# Coding with C++

C++ is an amazing programming language. Most of what you see in front of you when you power up your computer, regardless of whether you're using Windows, macOS or Linux, is created using C++. Being able to code in C++ opens a whole new world for you, in terms of desirable professional skills and the ability to code amazing apps and games.

C++ is an efficient and powerful language that's used to develop operating systems, applications, games and much more. It's used in science, engineering, banking, education, and the space industry to name but a few.

This section looks at the necessary equipment and software needed to get up and running with C++ for Windows, Mac and Linux systems.

# Why C++?

C++ is one of the most popular programming languages available today. Originally called C with Classes, the language was renamed C++ in 1983. It's an extension of the original C language and is a general purpose object-oriented (OOP) environment.

## C EVERYTHING

**Due to how complex the language can be, and its power and performance, C++ is often used to develop games, programs, device drivers and even entire operating systems.**

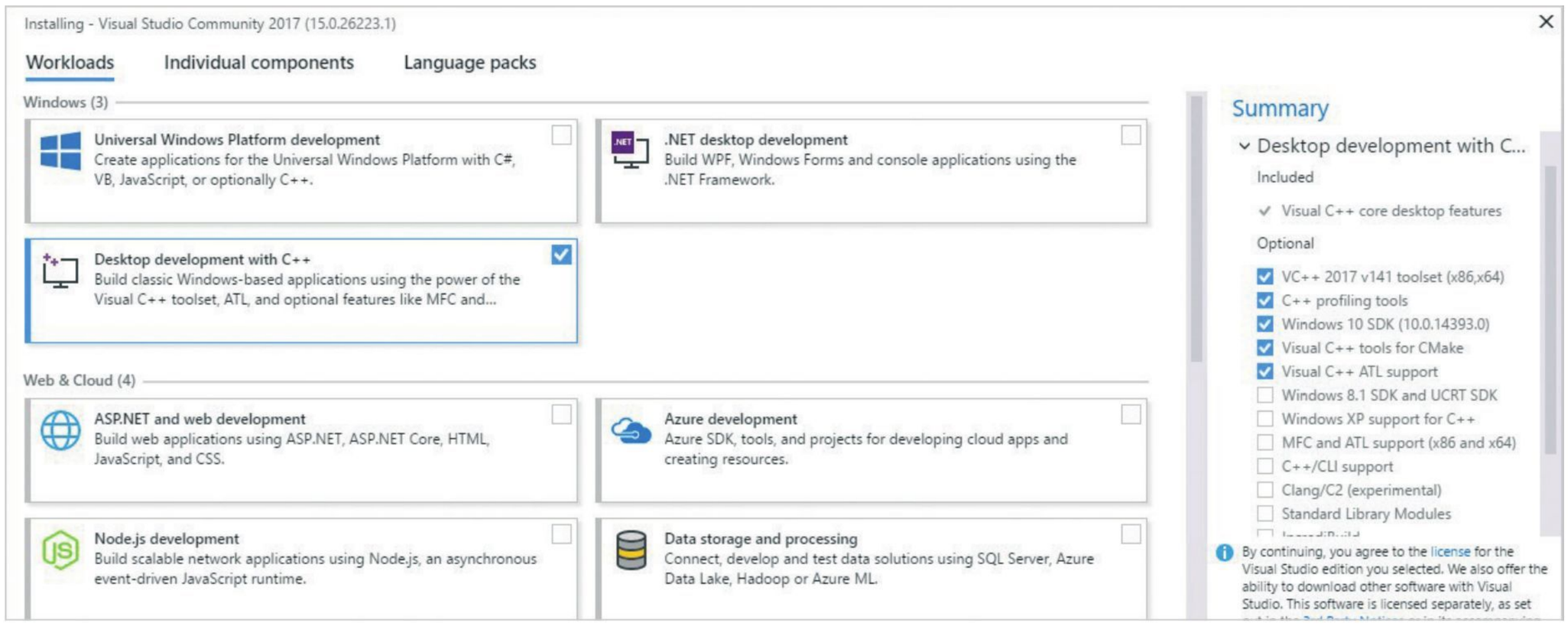
Dating back to 1979, the start of the golden era of home computing, C++, or rather C with Classes, was the brainchild of Danish computer scientist Bjarne Stroustrup while working on his PhD thesis. Stroustrup's plan was to further the original C language, which was widely used since the early seventies.

C++ proved to be popular among the developers of the '80s, since it was a much easier environment to get to grips with and more importantly, it was 99% compatible with the original C language. This meant that it could be used beyond the mainstream computing labs and by regular people who didn't have access to the mainframes and large computing data centres.

C++'s impact in the digital world is immense. Many of the programs, applications, games and even operating systems are coded using C++. For example, all of Adobe's major applications, such as Photoshop, InDesign and so on, are developed in C++. You will find that the browser you surf the Internet with is written in C++, as well as Windows 10, Microsoft Office and the backbone to Google's search engine. Apple's macOS is written largely in C++ (with some

**C++ code is much faster than that of Python.**

```
#include<iostream>
using namespace std;
void main()
{char ch;
 cout<<"Enter a charater to check it is vowel or not";
 cin>>ch;
      switch(ch)
      {
       case'a': case'A':
       cout<<ch<<" is a Vowel";
       break;
           case 'e': case'E':
           cout<<ch<<" is a Vowel";
           break;
               case 'i': case'I':
               cout<<ch<<" is a Vowel";
               break;
                   case 'o': case'O':
                   cout<<ch<<" is a Vowel";
                   break;
                       case 'u': case'U':
                       cout<<ch<<" is a Vowel";
```




**Microsoft's Visual Studio is a great, free environment to learn C++ in.**

other languages mixed in depending on the function) and the likes of NASA, SpaceX and even CERN use C++ for various applications, programs, controls and umpteen other computing tasks.

C++ is also extremely efficient and performs well across the board as well as being an easier addition to the core C language. This higher level of performance over other languages, such as Python, BASIC and such, makes it an ideal development environment for modern computing, hence the aforementioned companies using it so widely.

While Python is a great programming language to learn, C++ puts the developer in a much wider world of coding. By mastering C++, you canfind yourself developing code for the likes of Microsoft, Apple and so on. Generally, C++ developers enjoy a higher salary than programmers of some other languages and due to its versatility, the C++ programmer can move between jobs and companies without the need to relearn anything specific. However, Python is an easier language to begin with. If you're completely new to programming then we would recommend you begin with Python and spend some time getting to grips with programming structure and the many ways and means in which you find a solution to a problem through programming. Once you can happily power up your computer and whip out a Python program with one hand tied behind your back, then move on to C++. Of course, there's nothing stopping you from jumping straight into C++; if you feel up to the task, go for it.

Getting to use C++ is as easy as Python, all you need is the right set of tools in which to communicate with the computer in C++ and you can start your journey. A C++ IDE is free of charge, even the immensely powerful Visual Studio from Microsoft is freely available to download and use. You can get into C++ from any operating system, be it macOS, Linux, Windows or even mobile platforms.

Just like Python, to answer the question of Why C++ is the answer is because it's fast, efficient and developed by most of the applications you regularly use. It's cutting edge and a fantastic language to master.

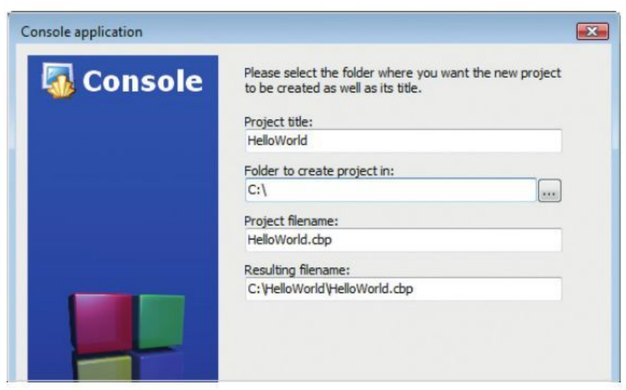

**Indeed, the operating system you're using is written in C++.**

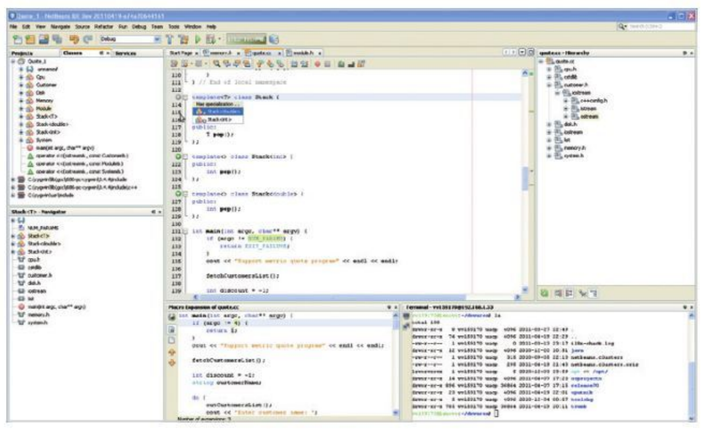
# Equipment You Will Need

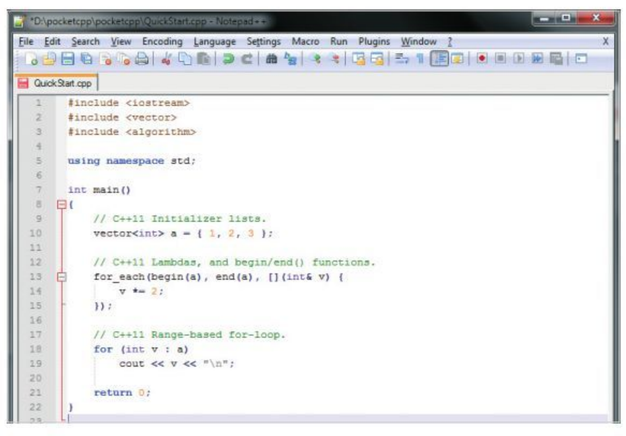
You don't need to invest a huge amount of money in order to learn C++, and you don't need an entire computing lab at your disposal either. Providing you have a fairly modern computer, everything else is freely available.

## C++ SETUPS

**As most, if not all, operating systems have C++ at their core, it stands to reason that you can learn to program in C++ no matter what OS you're currently using.**






### COMPUTER

Unless you fancy writing out your C++ code by hand on a sheet of paper (which is something many older coders used to do), then a computer is an absolute must have component. PC users can have any recent Linux distro or Windows OS, Mac users the latest macOS.

### AN IDE

As with Python, an IDE is used to enter and execute your C++ code. Many IDEs come with extensions and plugins that help make it work better, or add an extra level of functionality. Often, an IDE will provide enhancements depending on the core OS being used, such as being enhanced for Windows 10.

### COMPILER

A compiler is a program that will convert the C++ language into binary that the computer can understand. While some IDEs come with a compiler built in, others don't. Code::Blocks is our favourite IDE that comes with a C++ compiler as part of the package. More on this later.

### TEXT EDITOR

Some programmers much prefer to use a text editor to assemble their C++ code before running it through a compiler. Essentially you can any text editor to write code, just save it with a .cpp extension. However, Notepad++ is one of the best code text editors available.

### INTERNET ACCESS

While it's entirely possible to learn how to code on a computer that's not attached to the Internet, it's extraordinarily difficult. You will need to install the relevant software, keep it up to date, install any extras or extensions, and look for help when coding. All of which require access to the Internet.

### TIME AND PATIENCE

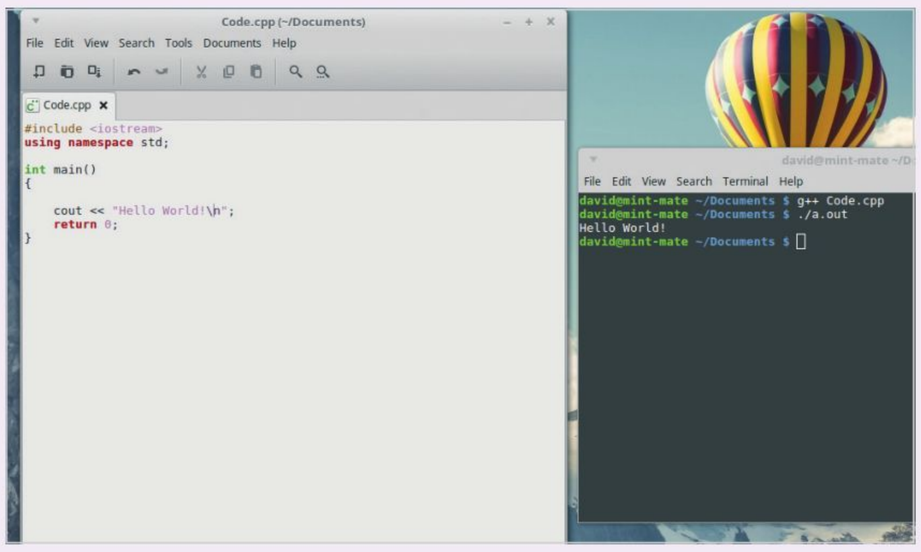
Yes, as with Python, you're going to need to set aside significant time to spend on learning how to code in C++. Sadly, unless you're a genius, it's not going to happen overnight, or even a week. A good C++ coder has spent many years honing their craft, so be patient, start small and keep learning.

