0.6 — Installing an Integrated Development Environment (IDE)

BY ALEX ON MAY 28TH, 2007 | LAST MODIFIED BY ALEX ON JANUARY 26TH, 2020

An **Integrated Development Environment (IDE)** is a piece of software that contains all of the things you need to develop, compile, link, and debug your programs.

With a typical C++ IDE, you get a code editor that does line numbering and syntax highlighting. Many (but not all) IDEs include a C++ compiler and a linker, which the IDE will know how to interface with in order to convert your source code into an executable file. And when you need to debug your program, you can use the integrated debugger.

Furthermore, IDEs typically bundle a number of other helpful editing features, such as integrated help, name completion, auto-formatting, and sometimes a version control system. So while you could do all of these things separately, it's much easier to install an IDE and have them all accessible from a single interface.

So let's install one! The obvious next question is, "which one?". Many IDEs are free (in price), and you can install multiple IDEs if you wish, so there's no "wrong decision" to be made here. We'll recommend a few of our favorites below.

If you have some other IDE in mind, that's fine too. The concepts we show you in these tutorials should generally work for any decent modern IDE. However, various IDEs use different names, layouts, key mappings, etc... so you may have to do a bit of searching in your IDE to find the equivalent functionality.

Tip

To get the most value of this tutorial, we recommend installing an IDE that comes with a C++17 capable compiler. Most of the lessons and examples will work with C++14 and C++11. However, if you encounter a lesson that uses concepts from C++17 and you're using an older language compiler, you'll have to translate it to your version, which may or may not be easy.

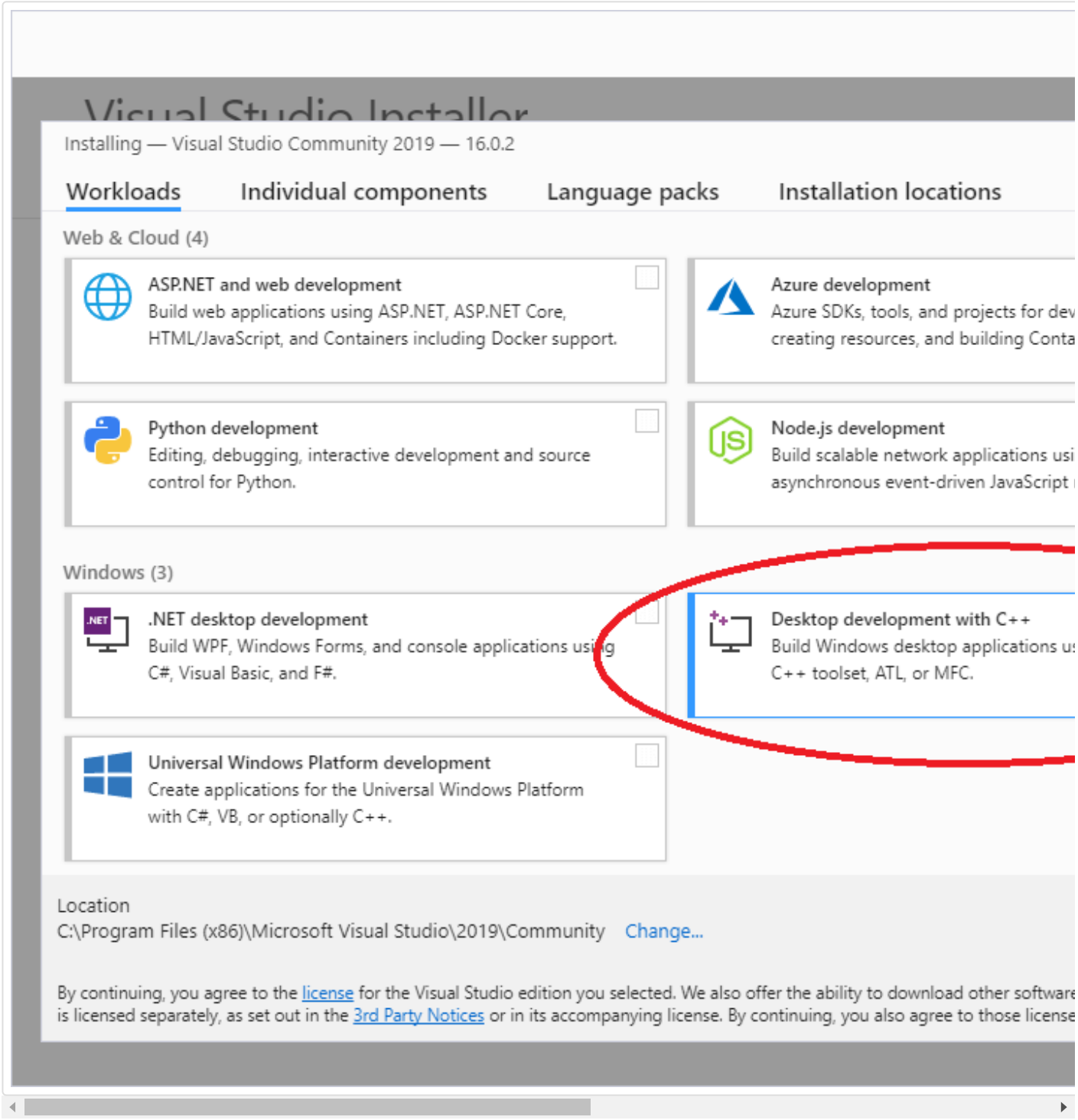
We recommend installing the newest version of a compiler. If you can't use the newest version, these are the absolute minimum compiler versions with C++17 support:

