Subject: Why do we have to manually add Core sub-packages?
Posted by jonarmani on Fri, 29 Jun 2012 15:38:42 GMT
View Forum Message <> Reply to Message

I have done the SMTP example here: http://www.ultimatepp.org/reference$SMTP$en-us.html

I couldn't compile it at first, then eventually figured out I need to add Core/SMTP to the list of packages in the upper left of TheIDE.

But Core was already listed, and my #include statement was to include Core/SMTP/SMTP.h. Shouldn't TheIDE have known where to look for the SMTP package? Why did I have to manually right-click Core, Add Package, and find Core/SMTP?

Thanks,

Jon

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by dolik.rce on Fri, 29 Jun 2012 16:57:28 GMT
View Forum Message <> Reply to Message

jonarmani wrote on Fri, 29 June 2012 17:38I have done the SMTP example here: http://www.ultimatepp.org/reference$SMTP$en-us.html

I couldn't compile it at first, then eventually figured out I need to add Core/SMTP to the list of packages in the upper left of TheIDE.

But Core was already listed, and my #include statement was to include Core/SMTP/SMTP.h. Shouldn't TheIDE have known where to look for the SMTP package? Why did I have to manually right-click Core, Add Package, and find Core/SMTP?

Thanks,

Jon
Hi Jon,

The SMTP package is optional part of Core. Most applications can be compiled without it, so there is no need to introduce the additional files that would only slow down the compilation. So that is why it is not added automatically when you add Core.

Now the more interesting part of your question: Why it is not done automatically... My best guess is that it is because it is not needed that often - just a couple times over the entire application development. But it is an interesting idea and it would be quite easy to implement. It actually makes sense - I always get bored when I have to wait for all the nests being scanned for packages (i have rather slow hdd) when opening the Add package dialog, even though I new exactly what package I want to add and where it resides. Adding the packages automatically would make TheIDE even more efficient. Similarly, the packages could be automatically removed

when the last relevant #include is removed from the code. That would be little trickier to implement, but still probably doable...

There are plans to do some major work on TheIDE in near future, so perhaps this feature might get in as well Thanks for your suggestion!

Best regards,
Honza

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by jonarmani on Fri, 29 Jun 2012 18:18:06 GMT
View Forum Message <> Reply to Message

I see, that makes sense. It is a matter of efficiency.

As far as seeing what you mentioned as a feature of TheIDE, I'm all for it. The purpose of computers/technology is to handle the manual labor for us humans, right?

Thanks so much for your reply.

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by mirek on Tue, 03 Jul 2012 07:41:55 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Fri, 29 June 2012 12:57
The SMTP package is optional part of Core. Most applications can be compiled without it, so there is no need to introduce the additional files that would only slow down the compilation. So that is why it is not added automatically when you add Core.

Now the more interesting part of your question: Why it is not done automatically... My best guess is that it is because it is not needed that often - just a couple times over the entire application development. But it is an interesting idea and it would be quite easy to implement. It actually makes sense - I always get bored when I have to wait for all the nests being scanned for packages (i have rather slow hdd) when opening the Add package dialog, even though I new exactly what package I want to add and where it resides. Adding the packages automatically would make TheIDE even more efficient. Similarly, the packages could be automatically removed when the last relevant #include is removed from the code. That would be little trickier to implement, but still probably doable...

Actually, I was thinking about reversed process. All packages would just #include "uses", which would be generated by TheIDE. You would then just add package in TheIDE and you would not need to add the #include...

Anyway, perhaps it is too huge departure from how things are usually done, given how little often you are adding packages...

Mirek

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by dolik.rce on Tue, 03 Jul 2012 08:47:58 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 03 July 2012 09:41Actually, I was thinking about reversed process. All packages would just #include "uses", which would be generated by TheIDE. You would then just add package in TheIDE and you would not need to add the #include...

Anyway, perhaps it is too huge departure from how things are usually done, given how little often you are adding packages...