# OS SPECIFIC NEEDS

**C++ will work in any operating system, however, getting all the necessary pieces together can be confusing to a newcomer. Here's some OS specifics for C++.**

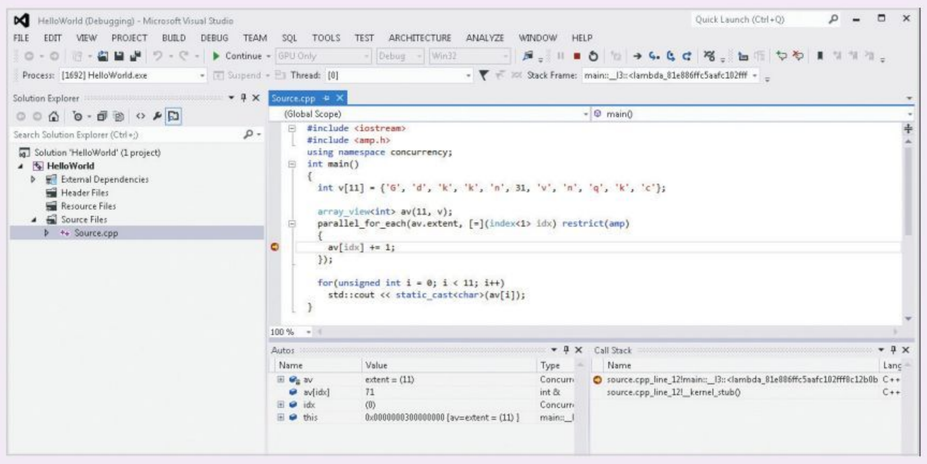
## LINUX

Linux users are lucky in that they already have a compiler and text editor built into their operating system. Any text editor will allow you type out your C++ code, when it's saved with a .cpp extension, use g++ to compile it.



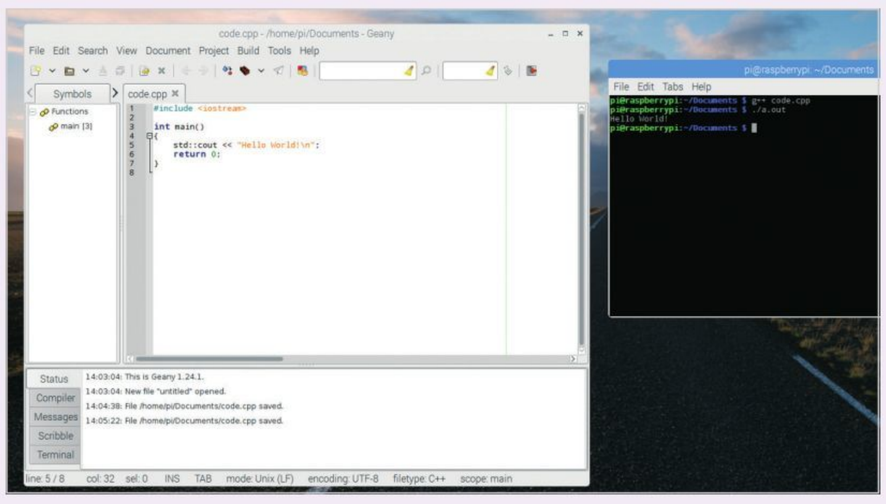

## WINDOWS

As we've mentioned previously, a good IDE is Microsoft's Visual Studio. However, a better IDE and compiler is Code::Blocks, which is regularly kept up to date with a new release twice a year, or so. Otherwise Windows users can enter their code in Notepad++ then compile it with MinGW – which Code::Blocks uses.



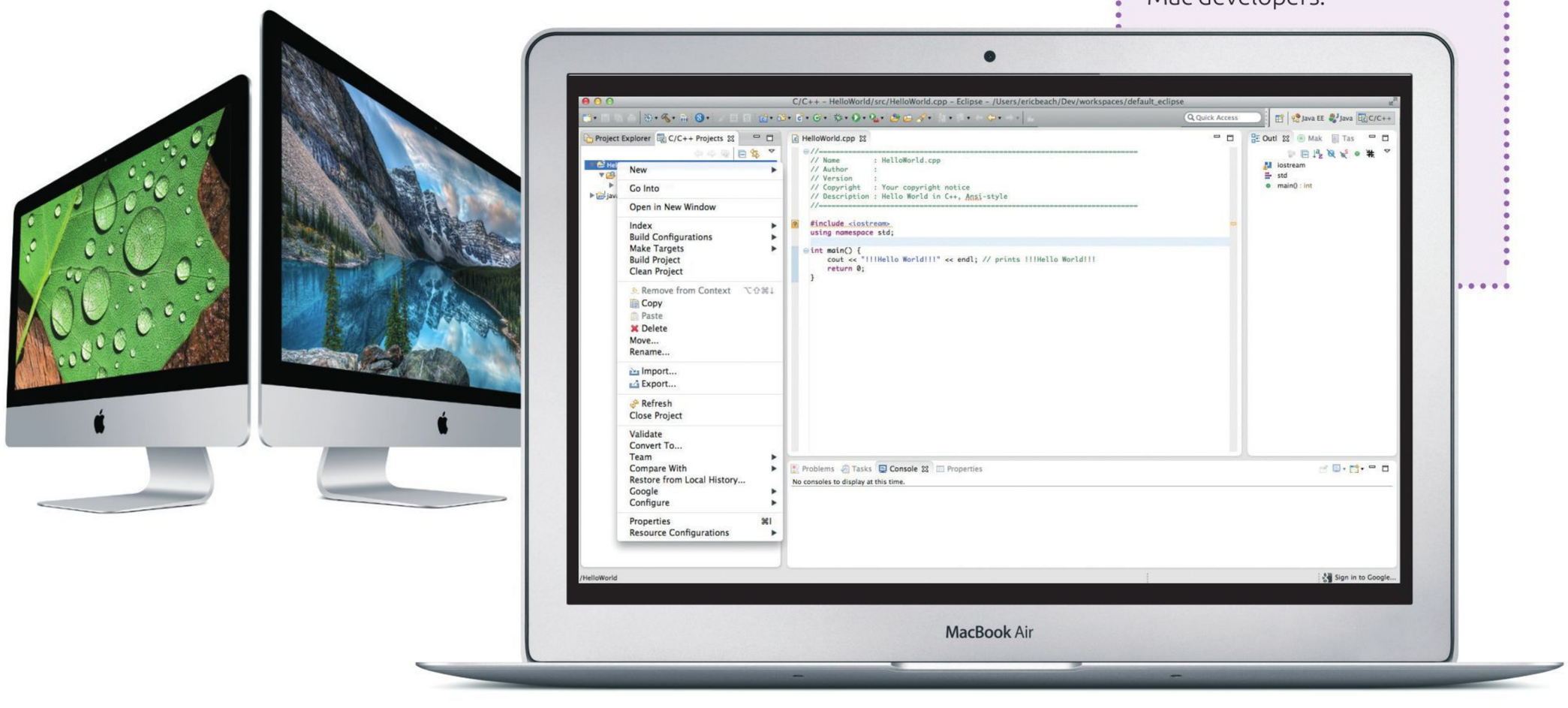

## RASPBERRY PI

The Raspberry Pi's operating system is Raspbian, which is Linux based. Therefore, you're able to write your code out using a text editor, then compile it with g++ as you would in any other Linux distro.




## MAC

Mac owners will need to download and install Xcode to be able to compile their C++ code natively. Other options for the macOS include Netbeans, Eclipse or Code::Blocks. Note: the latest Code::Blocks isn't available for Mac due to a lack of Mac developers.



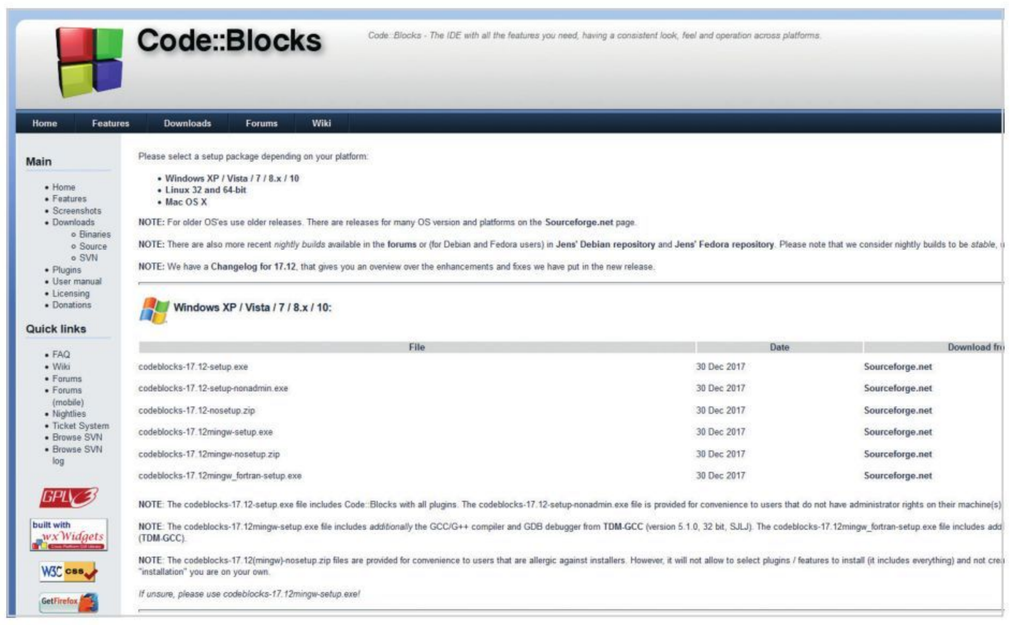
# How to Set Up C++ in Windows

Windows users have a wealth of choice when it comes to programming in C++. There are loads of IDEs and compilers available, including Visual Studio from Microsoft. However, in our opinion, the best C++ IDE to begin with is Code::Blocks.

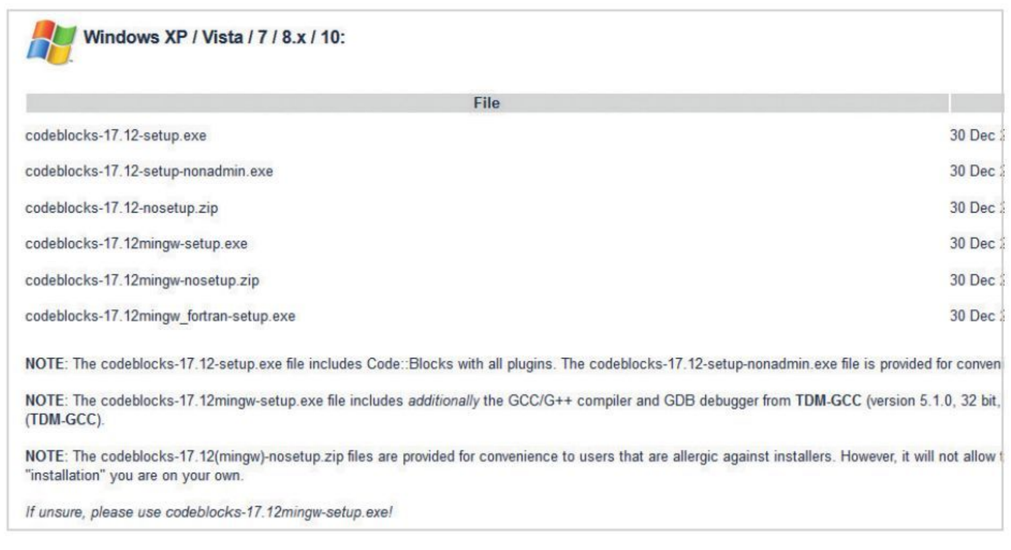
## CODE::BLOCKS

**Code::Blocks is a free C++, C and Fortran IDE that is feature rich and easily extendible with plugins. It's easy to use, comes with a compiler and has a vibrant community behind it too.**