- GCC/G++ 7
- clang++ 8
- Visual Studio 2017 15.7

Visual Studio (for Windows)

If you are developing on a Windows machine (as most of you are) and disk space and download size are not a constraint, then we strongly recommend **Visual Studio Community**. When you run the installer, you'll eventually come to a screen that asks you what workload you'd like to install. Choose *Desktop development with C++*. If you do not do this, then C++ capabilities will not be available.

The default options selected on the right side of the screen should be fine, but please ensure that the *Windows 10 SDK* is selected. The Windows 10 SDK can be used on older versions of Windows, so don't worry if you're still running Windows 7 or 8.



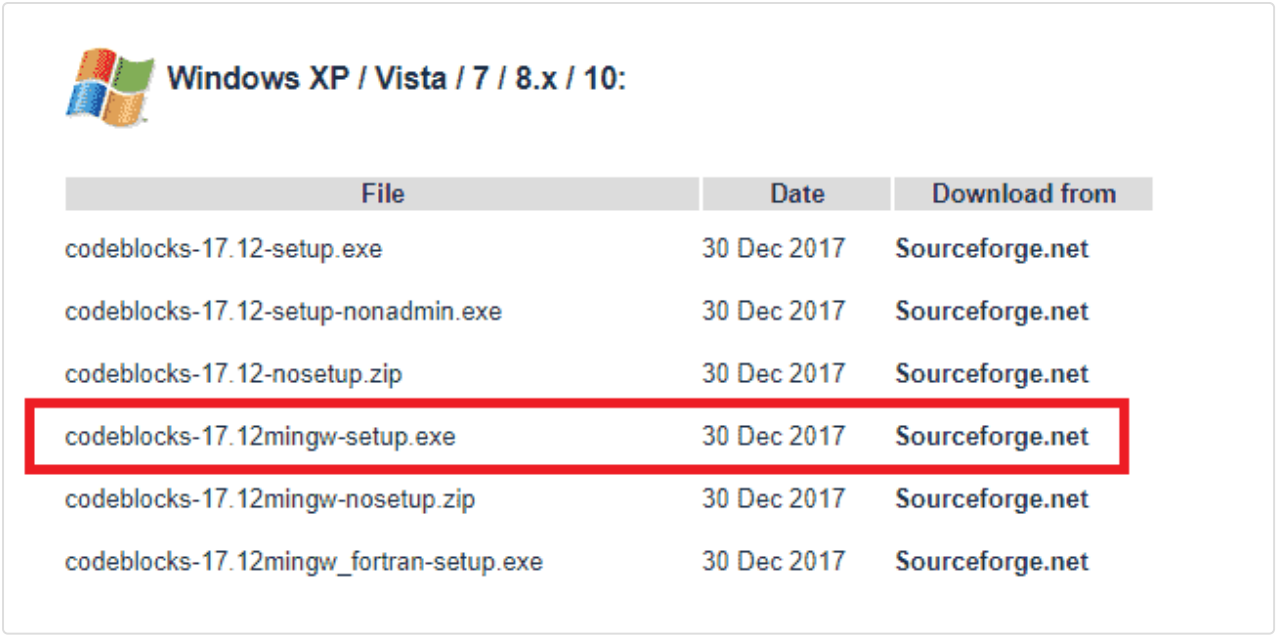
If disk space and/or download size are a challenge, then we recommend Microsoft's free **Visual Studio Express 2017 for Windows Desktop**, which you can find towards the bottom of the page.

Code::Blocks (for Linux or Windows)

If you are developing on Linux (or you are developing on Windows but want to write programs that you can easily port to Linux), we recommend **Code::Blocks**. Code::Blocks is a free, open source, cross-platform IDE that will run on both Linux and Windows.

