Install some basic tool

This lesson will introduce you to ways to compile and run C code on your personal computer based on your operating system.

We'll cover the following

No matter which operating system you are using (GNU/Linux, Mac, Windows), you will need at least two things:

1. A way to edit source code files
2. A C compiler (& debugger)

GNU/Linux

You have it easy. Most GNU/Linux distributions include a **package manager** that allows you to quickly and easily install software from centralized, up-to-date repositories.

Compiler

If you are using Ubuntu, just open up a Terminal and type the following command to install the tools you will need:

```
sudo apt-get install build-essential
```

That's it! Now you have a C compiler (`gcc`) and a bunch of other developer tools including a debugger (`gdb`). You can verify you have `gcc` installed by typing:

```
gcc --version
```

Source code editors

When you write C source code, you save it in one or more plain ascii text files. Any editor that is capable of reading and writing plain text files, can be used to work with C code.

For example, Ubuntu comes with a relatively nice GUI-based editor called `gedit`. It has multiple tabs, it has syntax highlighting for source code (not only for C but other languages as well) and it has various plugins for adding other functionality.

Some other classic editors include GNU Emacs and Vi / Vim. We won't get into an editor war here, but suffice it to say that both are powerful and useful, and both have relatively steep learning curves.

For the purposes of these tutorials, you can use any editor you want. If you don't know what to choose, then I suggest starting with `gedit`.

There's a pretty good cross-platform (linux, Mac OS X, Windows) source-code oriented editor called Sublime Text 2 that I can also recommend. It has a lot of neat functionality that's particularly oriented towards source code editing (in many languages, not just C). Check it out.

Mac OS X

Compiler

On the Mac, the standard way of obtaining the required developer tools is to download Apple's Xcode software. You can either download the entire **Xcode** package, which includes a GUI-based IDE and Apple's extensive Cocoa and iOS frameworks for programming on the Mac and on iOS devices, or you can install only the **Command Line Tools for OS X**. For this tutorial the command line tools will suffice.

Unfortunately Apple forces you to register for an Apple Developer ID first and give them all sorts of personal information. The Apple website makes it look like you have to pay money but in fact you don't.

Just like in GNU/Linux, you can verify that you have the compiler installed by opening up a Terminal (located in *Applications/Utilities*) and typing:

```
gcc --version
```

Source code editors

On the mac there are a host of editors to choose from. OS X comes with TextEdit, which can read/write rich-text format (rtf) as well as plain text. A better idea however is to use an editor that has syntax-highlighting and other features that are developer-friendly.

If you downloaded the full Xcode IDE then you can edit source code in Xcode. Note that Xcode is a gigantic and complex development environment, and so unless you already know how to use it, or are prepared to put the time in to learn how it works, I would suggest using a simpler editor. We won't have time to go through how to use Xcode in this series of tutorials.

Good choices are TextMate and BBEdit. You can of course also install Vim (macvim) and GNU Emacs (Aquamacs Emacs or Pure Emacs) on the mac if you prefer.

Sublime Text 2 also works on the mac.

There is another choice on the Mac which is called CodeRunner, which is a sort of very lightweight and simple IDE. It knows about a large number of languages besides C, and has a built-in console, which means that one can program in a slightly more interactive way than the usual edit -> compile -> run workflow.

Windows

If you are using Windows:

1. You can install and use Cygwin, which is a collection of tools that provide unix-like funcionality within Windows.
2. You can also install and use an IDE (e.g. Eclipse that will let you edit, compile and run C code.

Please note that I know very little about Windows and so unfortunately I won't be much help with doing things in Windows.

A final option is to install Ubuntu (or some other GNU/Linux distribution) on your Windows machine, either in a dual-boot configuration (Ubuntu instructions here, or within a software virtual machine using something like VMware (Mac,

Windows) or VirtualBox. This may be the fastest and easiest option to get up and running for this tutorial, which will be run using examples in GNU/Linux.

A note about IDEs

IDEs (Integrated Development Environments) are programs that combine several development tools together into one unified environment. IDEs typically include components such as a source code editor, automated build and compile tools, a debugger, extensive language and framework documentation, and sometimes other components such as GUI-based interface design tools, performance measurement tools, version control systems, Object-Oriented Programming tools (e.g. class browsers and object inspectors), and others. Examples of IDEs are Xcode (Mac) and Eclipse (GNU/Linux, Windows, Mac). There is a host of others as well, here is a Wikipedia page that lists many of them: A Comparison of IDEs.

The benefits of using an IDE start to show when your project involves a large amount of code spread over a large number of files, and/or when you make use of complex and extensive external libraries and frameworks, etc. For the purposes of these tutorials, I wouldn’t suggest using an IDE unless you are already familiar with one.