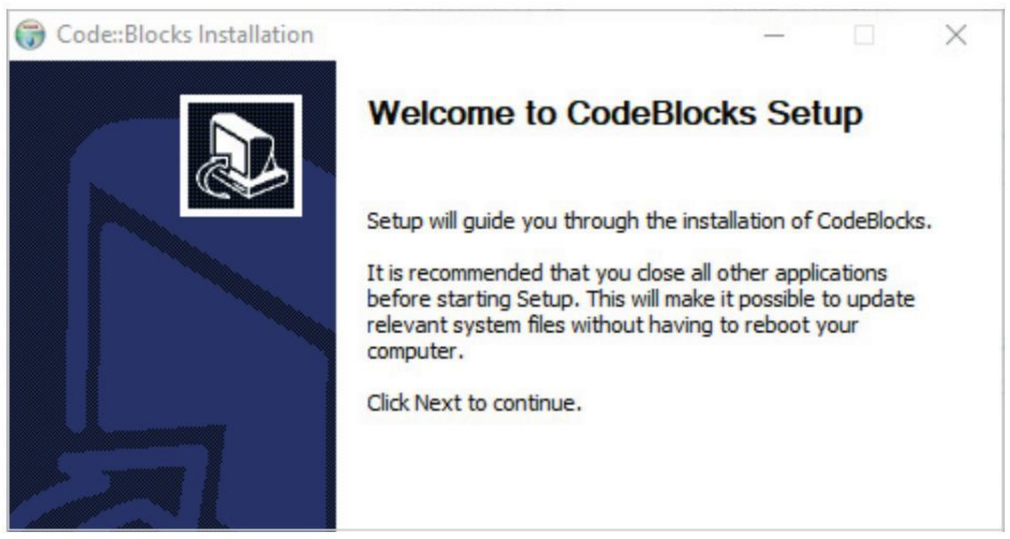
**STEP 1** Start by visiting the Code::Blocks download site, at www.codeblocks.org/downloads. From there, click on the 'Download the binary releases' link to be taken to the latest downloadable version for Windows.




**STEP 2** There you can see, there are several Windows versions available. The one you want to download has mingw-setup.exe at the end of the current version number. At the time of writing this is: codeblocks-17.12mingw-setup.exe. The difference is that the mingw-setup version includes a C++ compiler and debugger from TDM-GCC (a compiler suite).

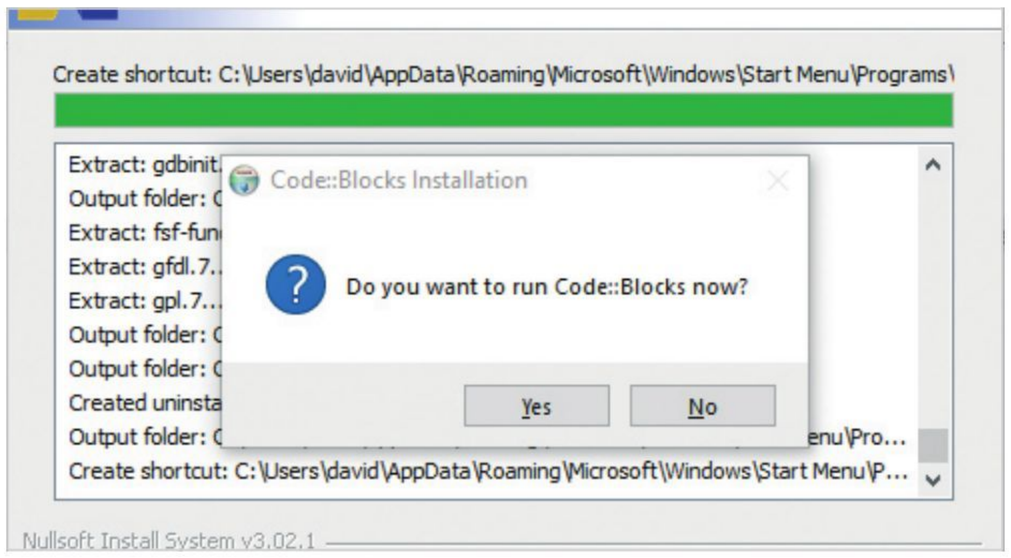



**STEP 3** When you've located the file, click on the Sourceforge.net link at the end of the line and a download notification window appears; click on Save File to start the download and save the executable to your PC. Locate the downloaded Code::Blocks installer and double-click to start. Follow the on-screen instructions to begin the installation.

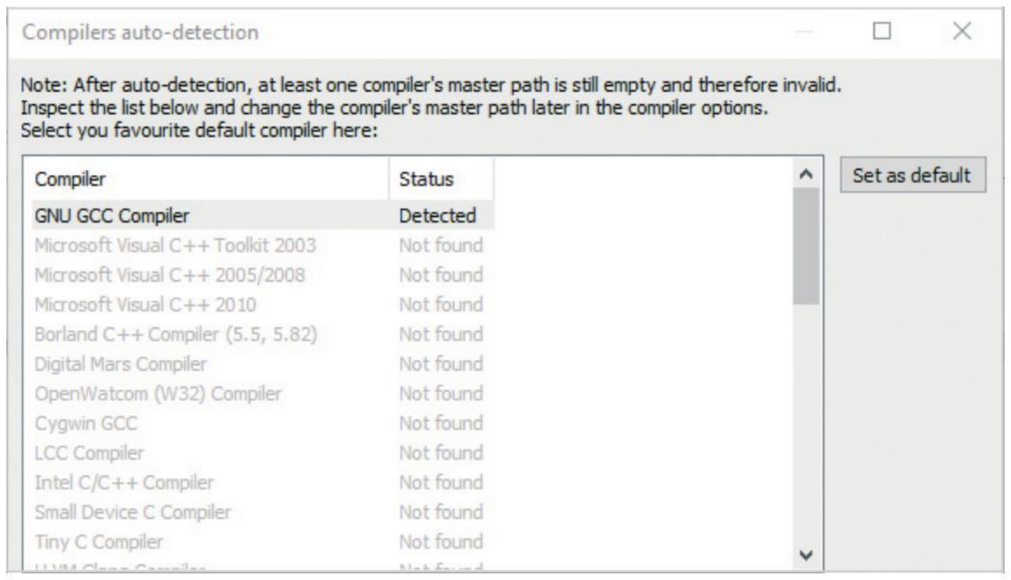



**STEP 4** Once you agree to the licencing terms, a choice of installation options becomes available. You can opt for a smaller install, missing out on some of the components but we recommend that you opt for the Full option, as default.

**STEP 5** Next choose an install location for the Code::Blocks files. It's your choice but the default is generally sufficient (unless you have any special requirements of course). When you click Next, the install begins; when it's finished a notification pops up asking you if you want start Code::Blocks now, so click Yes.

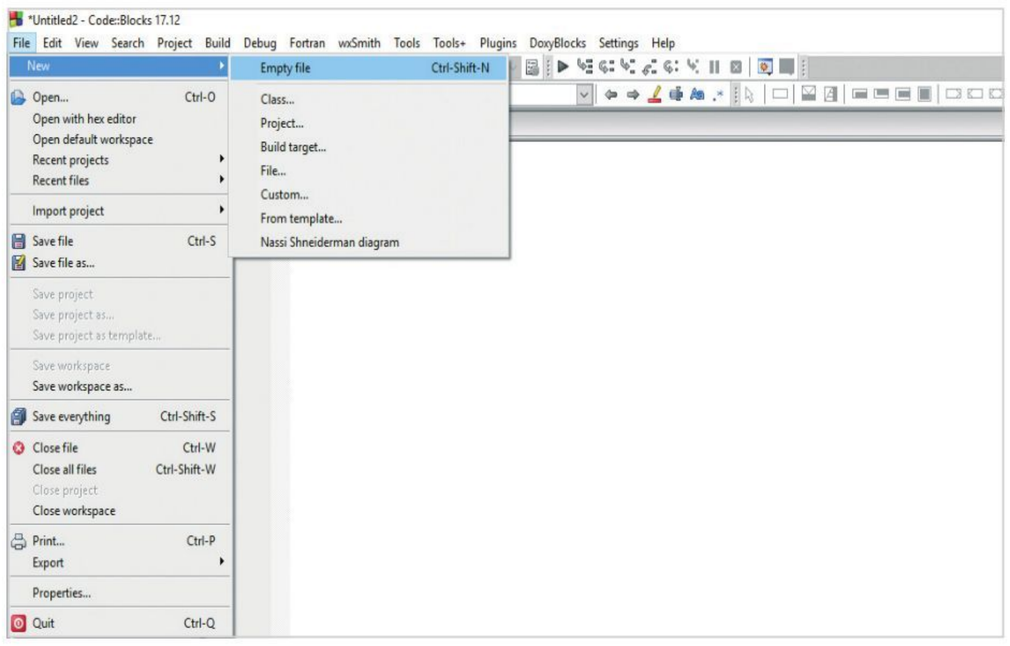



**STEP 6** The first time Code::Blocks loads it runs an autodetect for any C++ compilers you may already have installed on your system. If you don't have any, click on the first detected option: GNU GCC Compiler and click the Default button to set it as the system's C++ compiler. Click OK when you're ready to continue.




**STEP 7** The program starts and another message appears informing you that Code::Blocks is currently not the default application for C++ files. You have two options, to leave everything as it is or allow Code::Blocks to associate all C++ file types. Again, we would recommend you opt for the last choice, to associate Code::Blocks with every supported file type.

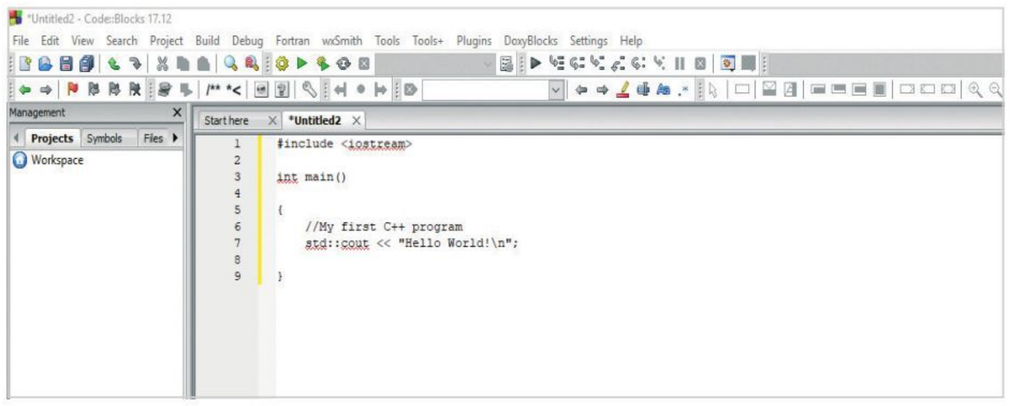

**STEP 8** There's a lot you can do in Code::Blocks, so you need to dig in and find a good C++ tutorial to help you get the most from it. However, to begin with, click on File > New > Empty File. This creates a new, blank window for you to type in.



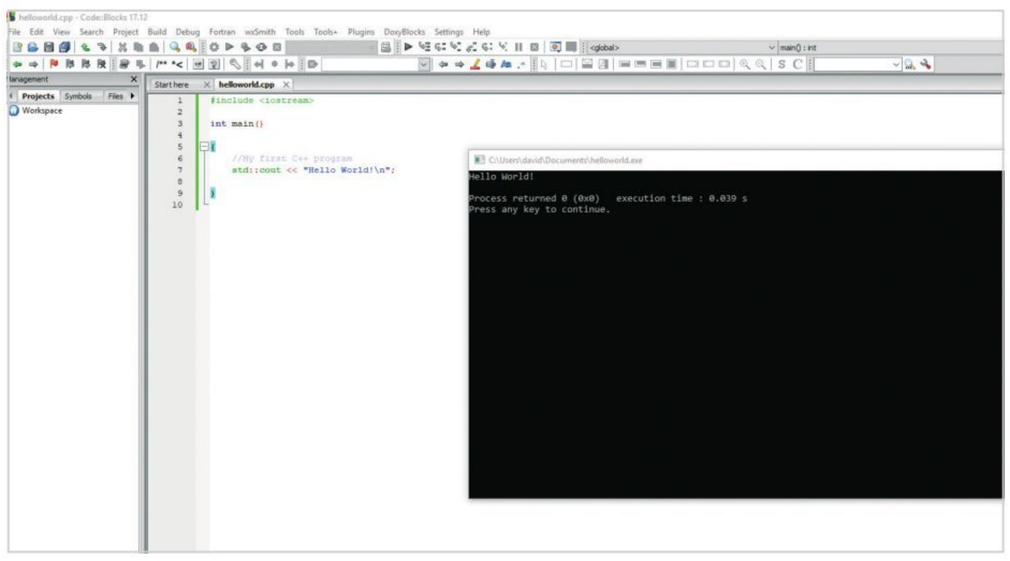

**STEP 9** In the new window, enter the following:

```
#include <iostream>

Int main()
{
//My first C++ program
Std::cout << "Hello World!\n";
}
```

Notice how Code::Blocks auto-inserts the braces and speech quotes.