Mirek
That would make U++ even more dependent on TheIDE, which is IMHO not good... (It is bad enough as it is now:)) Also, it would probably lead to including much more headers then necessary - not a problem with BLITZ, but as soon as you turn it off, the compilation time would probably increase a lot. And third problem I see is that it would impose a requirement for some mandatory package structure, so that theide knows which headers should be included in "uses".

Honza

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by mirek on Tue, 03 Jul 2012 09:04:29 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Tue, 03 July 2012 04:47mirek wrote on Tue, 03 July 2012 09:41Actually, I was thinking about reversed process. All packages would just #include "uses", which would be generated by TheIDE. You would then just add package in TheIDE and you would not need to add the #include...

Anyway, perhaps it is too huge departure from how things are usually done, given how little often you are adding packages...

Mirek
That would make U++ even more dependent on TheIDE, which is IMHO not good... (It is bad enough as it is now:))

Well, not really. "uses" (or "uses.h") would be a normal file in the package. Simply one source fail that is maintained by theide, but could be maintained by hand as easily..

Quote:
Also, it would probably lead to including much more headers then necessary - not a problem with BLITZ, but as soon as you turn it off, the compilation time would probably increase a lot.

And third problem I see is that it would impose a requirement for some mandatory package structure, so that theide knows which headers should be included in "uses".

Well, it would require some naming standard, but we already have one (the name of main package header same as the name of package), so I would put only such existing headers to "uses".

Using "uses" would not be mandatory, and packages no following the standard could still be added by normal include...

Mirek

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by lectus on Sat, 07 Jul 2012 01:53:44 GMT
View Forum Message <> Reply to Message

How about add a comment to every header file indicating the package name using some kind of standard formating.
For example, first line would be:
// ** Package: Web **
in Web/Web.h.

So when you
#include <Web/Web.h>
then on the first build TheIde would add the package automatically by searching for the first line and extracting the package name.

This wouldn't break the possibility of using U++ in another Ide, because it would just be a comment that only the TheIde uses, and a comment is easily ignored by the compiler, so no overhead on compilation required.

Another cool feature would be to automatically add #include files by parsing the source code. Similar to what Netbeans does with Java. For example when finding String or Vector in code it adds #include <Core/Core.h>
But this would be a little be harder to implement.

Subject: Re: Why do we have to manually add Core sub-packages?

Posted by mirek on Sat, 07 Jul 2012 06:50:07 GMT
View Forum Message <> Reply to Message

lectus wrote on Fri, 06 July 2012 21:53How about add a comment to every header file indicating the package name using some kind of standard formating.
For example, first line would be:
// ** Package: Web **
in Web/Web.h.

Why? I guess it is enough if you find Web.h in Web package (with .upp file there).

Mirek

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by dolik.rce on Sat, 07 Jul 2012 11:58:45 GMT
View Forum Message <> Reply to Message

Out of fun, I implemented the automatic package addition The attached patch contains proof of concept solution, it might be possible to do it simpler, I'm not sure.

The idea is that if Assist is used to select #include file, it is checked if it comes from a U++ package and if this package is not present yet, it is added to the actual package. Can't be much simpler than that

Honza

File Attachments
1) ide.diff, downloaded 416 times

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by mirek on Sat, 15 Nov 2014 17:25:37 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Sat, 07 July 2012 13:58Out of fun, I implemented the automatic package addition The attached patch contains proof of concept solution, it might be possible to do it simpler, I'm not sure.

The idea is that if Assist is used to select #include file, it is checked if it comes from a U++ package and if this package is not present yet, it is added to the actual package. Can't be much simpler than that

Honza

Not sure how or why I have missed this one, but it is now accepted into the trunk - I have only simplified it a bit and also added Prompt confirmation.

Mirek

Subject: Re: Why do we have to manually add Core sub-packages?
Posted by Klugier on Sat, 15 Nov 2014 19:42:31 GMT
View Forum Message <> Reply to Message

Hello Mirek,

Can we add option "Do not ask again" in prompt in use include recognition feature? Moreover it would be nice if it will also be coresponding menu entry to turn this prompt on/off for instant: "Use prompt in include recognition".

Sincerely,
Klugier