For Windows users

Make sure to get the version of Code::Blocks that has MinGW bundled (it should be the one whose filename ends in *mingw-setup.exe*). This will install MinGW, which includes a Windows port of the GCC C++ compiler:



Windows XP / Vista / 7 / 8.x / 10:

File	Date	Download from
codeblocks-17.12-setup.exe	30 Dec 2017	Sourceforge.net
codeblocks-17.12-setup-nonadmin.exe	30 Dec 2017	Sourceforge.net
codeblocks-17.12-nosetup.zip	30 Dec 2017	Sourceforge.net
codeblocks-17.12mingw-setup.exe	30 Dec 2017	Sourceforge.net
codeblocks-17.12mingw-nosetup.zip	30 Dec 2017	Sourceforge.net
codeblocks-17.12mingw_fortran-setup.exe	30 Dec 2017	Sourceforge.net

For Linux users

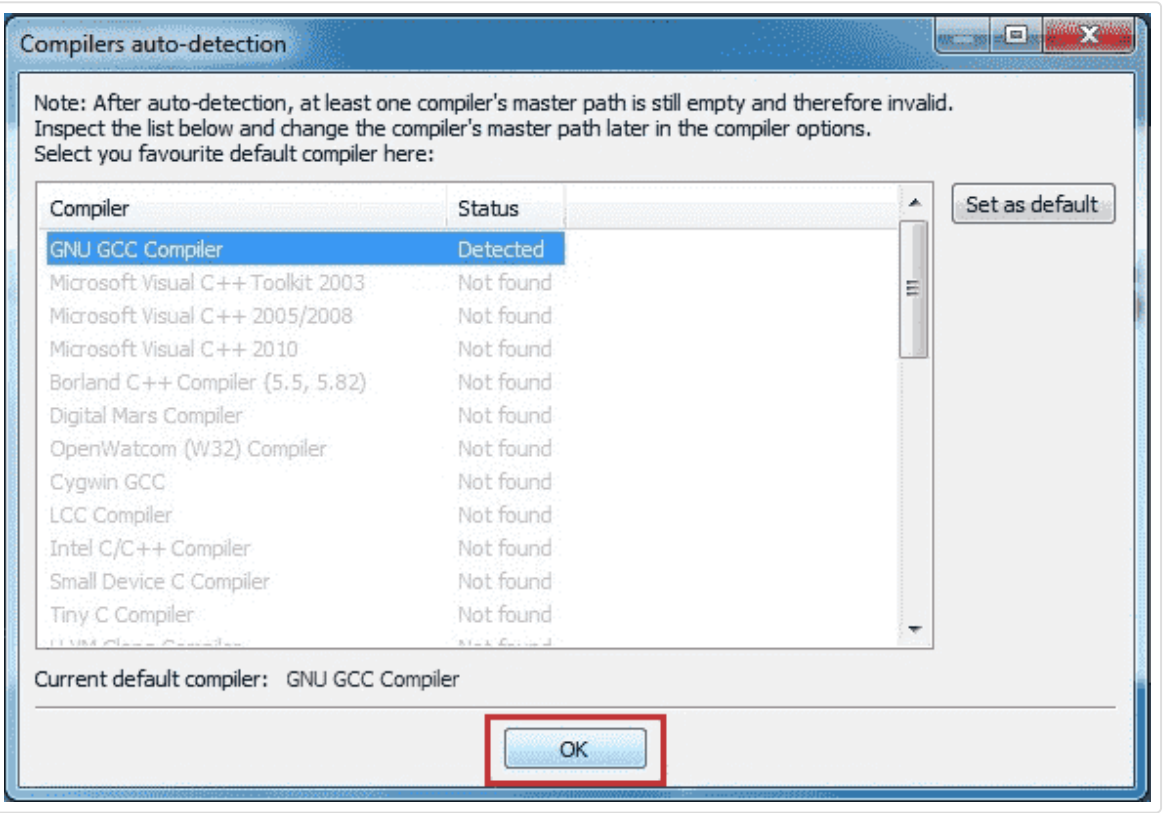
Some Linux installations may be missing dependencies needed to run or compile programs with Code::Blocks.

Debian-based Linux users (such as those on Mint or Ubuntu) may need to install the *build-essential* package. To do so from the terminal command line, type: `sudo apt-get install build-essential`.

Arch Linux users may need to install *base-devel*, which can be done via `pacman -Syu base-devel`.

Users on other Linux variants will need to determine what their equivalent package manager and packages are.

When you launch Code::Blocks for the first time, you may get a *Compilers auto-detection* dialog. If you do, make sure *GNU GCC Compiler* is set as the default compiler and then select the *OK* button.