**STEP 10** Click File > Save as and save the code with a .cpp extension (helloworld.cpp, for example). Code::Blocks changes the view to colour code according to C++ standards. To execute the code, click on the Build and Run icon along the top of the screen. It's a green play icon together with a yellow cog.



# How to Set Up C++ on a Mac

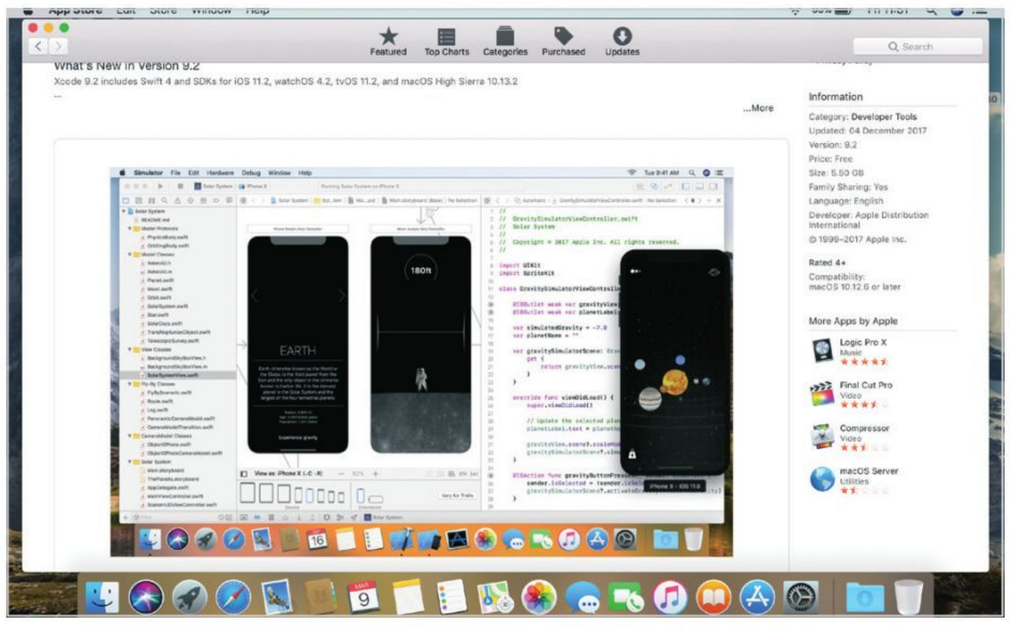
To start C++ coding on a Mac you need to install Apple's Xcode. This is a free, full featured IDE that's designed to create native Apple apps. However, it can also be used to create C++ code relatively easily.

## XCODE

**Apple's Xcode is primarily designed for users to develop apps for macOS, iOS, tvOS and watchOS applications in Swift or Objective-C, but you can use it for C++ too.**

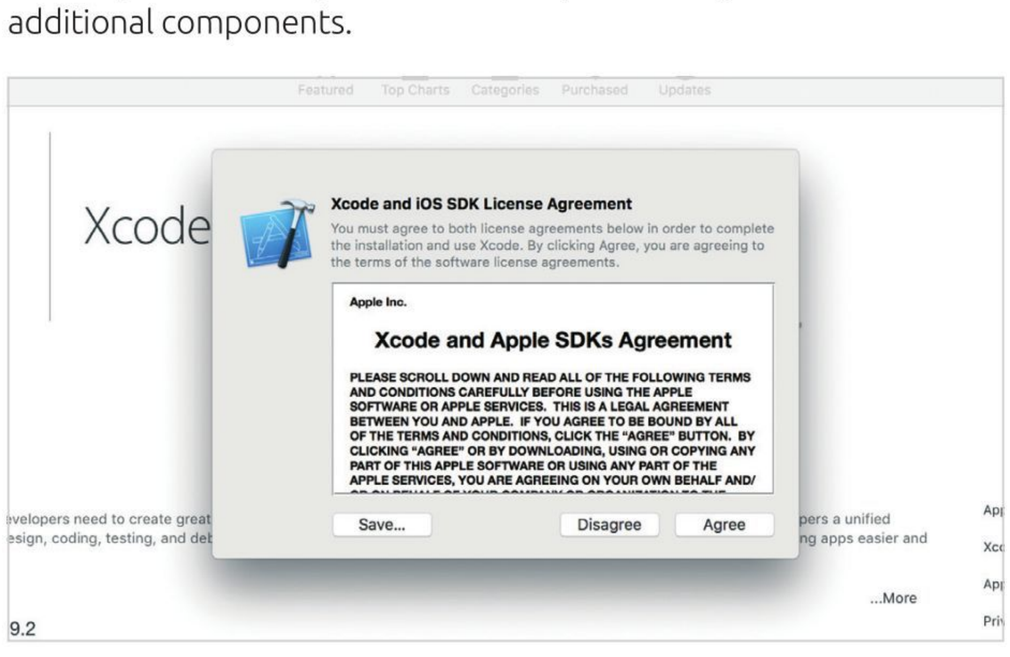
**STEP 1** Start by opening the App Store on your Mac, Apple Menu > App Store. In the Search box enter Xcode and press Return. There will be many suggestions filling the App Store window but it's the first option, Xcode, that you need to click on.


**STEP 2** Take a moment to browse through the app's information, including the compatibility to ensure you have the correct version of macOS. Xcode requires macOS 10.12.6 or later to install and work.

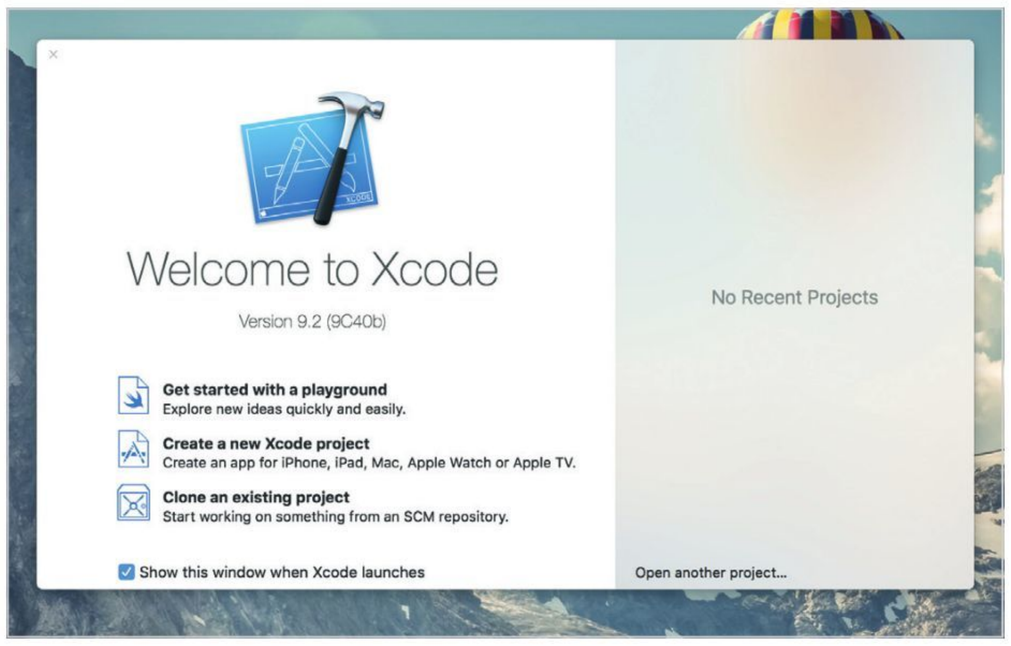



**STEP 3** When you're ready, click on the Get button which then turns into 'Install App'. Enter your Apple ID and Xcode begins to download and install. It may take some time depending on the speed of your Internet connection.

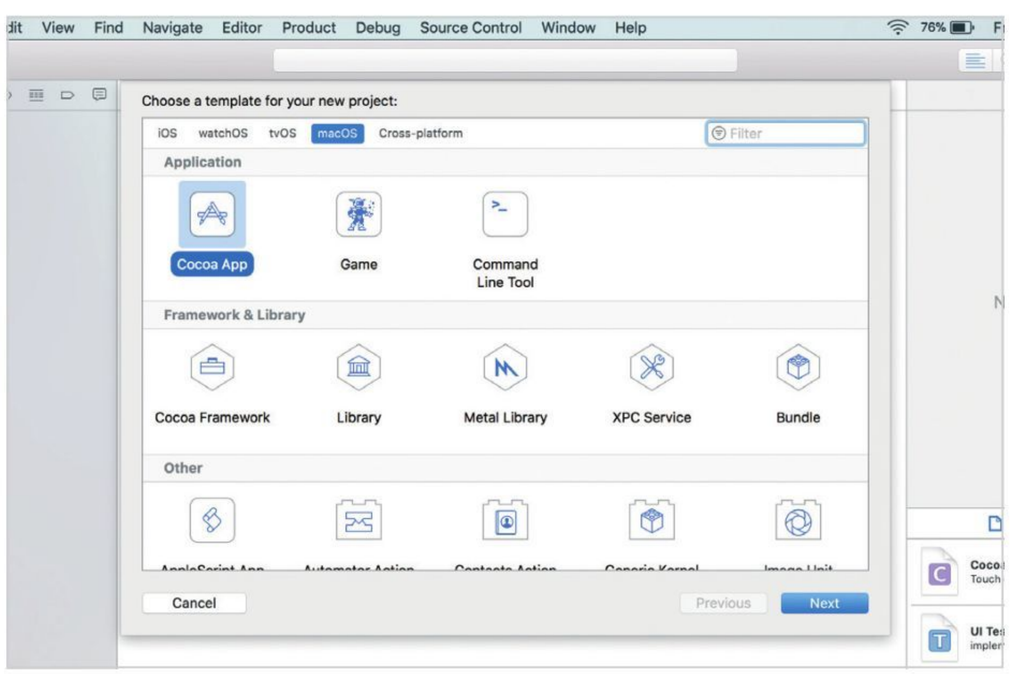

**STEP 4** When the installation is complete, click on the Open button to launch Xcode. Click Agree to the licence terms and enter your password to allow Xcode to make changes to the system. When you've done that, Xcode begins to install additional components.



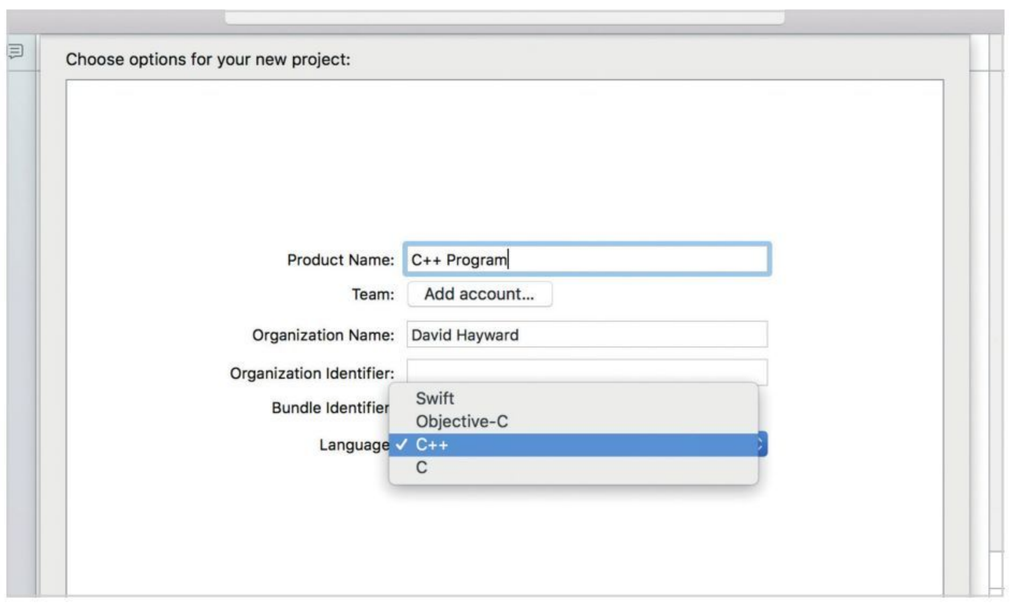
**STEP 5** With everything now installed, including the additional components, Xcode launches, displaying the version number along with three choices and any recent projects that you've worked on; although for a fresh install, this shows blank.



