Chapter 1: Prepare Yourself

Notice to Macintosh Users

I would like to accommodate you with nice explanations of tools, instructions for using them, etc. but I do not currently own a usable Macintosh computer. If you are ansy for Macintosh-specific instructions then please send me a usable Mac or money to buy one. ☺ I plan on purchasing one in the, possibly, near future but that is still the future and not the present. For now, my instructions center on PC-based operating systems: Linux, DOS, Windows, and BeOS.

Pre-Requisites

You are going to need to know how to use your computer in a basic way before plodding through this chapter and those past it. I suggest you know how to do the following things already:

- Navigate files: browse, rename, move, copy, etc.
- Install/uninstall programs
- Download files from the world wide web (internet)
- Write documents in an editor such as Word, Works, WordPerfect, Notepad, WordPad, etc.

Blarg.

Introduction

You'll need two things to build C++ programs:

1. Plain Text Editor
2. C++ Build Tools

Sometimes these things are combined into a single software package known as an *Integrated Development Environment* or "IDE". However, all C++ software comes with *build tools* for use on the command-line; otherwise known as "command-line build tools" or just "command line tools" (amazing!). So among other things[1], for the purpose of reaching the audience on a common ground, I will not be tutoring you in any IDE.

[1] Other reasons for not explaining all the current IDE's: there are tons of them, they change frequently (would make this book void very quickly), and they only shield you from what is actually going on. Eventually you will probably settle into using an IDE, but I believe it's a good idea to know what its doing first.

Plain Text Editor

To write any kind of *source code* you need a plain text editor, commonly known simply as a text editor. The difference between a text editor and a word-processing application is it has *no* fancy formatting options. Files saved in a text editor are *plain* text files. These files are *WYSIWYG*, or "what you see is what you get". Every character you write into a plain text file is a character in the file; no extra formatting is added. The files are bare to the bone. They have only what you type into them and nothing else.

A plain text editor is akin to a typewriter with no formatting options whatsoever. You know the ones, where the keys would stick if you hit them to hard. Fortunately the advances in software have made this "sticky-key" problem all but disappear. ☺

Plain text files are used to store *source code* which is the instructions for a C++ program. This source code must be totally plain, with no surrounding formatting; otherwise the C++ software will not recognize it. For this reason it is a bad idea to use word-processing software that is geared towards any kind of formatting: like Word, WordPad, Works, WordPerfect, etc. These programs will typically save text files that have more than just text and/or they may change certain characters. Once this happens the files are no longer usable.

Think of a plain text file as a piece of paper with directions to someone's house and a formatted text file as the same thing, but written in ancient Greek. The plain directions can be read by anyone who understands Basic English, while the ancient Greek can only be understood by a select few. This is why C++ source code is written in plain text rather than formatted text. The formatted text is only understood by word-processing software that understands specific types of files, it is the ancient Greek.

Some plain text editors display the text you are editing as if it was formatted: with colors, bold, italics, etc. These visuals are drawn from the *source code* itself and this is known as *syntax coloring* or *syntax highlighting*.

Author's Preference: Get an editor with syntax coloring. It'll help you distinguish the parts of the code as well as find typos faster.

It is easy to find plain text editors and there are many of them. However, not all text editors are created equal. I have my own biases as well as many other people. Some compilers come with text editors as part of their IDE; those will not be discussed here.

The following are a list of the most popular freeware or shareware[2] text editors, organized alphabetically (I try to be fair), and set aside a web site URL where you can find more information (more editors will be available here upon request):

[2] Shareware is the ability to try something indefinitely for the price of suffering through reminders that the program is not free. My advice is to buy it if you like it and use it, otherwise suffer through the reminders☺.

TextPad	http://www.textpad.com/
UltraEdit	http://www.ultraedit.com/

You should familiarize yourself with creating new, opening, and saving text files; especially those with different extensions like '`.cpp`' and '`.h`' (rather than just '`.txt`'). Create a new document and type the following into your text editor; save it as '`hello.cpp`':

```
#include <iostream.h>

int main()
{
    cout << "Hello World" << endl;
    return 0;
}
```

If your text editor is setup for syntax highlighting then the text of the above will be visually different after you save it with the '`.cpp`' extension. That is to say, it will be more colorful and easier to distinguish the individual parts.