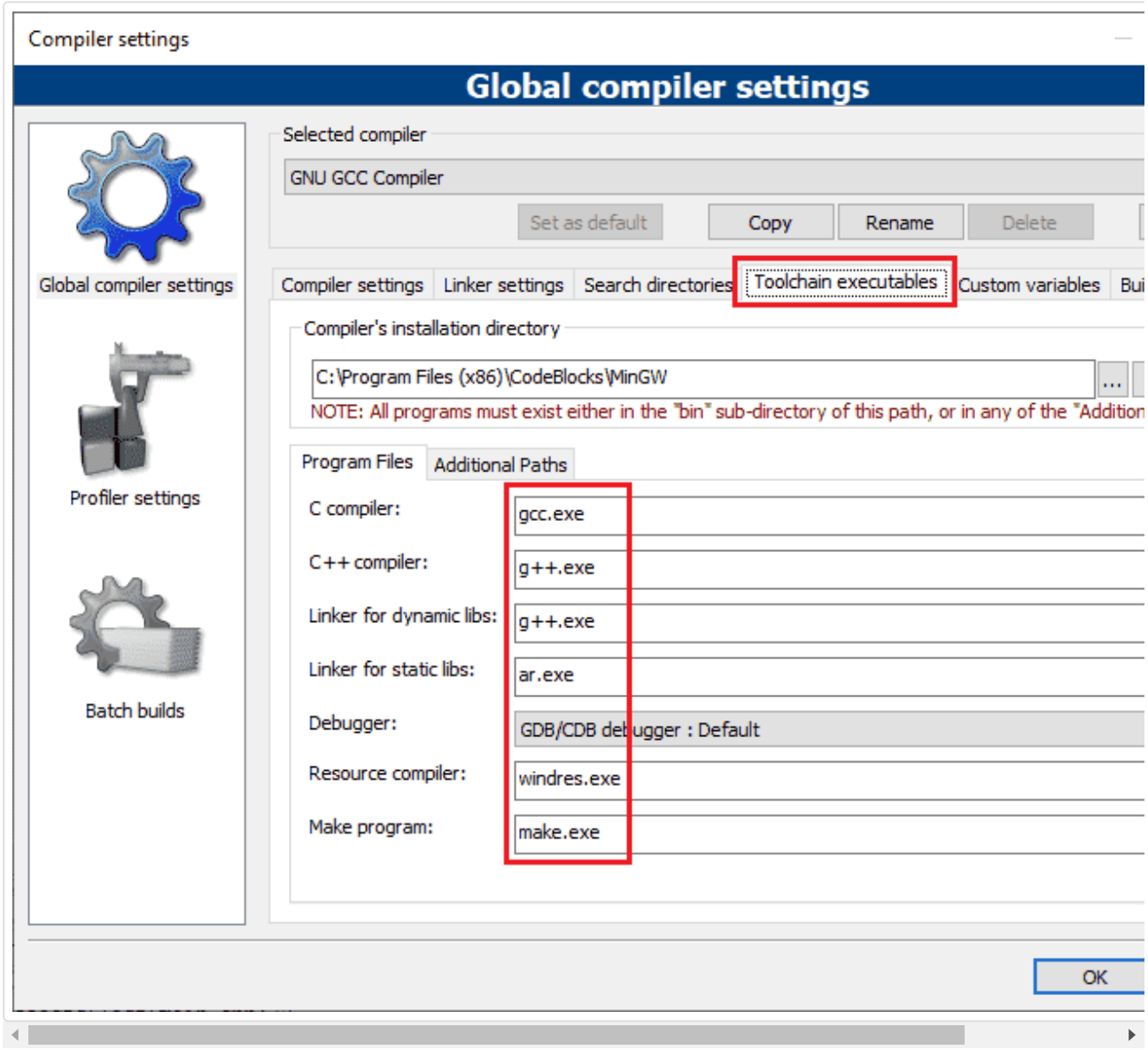
Warning

As of the time of writing, the latest version of Code::Blocks (17.12) doesn't come with a C++17 capable compiler.