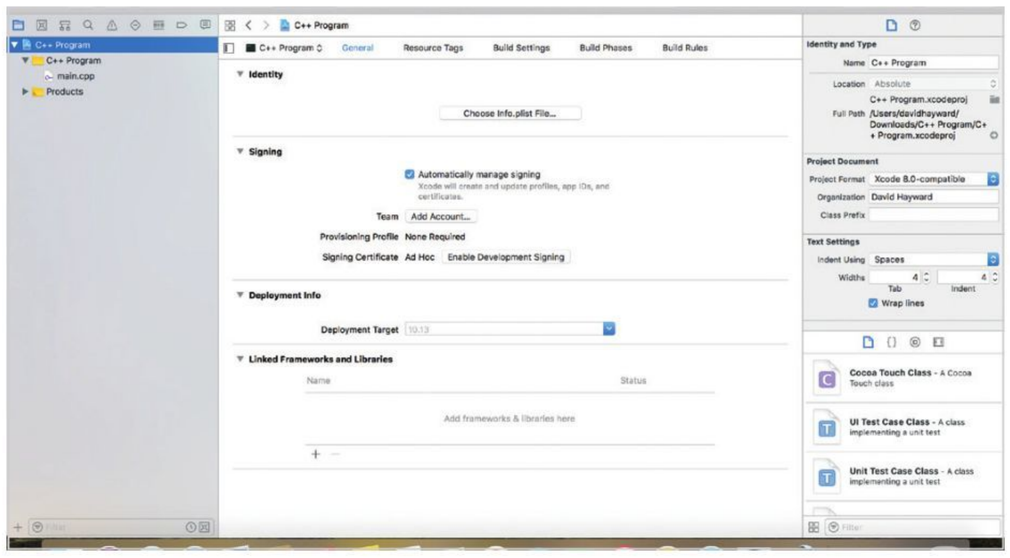

**STEP 6** Start by clicking on Create New Xcode Project; this opens a template window from which to choose the platform you're developing code for. Click the macOS tab, then the Command Line Tool option and finally, Next to continue.



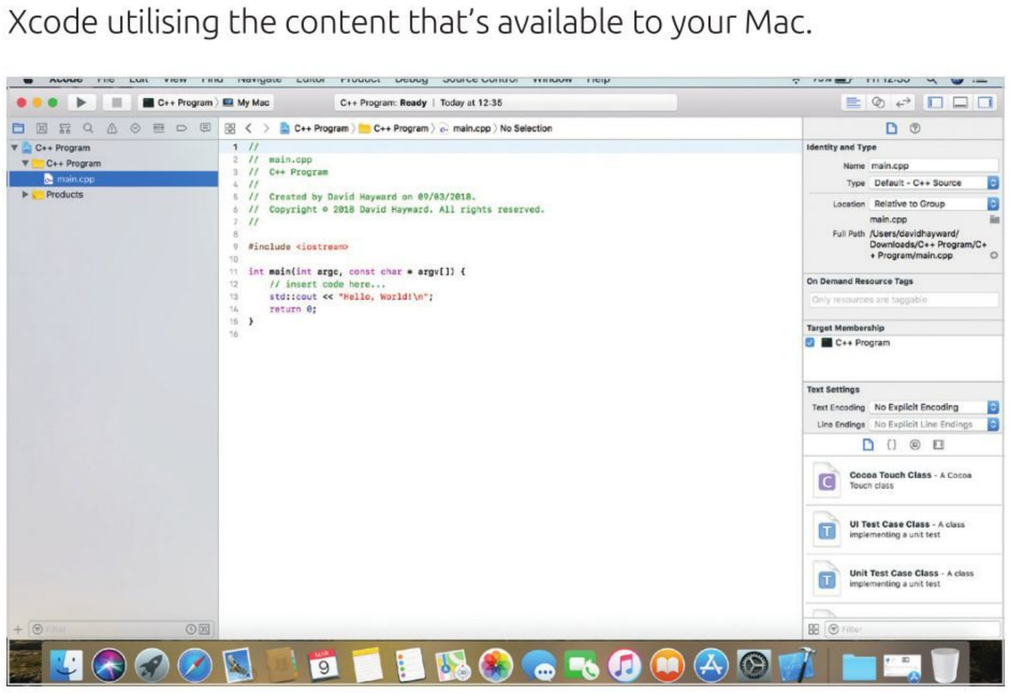

**STEP 7** Fill in the various fields but ensure that the Language option at the bottom is set to C++. Simply choose it from the drop-down list. When you've filled in the fields, and made sure that C++ is the chosen language, click on the Next button to continue.



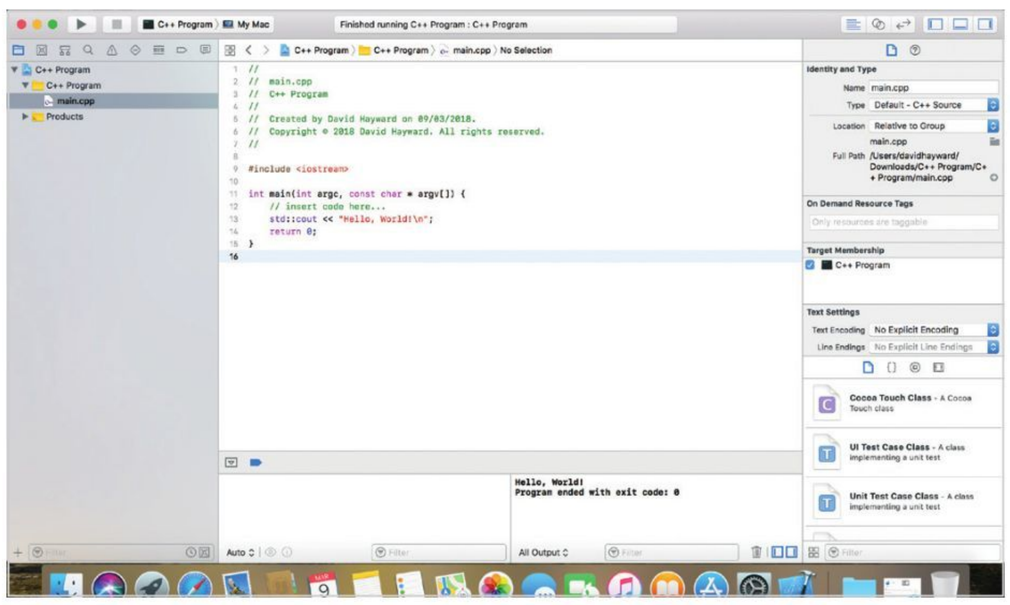

**STEP 8** The next step asks where to create a Git Repository for all your future code. Choose a location on your Mac, or a network location, and click the Create button. When you've done all that, you can start to code. The left-hand pane details the files used in the C++ program you're coding. Click on the main.cpp file in the list.




**STEP 9** You can see that Xcode has automatically completed a basic Hello World program for you. The differences here are that the int main () function now contains multiple functions and the layout is slightly different. This is just Xcode utilising the content that's available to your Mac.




**STEP 10** When you want to run the code, click on Product > Run. You may be asked to enable Developer Mode on the Mac; this is to authorise Xcode to perform functions without needing your password every session. When the program executes, the output is displayed at the bottom of the Xcode window.



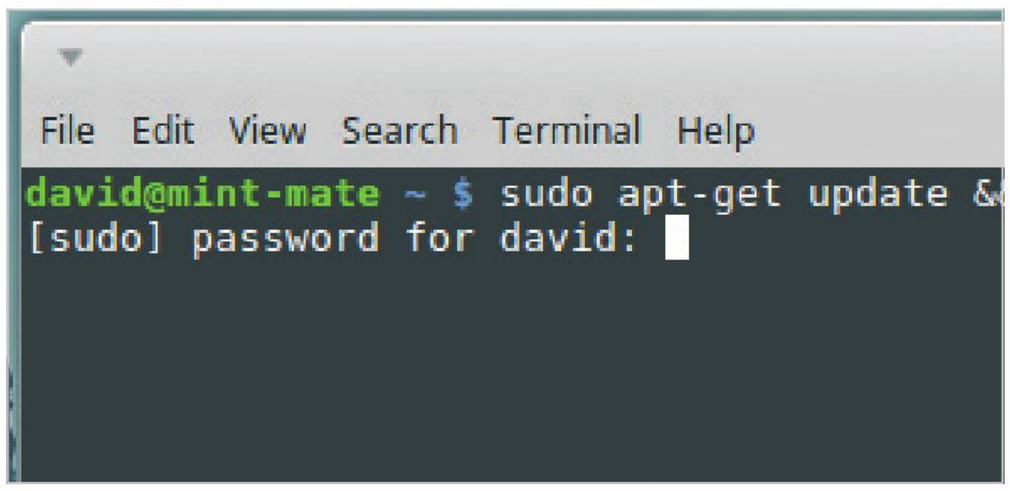
# How to Set Up C++ in Linux

Linux is a great C++ coding environment. Most Linux distros already have the essential components preinstalled, such as a compiler. The text editors are also excellent for entering code into, including colour coding. There's also tons of extra software available to help you out.

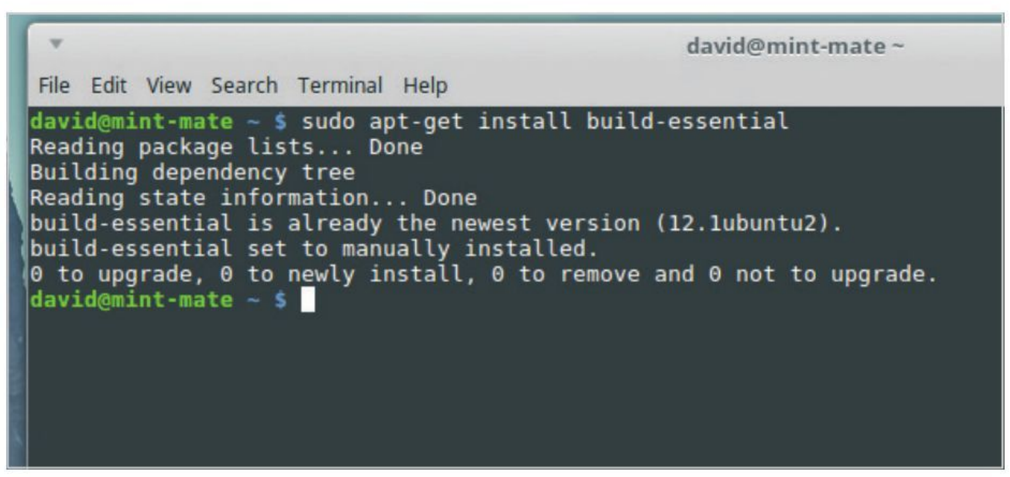
## LINUX++

**We're going to be using a fresh installation of Linux Mint for this particular tutorial. There's a coding on Linux section later in the book.**

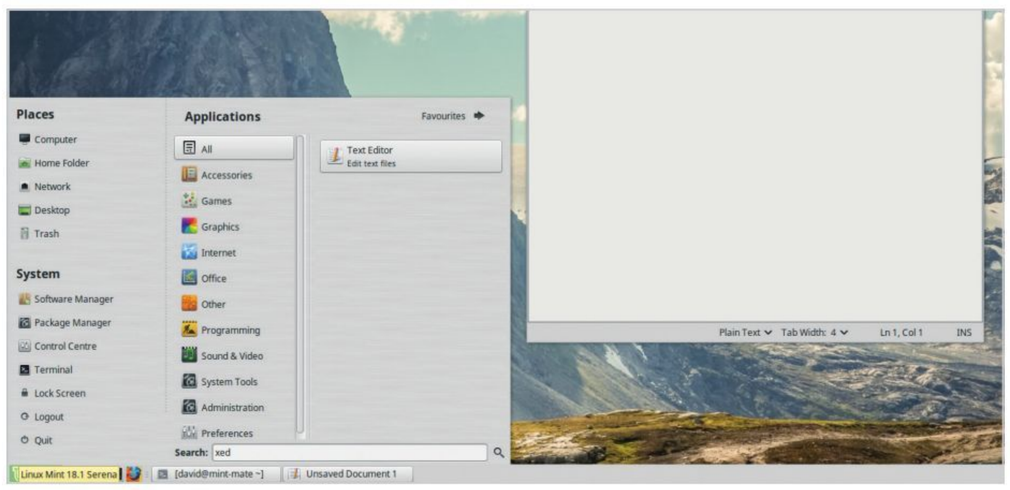
**STEP 1** The first step with ensuring Linux is ready for your C++ code is check the system and software are up to date. Open a Terminal and enter: `sudo apt-get update && sudo apt-get upgrade`. Press Return and enter your password. These commands updates the entire system and any installed software.