For an up to date listing or more information on installing and/or obtaining these fine text editors, please visit http://www.neilstuff.com/tools/editors. The explanations of software in this document are meant to be an introduction, but by no means full-fledged documentation. For more extensive works on the software, you should consult their web site and/or printed documentation.

Using TextPad

For more on TextPad you should visit http://www.textpad.com/. I would suggest this editor to all newbies and encourage you to purchase it. It is easy to use and has a lot of powerful features that you can slip into as you advance.

Creating New

When you first run TextPad, it should open up with a blank, or new, document for you. If you screw that up, accidentally close it, or just want to start over you can open a fresh new document by selecting "File > New" with the menu's or by pressing "Ctrl + N" on your keyboard.

Opening

To open a document you have already saved, select "Open" from the "File" menu. There you can either type in the full location of the file to open, or browse for it.

Saving

If the document you are working on has already been saved to a file, you can save to it again by selecting "File > Save" with the menu's or by pressing "Ctrl + S" on your keyboard. If you have not yet saved, you will have to select "File > Save As" from the menus or press "F12" on your keyboard.

When saving for the first time you will be asked to type in a file name and possibly a location. Once you have done this click the "Save" button with your mouse or press "enter" on your keyboard. If you choose not to save at this time, click the "Cancel" button with your mouse or press "Escape" on your keyboard.

Using UltraEdit

For more on UltraEdit you should visit http://www.ultraedit.com/. I would suggest this editor to advanced computer users who are willing to spend some time learning new software. UltraEdit is similar to TextPad, but has additional features you may or may not enjoy.

Creating New

blah blah.

Opening

blah blah blah.

Saving

blah blah blah.

Command Console

Before pointing out various C++ Build Tools you need to learn how to use the command console. A command console is a text editing window of sorts that lets you enter commands via the keyboard into the window and see the results in the same way. These are sometimes seen in games as well, like the console in any game of the Quake series. The term 'command console' is only one of the many terms that this goes by. The others are 'terminal', 'command window', 'DOS box', 'cmd window', 'shell', or 'console window'. These all refer to the ability to type commands and receive output in the same screen. If you've used DOS then this is what you've already done many, *many* times.

Depending on your operating system you will be working with one of two types of consoles: DOS-Based or Unices. If you are using OS/2, any flavor of Windows, or any flavor of DOS then see the section on "DOS-Based Console Usage". If you are using Mac OS X, any flavor of Linux or UNIX, or BeOS then read the section on "Unices

Console Usage". If you have an operating system different than what I have listed you will have to familiarize yourself with your system's command console without my assistance.

For up to date instructions on using a command console for a variety of operating systems, please visit http://www.neilstuff.com/tools/console.

DOS-Based Console Usage

Opening a Command Console

Blah blah blah.

Listing Files

Blah blah blah.

Deleting Files

Blah blah blah.

Renaming Files

Blah blah blah.

Running Programs

Blah blah blah.

!!!tbd!!! … various names, opening, exiting, list files, delete files, rename files

Unices Console Usage

Blah blah blah.

C++ Build Tools

C++ build tools are typically provided as a single package which includes a pre-processor, compiler, and linker. The term C++ *compiler* is often used to refer to such a package. People list free "compilers" or look for free "compilers", but they almost always want the whole package.

Plenty of free or cheap C++ build tools exist today. If you're using Linux then you must already know this because it comes with GNU C/C++ or GCC. BeOS also comes with GCC along with an IDE (BeIDE) to help you use it. If you have a more mainstream OS such as Mac or Windows you will have to find yourself C++ software:

Mac
- to be done

Windows
- Borland C++Builder 5.5
- MinGW

DOS
- DJGPP
- Turbo C++ Lite (TC-Lite)

Unices
- GNU C++

For a more up to date list and/or instructions on how to obtain and install these products, visit http://www.neilstuff.com/cpp/software.

Using Borland C++Builder 5.5

Blah blah blah.

Using MinGW

Blah blah blah.

Using DJGPP

Blah blah blah.

Using TC-Lite

Blah blah blah.