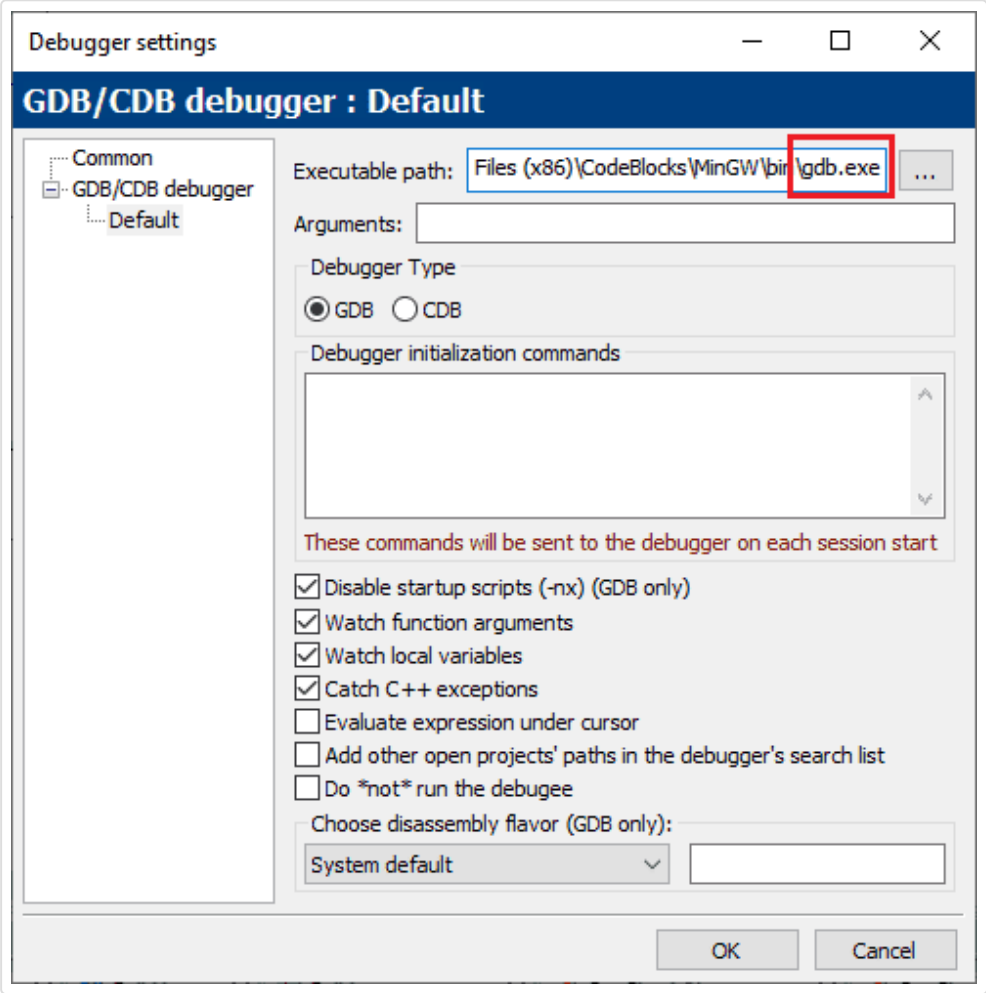
On Linux, you should be able to upgrade gcc via your package manager.

On Windows, you can upgraded the compiler by following these steps:

1. Exit Code::Blocks
2. Go to **https://nuwen.net/mingw.html** and download the latest version (currently mingw-17.1.exe)
3. Extract to C:\
4. Once finished, find your Code::Blocks installation directory (probably C:\Program Files (x86)\CodeBlocks) and delete the old MinGW directory
5. Move the MinGW directory you extracted to C:\ to your Code::blocks installation directory
6. Run Code::Blocks. You'll get a warning about the compiler paths being wrong.
7. Go to Settings -> Compiler -> Toolchain executables, and update the names to "gcc.exe" "g++.exe", "g++.exe", "ar.exe", (skip the debugger), "windres.exe", and "make.exe". Click OK to save.



8. Go to Settings -> Debugger -> GDB/CDB debugger -> Default and rename the executable to “gdb.exe”.



Once you've upgraded your compiler, you'll need to tell Code::Blocks to use C++17 language standard. We cover how to do so in lesson **0.12 -- Configuring your compiler: Choosing a language standard**.

Q: What do I do if I get a "Can't find compiler executable in your configured search paths for GNU GCC Compiler" error?

Try the following:

1. If you're on Windows, make sure you've downloaded the version of Code::Blocks WITH MinGW. It's the one with "mingw" in the name.
2. Try going to settings, compiler, and choose "reset to defaults".
3. Try going to settings, compiler, toolchain executables tab, and make sure "Compiler's installation directory" is set to the MinGW directory (e.g. C:\Program Files (x86)\CodeBlocks\MinGW).
4. Try doing a full uninstall, then reinstall.
5. **Try a different compiler.**

Mac OSX IDEs

Popular Mac choices include **Xcode** (if it is available to you), or **Eclipse**. Eclipse is not set up to use C++ by default, and you will need to install the optional C++ components.

Although Visual Studio for Mac has been released, as of Aug 2018 it does not support C++, so at this time we can not recommend it.

Other compilers or platforms

Q: Can I use a web-based compiler?

Yes, for some things. While your IDE is downloading (or if you're not sure you want to commit to installing one yet), you can continue this tutorial using a web-based compiler, such as the one at **TutorialsPoint**.