**STEP 2** Most Linux distros come preinstalled with all the necessary components to start coding in C++. However, it's always worth checking to see if everything is present, so still within the Terminal, enter: `sudo apt-get install build-essential` and press Return. If you have the right components, nothing is installed but if you're missing some then they are installed by the command.

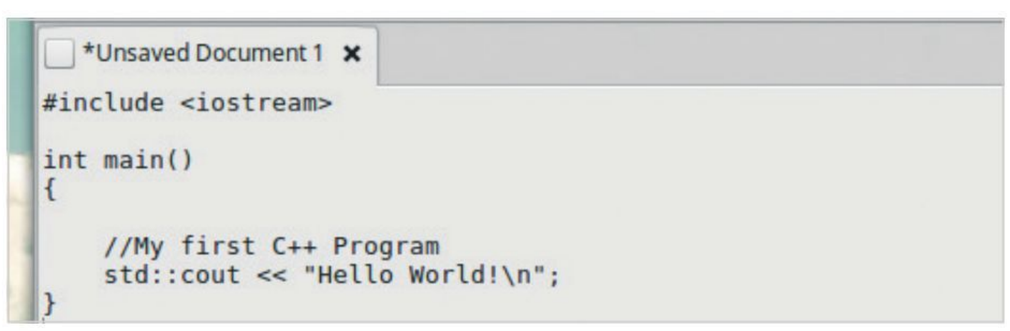



**STEP 3** Amazingly, that's it. Everything is all ready for you to start coding. Here's how to get your first C++ program up and running. In Linux Mint the main text editor is Xed can be launched by clicking on the Menu and typing `Xed` into the search bar. Click on the Text Editor button in the right-hand pane to open Xed.

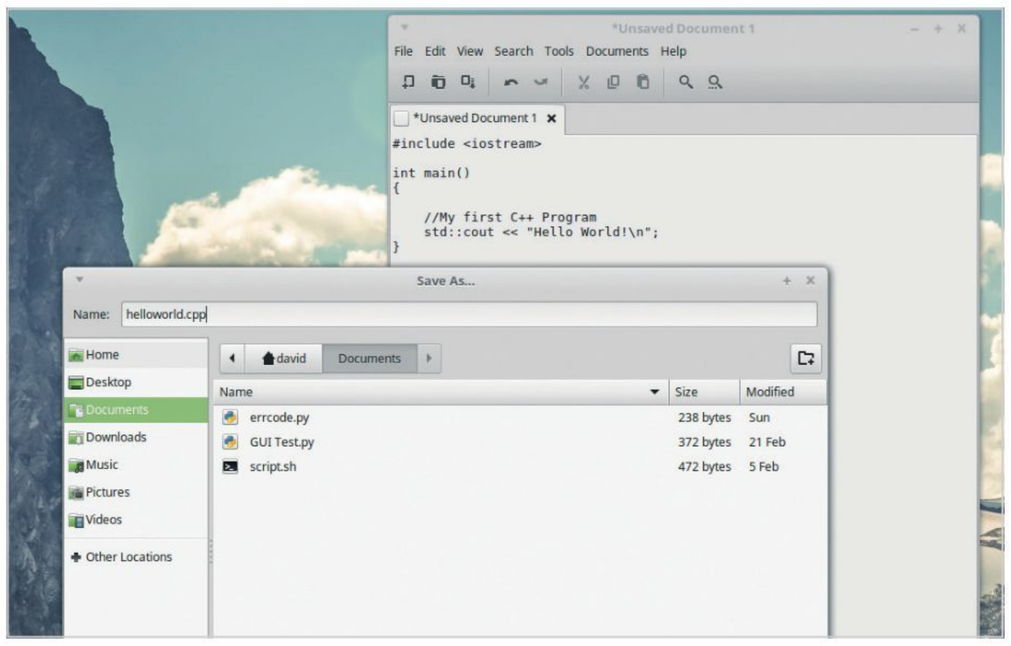



**STEP 4** In Xed, or any other text editor you may be using, enter the lines of code that make up your C++ Hello World program. To remind you, its:

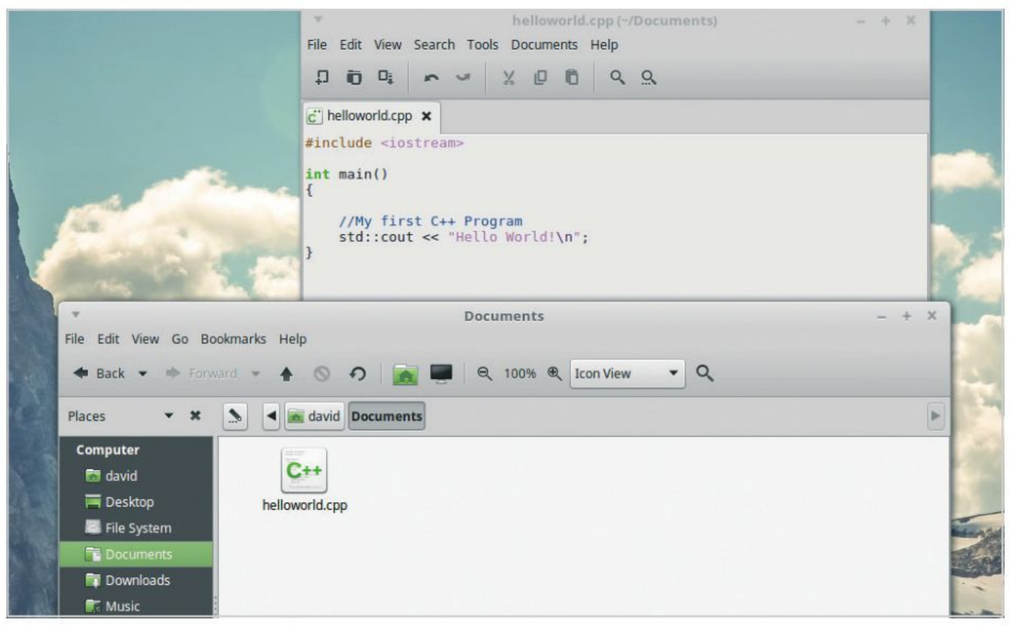
```
#include <iostream>

int main()

{

//My first C++ program

std::cout <<  "Hello World!\n";
```



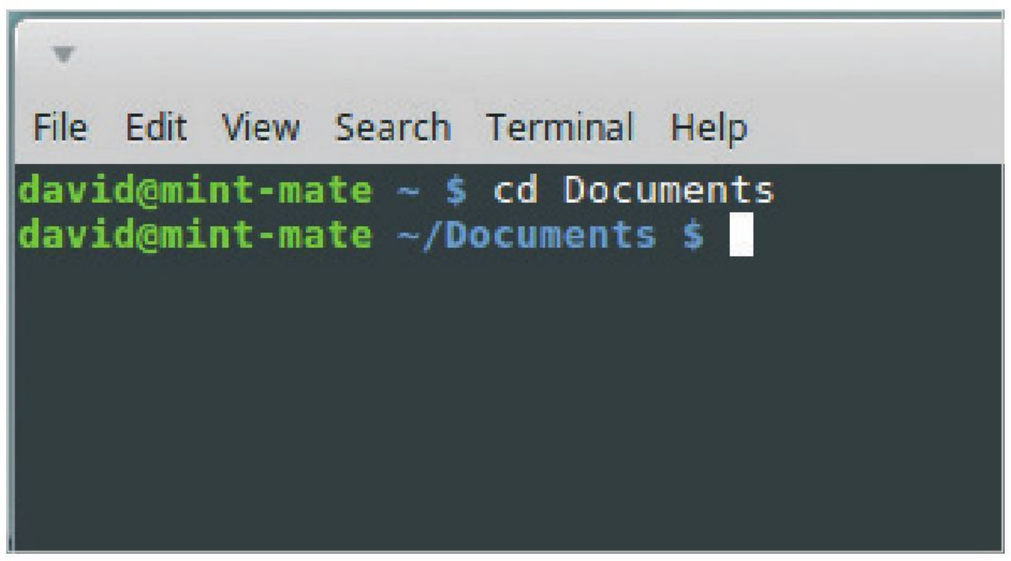
**STEP 5** When you've entered your code, click File > Save As and choose a folder where you want to save your program. Name the file as helloworld.cpp, or any other name just as long as it has .cpp as the extension. Click Save to continue.



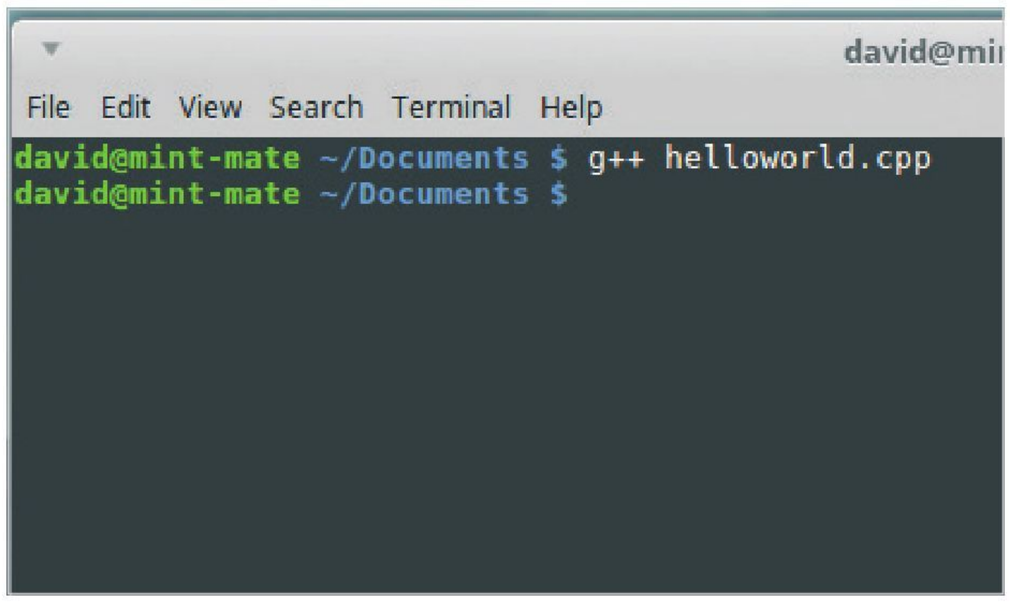

**STEP 6** The first thing you can see is that Xed has automatically recognised this as a C++ file, since the file extension is now set to .cpp. The colour coding is present in the code and if you open up the file manager you can also see that the file's icon has C++ stamped on it.



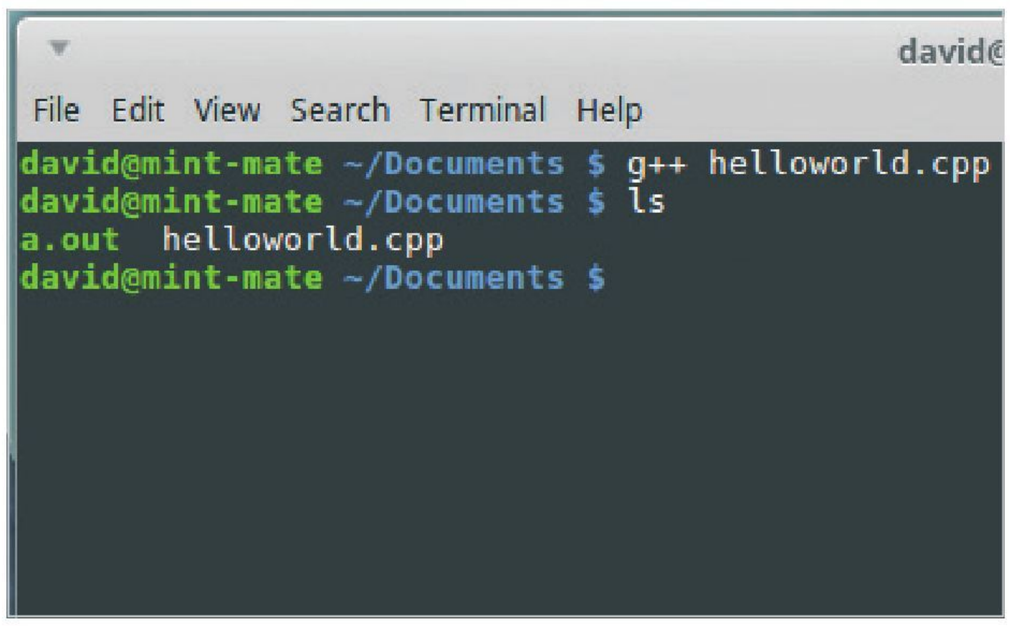

**STEP 7** With your code now saved, drop into the Terminal again. You need to navigate to the location of the C++ file you've just saved. Our example is in the Documents folder, so we can navigate to it by entering: `cd Documents`. Remember, the Linux Terminal is case sensitive, so any capitals must be entered correctly.



