

Forum:[Hack Forums](#)

Thread: [CodeBlocks arguments?](#)

ShallowME - September 23, 2020, 5:15 am

While compiling with the command line instead of IDE, how do I get the same arguments that CodeBlocks uses as default ones while compiling let's say for Console and such?

Thanks

HFDone - September 23, 2020, 5:51 am

you can use many compilers with codeblocks

mingw gcc is the most used and you can check settings compilers to find out

i personally don't use ide and using compilers cmd line

ShallowME - September 23, 2020, 6:10 am

(Sep 23, 2020 05:51 AM)HFDone Wrote: [\[CITE\]](#) you can use many compilers with codeblocks

mingw gcc is the most used and you can check settings compilers to find out

i personally don't use ide and using compilers cmd line

Yes, I meant mingw gcc as well. Compiling using commandline v/s the IDE doesn't give me the same result and I want to know the arguments it uses and such.

HFDone - September 23, 2020, 9:42 am

it depends there are many options

you can google gcc \ gcc++ compiler command line

<https://gcc.gnu.org/onlinedocs/gcc/Invoking-GCC.html>

Sajuuk - September 24, 2020, 7:47 pm

Standard base GCC arguments are as follows:

Code gcc "main.c" -o "main.o"

That, by itself, would simply compile the source file into object code ready to be linked. However most if not all C compilers nowadays integrate the linker into the compiler somehow. That being the case with GCC, neither you nor Code::Blocks have any inherent requirement to call the linker separately. I compile from command line with the very line above all the time, given there aren't any special circumstances where I'd otherwise add additional arguments.

I doubt Code::Blocks calls that line verbatim, though. Even if it did, that's what you'll want to use unless it's a large project with multiple files to be linked. It can be done via command line but in my opinion that's where either an IDE or Makefiles come into play. Those methods were put into action for that very reason and they've always done it very well. If you need to, you can check on additional arguments and options in either the manual included in your GCC version's package or by calling "gcc --help" in the command prompt. (IIRC it's "--help", although the options "/" and "-h" may also work in the same manner.) HFDone has linked to the online documentation. I suggest that over reading walls of text within CMD.