Web-based compilers are fine for dabbling and simple exercises. However, they are generally quite limited in functionality -- many won't allow you to save projects, create executables, or effectively debug your programs. You'll want to migrate to a full IDE when you can.

Q: Can I use a command-line compiler (e.g. g++ on Linux)?

Yes, but we don't recommend it for beginners. You'll need to find your own editor and look up how to use it elsewhere. Using a command line debugger is not as easy as an integrated debugger, and will make debugging your programs more difficult.

Q: Can I use other code editors or IDEs, such as Eclipse, Sublime, Notepad++, or Visual Studio Code?

Yes, but we don't recommend it for beginners. There are many great code editors and IDEs that can be configured to support a wide variety of languages, and allow you to mix and match plugins to customize your experience however you like. However, many of these editors and IDEs require additional configuration to compile C++ programs, and there's a lot that can go wrong during that process. For beginners, we recommend something that works out of the box, so you can spend more time learning to code and less time trying to figure out why your code editor isn't working properly with your compiler or debugger.

IDEs to avoid

We recommend avoiding the following IDEs because they are no longer actively supported or maintained:

- Borland Turbo C++
- Dev C++

If possible, use an IDE that receives regular updates and bug fixes, and that will be updated to incorporate new language features over time.

When things go wrong (a.k.a. when IDE stands for "I don't even...")

IDE installations seem to cause their fair share of problems. Installation might fail outright (or installation might work but the IDE will have problems when you try to use it due to a configuration issue). If you encounter such issues, try uninstalling the IDE (if it installed in the first place), reboot your machine, disable your antivirus or anti-malware temporarily, and try the installation again.

If you're still encountering issues at this point, you have two options. The easier option is to try a different IDE. The other option is to fix the problem. Unfortunately, the causes of installation and configuration errors are varied and specific to the IDE software itself, and we're unable to effectively advise on how to resolve such issues. In this case, we recommend copying the error message or problem you are having into a Google search and trying to find a forum post elsewhere from some poor soul who has inevitably encountered the same issue. Often there will be suggestions on things you can try to remedy the issue.

Moving on

Once your IDE is installed (which can be one of the hardest steps if things don't go as expected), or if you're temporarily proceeding with a web-based compiler, you are ready to write your first program!

0.7 -- Compiling your first program

Index

0.5 -- Introduction to the compiler, linker, and libraries

C++ TUTORIAL | PRINT THIS POST

525 comments to 0.6 — Installing an Integrated Development Environment (IDE)

« Older Comments 1 ... 7 8 9

Vitaliy Sh.
December 20, 2019 at 1:54 am · Reply

...
Eclipse
...

Please change to "https", as it redirects there anyway.

nascardriver
December 20, 2019 at 2:09 am · Reply

Updated, thanks! Code::Blocks is still http-only, oof.

Vitaliy Sh.
December 20, 2019 at 2:13 am · Reply

Sir!
They do have:
https://wiki.codeblocks.org/
https://wiki.codeblocks.org/downloads/26

But i'm unaware of https for
http://wiki.codeblocks.org/index.php/Installing_a_supported_compiler link...

nascardriver

December 20, 2019 at 2:21 am · Reply

I see, they moved the downloads to the wiki subdomain. I updated that link, but there doesn't seem to be a real wiki on the https wiki subdomain.

Vitaliy Sh.
December 20, 2019 at 2:32 am · Reply

Shh.. Conspiration! It hidden in http XD...

Vitaliy Sh.
December 20, 2019 at 1:06 am · Reply

...
>Try a different compiler.

...

Possible typo: '.' before .

Vitaliy Sh.
December 20, 2019 at 12:41 am · Reply

"Q: I'm getting a "Can't find compiler executable in your configured search paths for GNU GCC Compiler" error"

What about to:
Insert a "What to do if " before "I'm", then add a '?' at the end of sentence, and replace ' ' with
 after "getting a" (for a better contrast)?

Vitaliy Sh.
November 15, 2019 at 12:57 am · Reply

...
Note that Code::Blocks may not enable C++11 capabilities by default.

...

Maybe change: "C++11 (or newer) capabilities"

Because:
1) "Some lessons additionally cover topics introduced in C++14/17/20." in here.
2) "Check the one with the highest number" in 0.12.

Alex
November 15, 2019 at 4:17 pm · Reply

Amended, thanks!

Vitaliy Sh.
November 11, 2019 at 5:37 am · Reply

... Some linux installations may be missing ...
Minor typo: l. case "l" in "Linux".