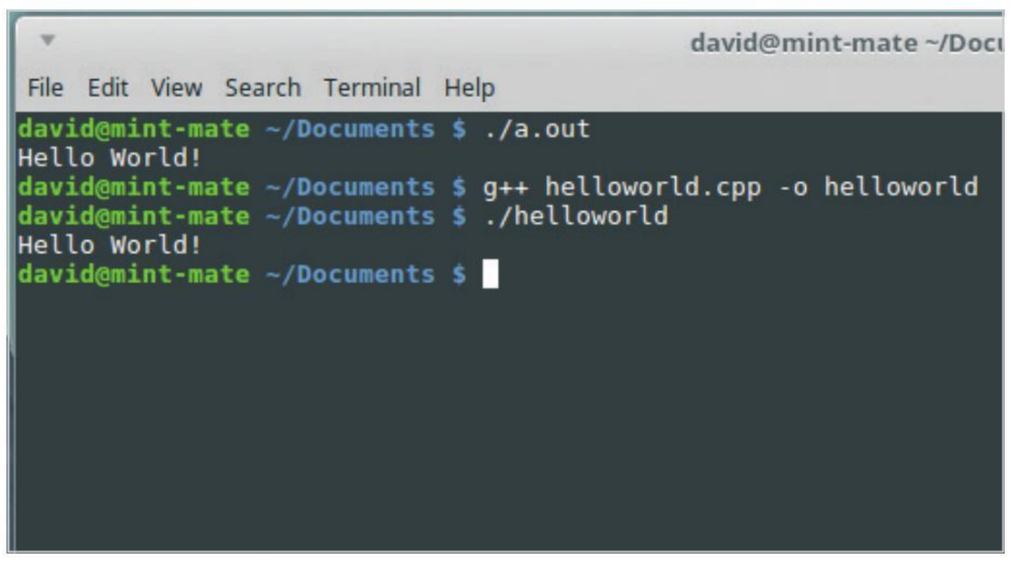

**STEP 8** Before you can execute the C++ file you need to compile it. In Linux it's common to use g++, an open source C++ compiler and as you're now in the same folder as the C++ file, go to the Terminal, enter: `g++ helloworld.cpp` and press return.




**STEP 9** There will be a brief pause as the code is compiled by g++ and providing there are no mistakes or errors in the code you are returned to the command prompt. The compiling of the code has created a new file. If you enter `ls` into the Terminal you can see that alongside your C++ file is a.out.




**STEP 10** The a.out file is the compiled C++ code. To run the code enter: `./a.out` and press Return. The words 'Hello World!' appears on the screen. However, a.out isn't very friendly. To name it something else post-compiling, you can recompile with: `g++ helloworld.cpp -o helloworld`. This creates an output file called helloworld which can be run with: `./helloworld`.



# Other C++ IDEs to Install

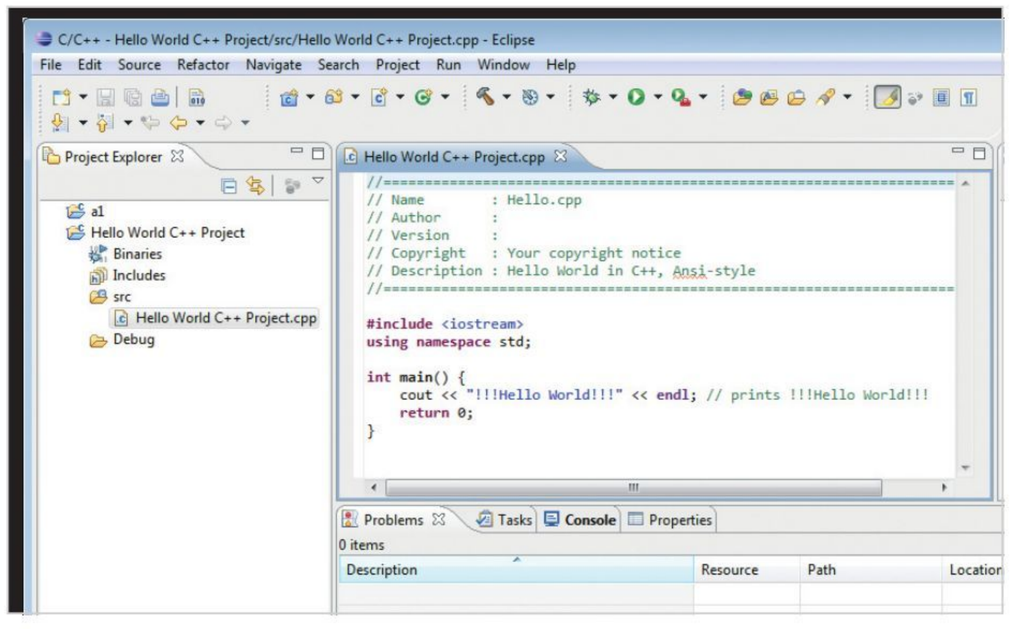
If you want to try a different approach to working with your C++ code, then there are plenty of options available to you. Windows is the most prolific platform for C++ IDEs but there are plenty for Mac and Linux users too.

## DEVELOPING C++

**Here are ten great C++ IDEs that are worth looking into. You can install one or all of them if you like, but find the one that works best for you.**

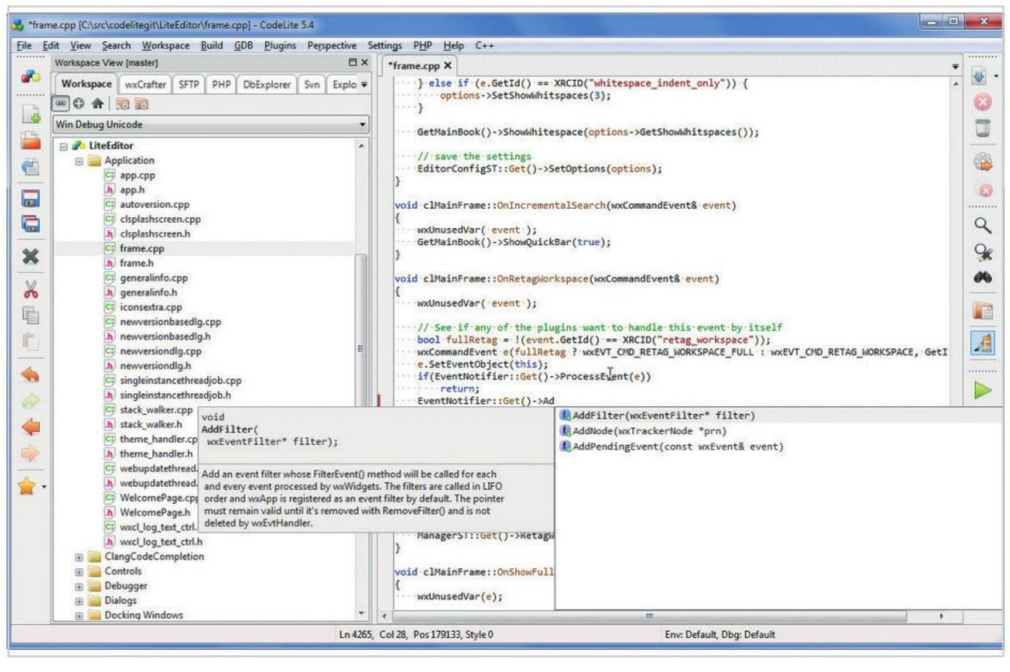
### ECLIPSE

Eclipse is a hugely popular C++ IDE that offers the programmer a wealth of features. It has a great, clean interface, is easy to use and available for Windows, Linux and Mac. Head over to **www.eclipse.org/downloads/** to download the latest version. If you're stuck, click the Need Help link for more information.



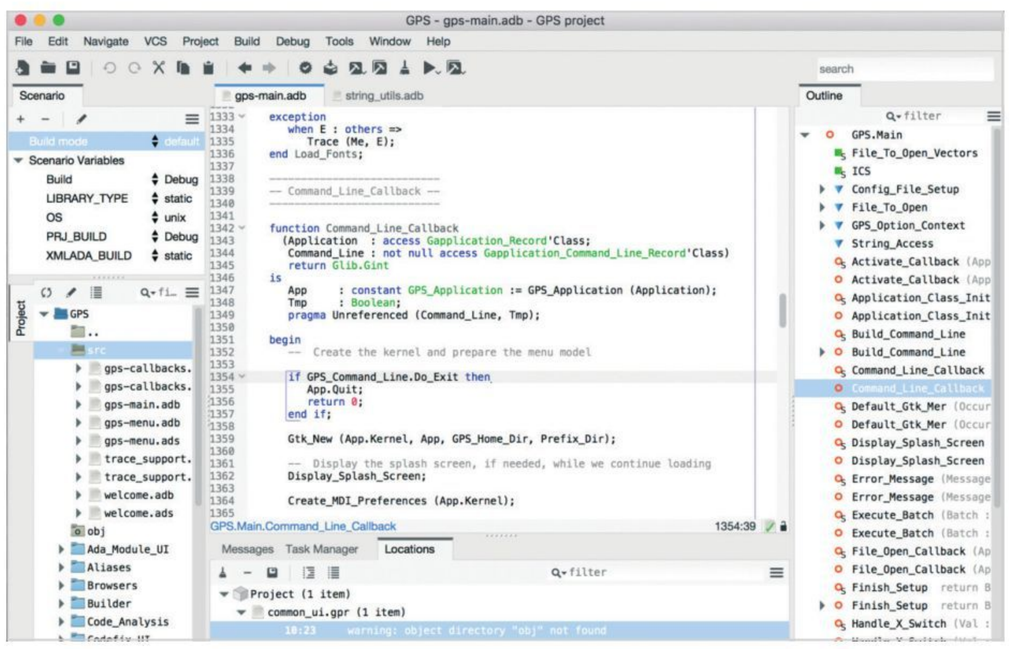

### CODELITE

CodeLite is a free and open source IDE that's regularly updated and available for Windows, Linux and macOS. It's lightweight, uncomplicated and extremely powerful. You can find out more information as well as how to download and install it at **www.codelite.org/**.



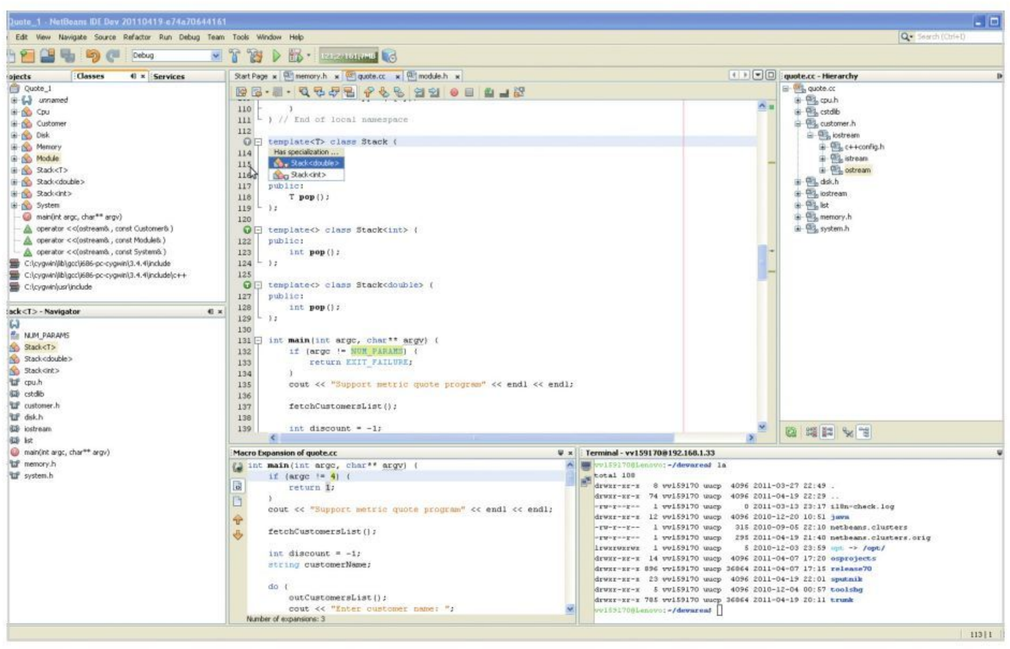

### GNAT

The GNAT Programming Studio (GPS) is a powerful and intuitive IDE that supports testing, debugging and code analysis. The Community Edition is free, whereas the Pro version costs; however, the Community Edition is available for Windows, Mac, Linux and even the Raspberry Pi. You can find it at www.adacore.com/download.



