Subject: U++ compiles under GCC 4.9.1-2 with argument -std=c++11 are failing.
Posted by Omnipraesens on Sat, 11 Oct 2014 20:17:44 GMT
View Forum Message <> Reply to Message

When using GCC 4.9.1-2 to compile ultimate++ libraries (specifically, nightly #7755) while using the argument -std=c++11 the compile fails. An example of the compiler output can be found at http://pastebin.com/p2xNQ2BZ. This example is from attempting to compile the Address Book XML (DOM-like) example. Other examples fail with similar results.

Does Ultimate++ even support the C++11 standard?

Please note, the compile succeeds when using the -std=c++03 or -std=c++98 arguments instead of the -std=c++11 argument.

Thanks in advance for your time.

Subject: Re: U++ compiles under GCC 4.9.1-2 with argument -std=c++11 are failing.
Posted by dolik.rce on Sun, 12 Oct 2014 07:13:11 GMT
View Forum Message <> Reply to Message

Hi

U++ supports C++11, but it is rather new and perhaps not well tested with all the compiler versions. I'm using GCC 4.9.1 too and I was able to reproduce the problem. The problem is a missing include.

I already sent patch to Mirek, it will be fixed soon. If you want to watch the current status of this issue or see the patch, it is here in redmine.

Best regards,
Honza

Subject: Re: U++ compiles under GCC 4.9.1-2 with argument -std=c++11 are failing.
Posted by Omnipraesens on Sun, 12 Oct 2014 15:21:39 GMT
View Forum Message <> Reply to Message

Thanks for the reply, I'm amazed at how fast you guys respond to issues!
It seems as if the Ultimate++ framework and its' developers deserve a lot more attention than you're currently getting. I would love to see it used more widely; the quality and ingenuity of the code by itself warrants it. U++ Core by itself is HUGE... I don't think anything except Boost has quite the same number of features included within one library; XML and JSON IO, SKYLARK, the NTL... and all those other things that I forgot. To say the least, I'm impressed.

Oh, BTW... does the IDE have a feature allowing me to compile some things as C++11 and some as C++03 within the same package?
That would help with integrating libraries that aren't wholly compatible with C++11. I'm willing to attempt to work on such a feature if said feature does not exist, and if said feature would even be desirable.

Subject: Re: U++ compiles under GCC 4.9.1-2 with argument -std=c++11 are failing.
Posted by dolik.rce on Sun, 12 Oct 2014 17:04:50 GMT
View Forum Message <> Reply to Message

Omnipraesens wrote on Sun, 12 October 2014 17:21Thanks for the reply, I'm amazed at how fast you guys respond to issues!
It seems as if the Ultimate++ framework and its' developers deserve a lot more attention than you're currently getting. I would love to see it used more widely; the quality and ingenuity of the code by itself warrants it. U++ Core by itself is HUGE... I don't think anything except Boost has quite the same number of features included within one library; XML and JSON IO, SKYLARK, the NTL... and all those other things that I forgot. To say the least, I'm impressed.Well, the codebase is huge. But most developpers using it look inside from time to time - either out of curiosity, or because of lacking documentation :) - so there is always enough people who know where to look when a new problem is discovered.

Omnipraesens wrote on Sun, 12 October 2014 17:21Oh, BTW... does the IDE have a feature allowing me to compile some things as C++11 and some as C++03 within the same package?
That would help with integrating libraries that aren't wholly compatible with C++11. I'm willing to attempt to work on such a feature if said feature does not exist, and if said feature would even be desirable.
I don't think there is something that would allow doing this directly. But I guess you could achieve switching between various compilation flags for various files either of these two approaches:

1) Split the code into two separate packages, then you could set extra compiler options for each one. This is probably what you want to avoid, but it is probably the simpler option.

2) Use different extensions for each group of files and handle one using "Custom build steps" feature (see this article for some details). This would of course mean, that you'd effectively bypass theides build system and loose some of the functionality. But it might work if you only need different behavior for few files.

I'm not sure that I'd even try that myself... It might be actually easier to fix the problematic library ;)

Honza