Maybe a typo:
... Arch-linux users may need ...
As they possibly prefer:
(Arch Linux | Arch | archlinux)

https://wiki.archlinux.org/index.php/Arch_terminology#Arch_Linux

Vitaliy Sh.
November 10, 2019 at 9:08 am · Reply

...Q: I'm getting a "Can't find compiler executable in your configured search paths for GNU GCC Compiler" error...

...
5. Try a different compiler.
...

Maybe add the:
https://en.wikipedia.org/wiki/Code::Blocks#Compilers
or (less likely)
https://en.wikipedia.org/wiki/List_of_compilers#C++_compilers

as a link to word compiler?

Vitaliy Sh.
November 10, 2019 at 8:03 am · Reply

...In you're on Windows, make sure you've downloaded...

Typo: like to be "If" instead of "In".

Vitaliy Sh.
November 10, 2019 at 5:31 am · Reply

...If so, this will be labeled Have g++ follow the coming C++11YY (aka C++XX) ISO C++ language standard [-std=c++ZZ]...

Maybe better C++XX (aka C++XX)? That will better match the diagram above.

Vitaliy Sh.
November 10, 2019 at 5:09 am · Reply

...may need to install the build-essential package From...
Typo: no period between "package" and "From".

nascardriver
November 10, 2019 at 5:15 am · Reply

Lesson updated, thanks for your consecutive feedback!

Zwaffel
November 8, 2019 at 12:03 am · Reply

Hey i just upgraded from version Visual studio 2017 to visual studio 2019 hoping to get a more recent c++ compiler.
But seems like im still stuck on c++11 since i cant use string_view etc.

How can i upgrade to a more recent compiler in visual studio. I have been researching on the web but i cant really find any good info about it. Or the method's they show dont work for me.

nascardriver
November 9, 2019 at 2:51 am · Reply

You need to select a higher standard in the project settings
https://stackoverflow.com/a/46759740

Alex
November 11, 2019 at 3:57 pm · Reply

I added a section called "Enabling C++17 or C++20 language features in Visual Studio" to the next lesson discussing how to enable C++17 or C++20 capability in Visual Studio.

Dexter
October 5, 2019 at 2:28 am · Reply

I already have a dev c++ and the program "Hello world"works, Why do I still need to download a compiler like visual studio or code blocks

nascardriver
October 6, 2019 at 2:37 am · Reply

Dev c++ is using an outdated compiler. You can't use a lot of the features added in the past 5 years.
Visual Studio and Code::Blocks are IDEs. Visual Studio ships with a good compiler and Code::Blocks lets you choose which compiler to use, so you can always use the latest features.

johannes
September 30, 2019 at 1:34 am · Reply

i have a dev-c++ is this also an IDE?

Alex
September 30, 2019 at 9:15 am · Reply

Yes, but I recommend you use another IDE as dev-c++ isn't currently an active project.

Sean
September 18, 2019 at 3:31 pm · Reply

I've downloaded Visual Studio "Code" on Manjaro Linux, what extensions should I use and which C++ standard etc. should I have the app set to? Would be appreciated as I don't see much about it in this article. Just wanna have it set up right so I don't worry later. All help is appreciated. (Currently have the C/C++ extension and standards set to latest)

nascardriver
September 19, 2019 at 12:19 am · Reply

C/C++
CMake (You should use a build system, CMake makes compilation easier (Once you set it up right))
Code Spell Checker
TODO Highlight

Those should give you a good starting point. I don't know what "C_Cpp.default.cppStandard" does, I never touched it. What matters is the standard version you set in your compiler options (CMake). Use the latest (C++20 or C++2a (Or C++17 if you don't want to use an unfinished standard)).

Sean
September 19, 2019 at 3:51 am · Reply

Thank you :)

Badreddine Boukheit
August 24, 2019 at 1:51 am · Reply

I would appreciate it if you taught us how to compile programs in command line.

nascardriver
August 24, 2019 at 2:23 am · Reply

Install a compiler, eg. clang++. You'll have to search in your package manager to find the most recent version, or download the compiler right from the developers.

Then it's as easy as

```
// to compile
clang++ ./main.cpp

// to run
./a.out
```

The upcoming lessons include compiler setup instructions for gcc, which can also be used for clang++.

oswaldo
August 15, 2019 at 1:08 pm · Reply

Dev-C++ ?

Monolog
August 6, 2019 at 10:53 pm · Reply

I´m moving on through these lessons and now I have found some time to thank the autor for his excellent job. I´m just downloading Visual Studio. Also I´m continously learning English by these.
Thank you one more time from the Czech Republic.

Dave Blowe
June 25, 2019 at 9:47 am · Reply

When you are a beginner, it is best to use the tools that your mentor uses. She/he will be able to guide that use.

