Subject: bug: code completion is not working when namespaces are used
Posted by aftershock on Fri, 08 Jan 2010 11:13:30 GMT
View Forum Message <> Reply to Message

Hi,

The code completion mechanism does not work if you try something like that:
mynamespace::varprefix

Subject: Re: bug: code completion is not working when namespaces are used
Posted by mirek on Sun, 10 Jan 2010 13:44:08 GMT
View Forum Message <> Reply to Message

aftershock wrote on Fri, 08 January 2010 06:13Hi,

The code completion mechanism does not work if you try something like that:
mynamespace::varprefix

namespaces are ignored.

Subject: Re: bug: code completion is not working when namespaces are used
Posted by aftershock on Sun, 10 Jan 2010 15:43:32 GMT
View Forum Message <> Reply to Message

They should not be ignored..Do not you think?

Subject: Re: bug: code completion is not working when namespaces are used
Posted by mirek on Mon, 11 Jan 2010 07:24:20 GMT
View Forum Message <> Reply to Message

aftershock wrote on Sun, 10 January 2010 10:43They should not be ignored..Do not you think?

In ideal world, yes.

In real world, I trade fast autocompletition for ignoring namespaces.

BTW, it is still the same issue: You can do this "correct way": that means running preprocesor on the file each time you request autocomplete and each time you leave the file (to have the global map of symbols).

Preprocessor itself on average file runs for 2-3 seconds. It creates an output that has about one million of lines (because of includes) - then you have to parse this file.

Plus, autocomplete would cease to work in certain header files too...

That is why I believe it is essential to cheat here.

Mirek

Subject: Re: bug: code completion is not working when namespaces are used
Posted by aftershock on Thu, 14 Jan 2010 08:35:42 GMT
View Forum Message <> Reply to Message

I cannot see why would you need to run the preprocessor each time.
You could cache it the symbol table deduced from the preprocessed files.
If the lines before includes change or header file change you could make the program redo the preprocessising.
Otherwise, you could just use the cached information.
In most cases, it would be fast.