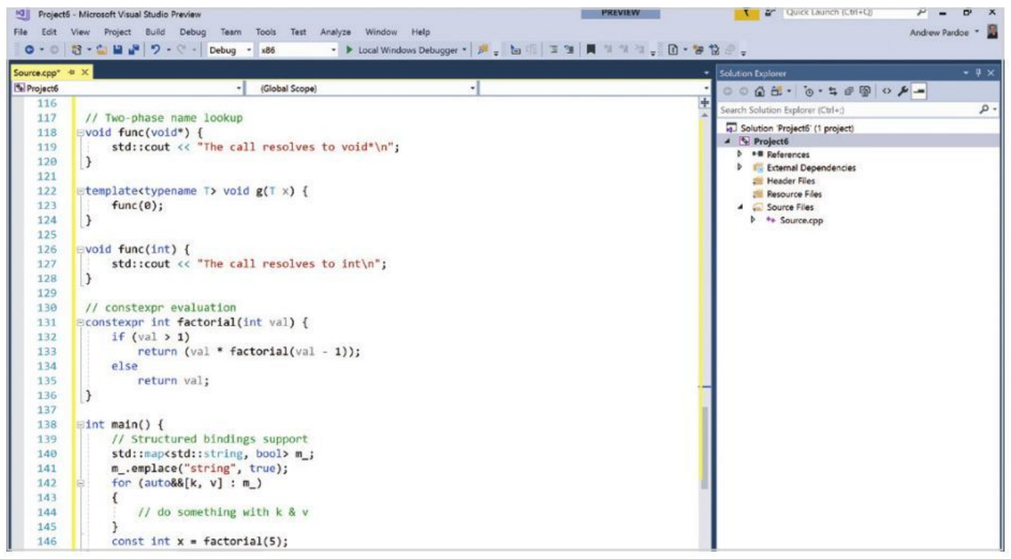

### NETBEANS

Another popular choice is NetBeans. This is another excellent IDE that's packed with features and a pleasure to use. NetBeans IDE includes project based templates for C++ that give you the ability to build applications with dynamic and static libraries. Find out more at **www.netbeans.org/features/cpp/index.html.**



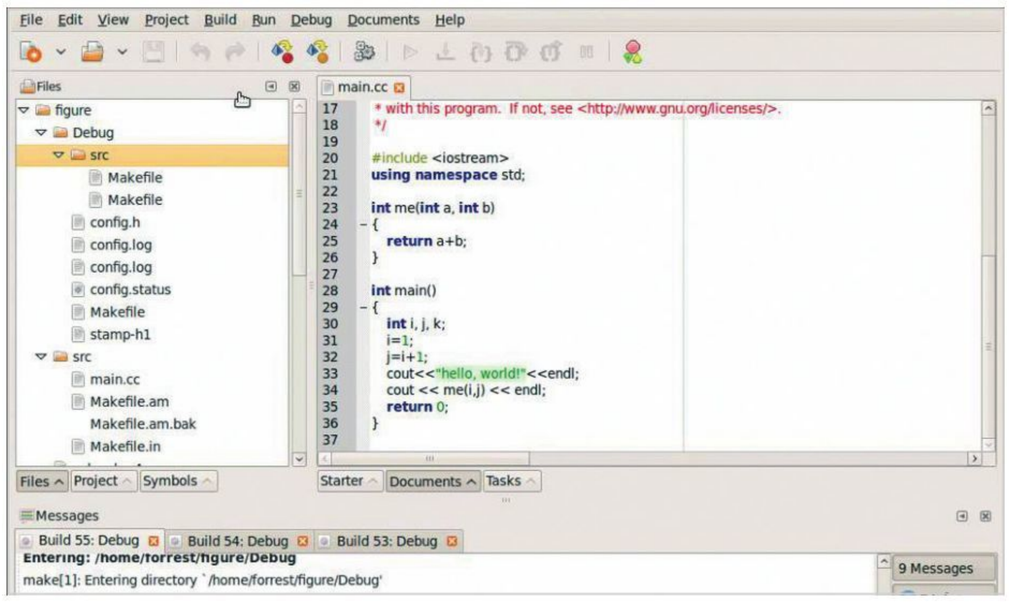
## VISUAL STUDIO

Microsoft's Visual Studio is a mammoth C++ IDE that allows you to create applications for Windows, Android, iOS and the web. The Community version is free to download and install but the other versions allow a free trial period. Go to **www.visualstudio.com/** to see what it can do for you.



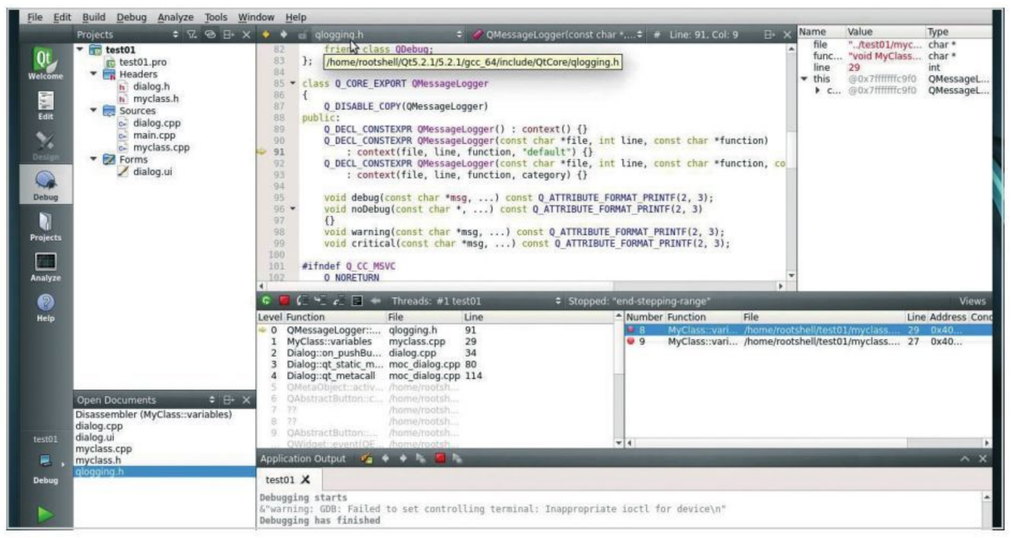

## ANJUTA

The Anjuta DevStudio is a Linux-only IDE that features some of the more advanced features you would normally find in a paid software development studio. There's a GUI designer, source editor, app wizard, interactive debugger and much more. Go to **www.anjuta.org/** for more information.



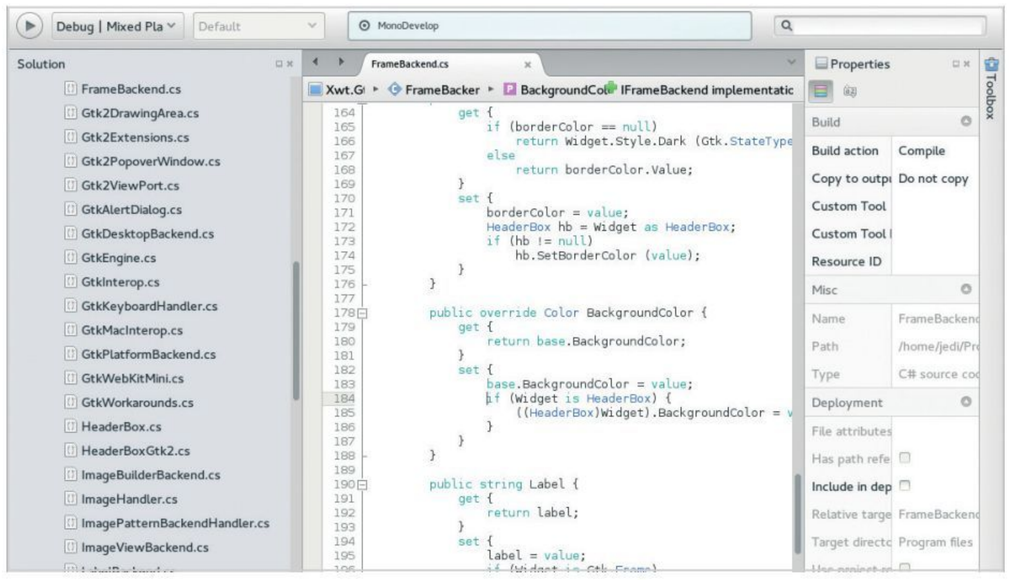

## QT CREATOR

This cross-platform IDE is designed to create C++ applications for desktop and mobile environments. It comes with a code editor and integrated tools for testing and debugging, as well as deploying to you chosen platform. It's not free but there is a trial period on offer before requiring purchasing: **www.qt.io/qt-features-libraries-apis-tools-and-ide/**.



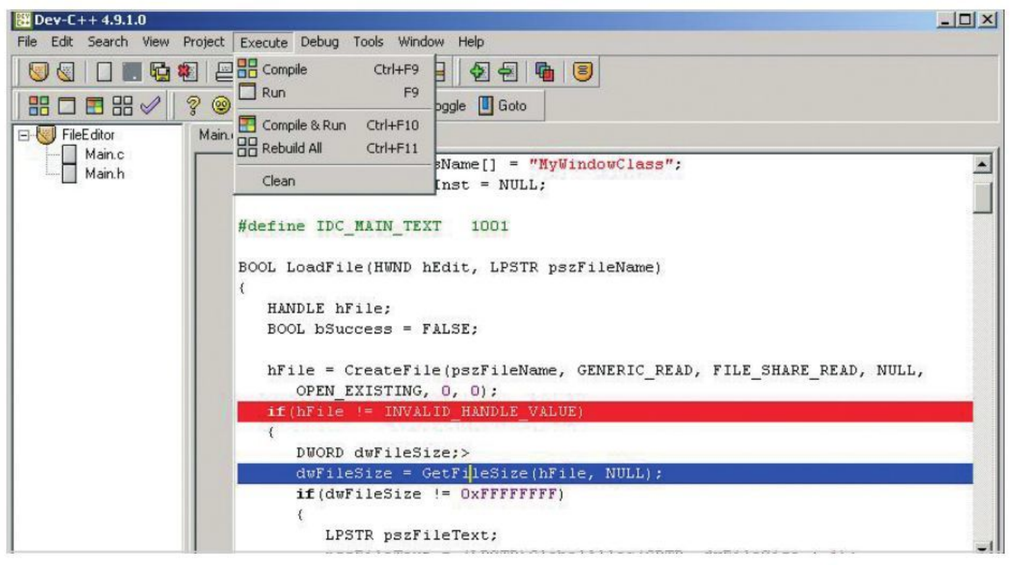

## MONODEVELOP

This excellent IDE allows developers to write C++ code for desktop and web applications across all the major platforms. There's an advanced text editor, integrated debugger and a configurable workbench to help you create your code. It's available for Windows, Mac and Linux and is free to download and use: **www.monodevelop.com/**.




## DEV C++

Bloodshed Dev C++, despite its colourful name, is an older IDE that is for Windows systems only. However, many users praise its clean interface and uncomplicated way of coding and compiling. Although there's not been much updating for some time, it's certainly one to consider if you want something different: **www.bloodshed.net/devcpp.html**.




## U++

Ultimate++ is a cross-platform C++ IDE that boasts a rapid development of code through the smart and aggressive use of C++. For the novice, it's a beast of an IDE but behind its complexity is a beauty that would make a developer's knees go wobbly. Find out more at **www.ultimatepp.org/index.html**.