I started with C using the Turbo C compiler from Borland. It came with a simple IDE which is about the same as Geany today. Geany is a cross-platform editor, with just a little more added to help with compiled languages. This is an MS-DOS program. $20.

After a few years doing C, my job plans required that I learn C++. By that time, Borland had made Turbo C++, which was the same as TurboC, just with a C++ compiler and debugger. This is an MS-DOS program. $20.

At work, we were using the Borland C++ environment to create DOS and Windows programs. $290. For every other platform, we supported 12 in total, we used gcc and g++ for development. Software development on Unix/Linux is very different because the entire OS is a development environment. There isn't any need for an IDE.

What is an IDE? It is:
* editor
* compiler
* linker
* debugger
Well, Unix OSes have all that covered so you don't have to settle for a bad editor (like eclipse or Visual-whatever) if you don't prefer those editors.

Pick your own:
* editor - vim, emacs, geany, Atom, whatever.
* compiler - g++, gcc, cc, CC, whatever your company says/requires.
* linker - usually the same as the compiler, just run in a different _mode_
* debugger - gdb is the normal debugger for any of the GNU compiled languages. Just be certain that your compiler and linker options include "symbols" or "debug symbols" until you go to production. Then you don't want symbols included an usually want to compile and link at higher optimization levels.

Whichever platform you develop on, it will be tempting to use all sorts of native platform libraries. Resist doing this, since it will make porting the code that much more difficult on other platforms.

If you want to develop using a Chromebook, you'll need to setup a normal Linux environment first, probably using Crouton or the fork of crouton. At that point, follow the Linux instructions.

I always found chromeOS to be too limiting and ended up wiping the OS, breaking the hardware warranty, so I could boot directly into a normal Linux OS, like Ubuntu-Mate. A typical Chromebook needs to have at least 4GB of RAM and probably 32G of storage to be useful for non-trivial C/C++ development.

No chromebooks are sufficient to do Java development. Java is a hog. It has been since SunMicro first introduced it in the mid-1990s. They made all sorts of promises about speed, optimizations, lower memory use - none of those have happened 25 yrs later. Java is still a hog.

JTW
November 2, 2019 at 1:30 am · Reply

"No chromebooks are sufficient to do Java development. Java is a hog. It has been since SunMicro first introduced it in the mid-1990s. They made all sorts of promises about speed, optimizations, lower memory use - none of those have happened 25 yrs later. Java is still a hog."

The misinformation is strong in this one.
1) Java was NOT a SunMicro product
2) Java is NOT slow, it's NOT a hog. Yes, the JVM requires memory to run, but runtime performance of a Java program can rival anything else out there including C++ (I should know, I've been writing Java software professionally for over 2 decades and have just last month delivered a Java application that ran TOO FAST for the requirements which had been written assuming a C implementation and had to be artificially slowed down to allow the required polling speed to be achieved as it was processing far faster than called for. We did nothing special to make this performance happen, just proper coding practices and knowing your tools

3) If you can't use a Chromebook to code Java, it's most likely because there is no proper JVM for them (I've never bothered to check), not because they lack the hardware.

" a bad editor (like eclipse or Visual-whatever)"

More utter misinformation. The current versions of both Eclipse and Visual Studio are pretty good. Personally I don't like them, but they're high quality products.
On the other hand, Atom which you propose is a memory hog, runs exceedingly slowly even on a high end machine. Weird that you'd advise it if you're so adamant that something far faster is too slow...

Skazok
June 15, 2019 at 5:46 am · Reply

It seams that to have Code::Blocks on Ubuntu you have compile it. There is no ready installation package for Ubuntu. If you want to compile some thing just compile soft for compiling before :) I heard that programmers like recursion but I am not ready for this things yet.

So I decide to use Eclipse it is easy to install and it works.

Krishna
June 3, 2019 at 4:08 am · Reply

My computer has "intel c/c++ compiler" as default and also there is no option for "GNU GCC compiler" in the list.
while i have installed "code::block mignw setup.exe".

I mean all these are not working, not compiling my program.

person
June 2, 2019 at 3:56 am · Reply

what IDE should i use for chrome os?
are even there any good IDEs for chrome os?

Mabel
May 10, 2019 at 11:02 pm · Reply

I've just finished downloading visual studio and a page came up asking me to choose which development setting to use. - General

- JavaScript
- Visual Basics
- Visual C#
- Visual C++
- Web development
- Web Development (code only)

Do I chose Visual C++? orrrrrrr
And which color theme is commonly used? Blue, Dark or Light?

nascardriver
May 11, 2019 at 4:36 am · Reply

Visual C++, Dark

Mabel
May 13, 2019 at 4:07 am · Reply

Thank you

« Older Comments 1 ... 7 8 9