-DbmYEAR=2007 -DbmMONTH=11 -DbmDAY=8 -DbmHOUR=21 -DbmMINUTE=24
-DbmSECOND=14 -ggdb -g2  -fexceptions  D_DEBUG -O0 -x c++
"/home/ped/UppVA/VATestsAll/VATestsAll.cpp" -o
"/home/ped/upp/out/VATestsAll/GCC.Debug_full.Main.Shared/VATestsAll.o"
```

Now look for the string you did insert into "includes" where it is used in compiler invocation.

See my -I"/home/ped/UppVA" .. that's what you are looking for.

If you have your include path somewhere on the line, you only need to make sure there's correct option ahead of it (-I also for MSVC? Check it in help).

You don't really need to know cmd.exe well, just check your compiler help for command line options to understand which are used and why.

If it is missing completely, than you put that include path in some wrong place, or you did hit Cancel by accident, or it is applied only in different mode of compilation or simply something went wrong and you will have to find out what.
In such case it may be of some help to post here the content of your package.upp file which will show where the include path is stored.

It's good to know your tools a bit. You don't need to memorize every compilation option, but you should be aware how the final binary is produced, in what steps (preprocessing, object compilation, linking), and what possible options your compiler has... you should read it at least once in life.
(and maybe you will like some of them, like /Wall)

Subject: Re: setting include-path
Posted by mr_ped on Thu, 08 Nov 2007 20:44:27 GMT
View Forum Message <> Reply to Message

Also setting up include paths in Setup/Build methods make it available for any project, which is compiled by the modified build method. That's maybe not what you really want, if those include paths are relevant only for one package.

You may in such case go to Project/Package organizer

There right-click in upper right panel, select New Compiler option, and add your include path WITH the include path option switch.
(In my case with gcc that previous example would be like this:
-I"/home/ped/UppVA")

It's different from build methods dialog, where you need to add only path itself.

And one more thing. The build methods are not stored in package.upp file, so that file will not help to diagnose the problem in case you insist on using the include paths from build methods.

Those are stored elsewhere.. some file with "bm" somewhere around upp dir maybe.

Subject: Re: setting include-path
Posted by amando1957 on Fri, 09 Nov 2007 11:29:39 GMT
View Forum Message <> Reply to Message

hello mr_red,

Quote:See my
-I"/home/ped/UppVA"
.. that's what you are looking for.
I did not find my include path here. I only found those of Upp and MS-VC7. But the following excerpt says, it is searching
in the wrong dir:

F:\upp\MyApps\sometext_t\sometext_t.cpp(36) : fatal error C1083: Include-file cannot be opened: 'sometext.h': No such file or directory

So i know its serching in the "MyApps", where it is not.

Here is my *.upp in case of interest:
Quote:description "test of sometext.h";

uses
 Core;

file
 F:\Cplus\112include\clip.h,
 F:\Cplus\112include\file.h,
 F:\Cplus\112include\getkey.h,
 F:\Cplus\112include\use.h,
 F:\Cplus\112include\codepars.h,
 F:\Cplus\112include\sometext.h,
 sometext_t.cpp;

mainconfig
 "" = "";

Now that does not tell me the was its using the pathes. It is true i can use them all, as long i'm providing the full path. Actually the path installed is only for convenience.

Obvious it is not as simple as i thought. I will study your tips now, and the compiler-options in manual, and then try again, but that will take me some time.

Thanks a lot for your effort, mr_red. If not other, i will write a proggy which accumulates them all to a single long *.h, similar as the preprocessor does, so i only need that.

kind greets
amando

Subject: Re: setting include-path
Posted by mr_ped on Fri, 09 Nov 2007 17:50:37 GMT
View Forum Message <> Reply to Message

So adding "F:/Cplus/112include" in the 'setup / build method' does not work? That's curious, did you add it to correct build method? (the one you use for compilation?)

You may (and should?) still go with the package organizer way, doing "Add compiler option" with include path option and absolute path.

But using external includes/libraries in TheIDE is not as simple as using "packages", so if you by any chance posses sources of those libraries and they are compilable under their own U++ package, I would convert it. (I'm working with Unit++ unit test library in this way, so I don't need to worry about include paths, for packages those are solved by TheIDE)

Subject: Re: setting include-path
Posted by mdelfede on Fri, 09 Nov 2007 18:55:20 GMT
View Forum Message <> Reply to Message

Another option for external include files is to put them inside an assembly path (or, make an assembly path point to include dir...).

Ciao

Max

Subject: Re: setting include-path
Posted by amando1957 on Sat, 10 Nov 2007 19:40:11 GMT
View Forum Message <> Reply to Message

hello folks,

I will try to make a package of it, then its simple, and study the compiler options. Thanks to all for your effort!

sincerly
amando
