Subject: VC++ on Linux?
Posted by TacticalTaco on Wed, 30 Mar 2016 04:39:24 GMT
View Forum Message <> Reply to Message

I'm asking for a friend...
We're looking for ways to develop cross-platform, and VC++ non-compliance with ISO standards has become a pain. I just found out about U++ looking for ways to deal with this, and it looks like it can compile VC++ 9.0? Everything else I've read says there's no compile for VC++ on linux so I figured I'd ask about that here.

Subject: Re: VC++ on Linux?
Posted by Lance on Fri, 01 Apr 2016 19:12:10 GMT
View Forum Message <> Reply to Message

If what you are looking is cross-platform C++ GUI development, why not try U++?

I vaguely remember somebody is successful in have vc++ earlier version run on wine, but I don't think that's what you're looking for, plus wine will not be able to catch up with the more complicated environment recent vc++ requested.

Subject: Re: VC++ on Linux?
Posted by mr_ped on Sat, 02 Apr 2016 22:20:26 GMT
View Forum Message <> Reply to Message

If I understand your question correctly, you misunderstood the thing.

Nobody cares about VC9 under linux, we simply use GCC or Clang. For some time I also did build windows binaries with TDM (gcc for win fork), but nowadays I don't have to support win anymore, so I'm not aware of current windows compilers situation.

If you are starting some project from scratch, the best way is to set up the project from the start on all platforms (for example VC9+win vs GCC+linux), and compile everything each day or so (or to have CI server). And write the source in compatible way, so it's compilable also under VC9, although sometimes it's a bit of pain.

If you have legacy code, which doesn't compile under GCC, then maybe you should fix it, but where to get the budget and not steal... no idea, sorry.

Anyway, Ultimate++ is cross platform framework, so for many OS things and similar it will provide you with unified solution (a bit limited when compared to native specialized thing, but most of the time fully sufficient). So you have to develop only inner parts of app, and just build it on all supported platforms from the same source. That's already super convenient, if you ask me (But I

have to support android, so I actually build in TheIDE only for linux, then I run android NDK build externally, so for me there's some hassle involved, but negligible to things like JNI and Java hell).