**Compilers**

The essential tools needed to follow these tutorials are a computer and a compiler toolchain able to compile C++ code and build the programs to run on it.  
  
C++ is a language that has evolved much over the years, and these tutorials explain many features added recently to the language. Therefore, in order to properly follow the tutorials, a recent compiler is needed. It shall support (even if only partially) the features introduced by the 2011 standard.  
  
Many compiler vendors support the new features at different degrees. See the bottom of this page for some compilers that are known to support the features needed. Some of them are free!  
  
If for some reason, you need to use some older compiler, you can access an older version of these tutorials [here](http://www.cplusplus.com/doc/oldtutorial/) (no longer updated).

**What is a compiler?**

Computers understand only one language and that language consists of sets of instructions made of ones and zeros. This computer language is appropriately called *machine language*.  
  
A single instruction to a computer could look like this:

|  |  |
| --- | --- |
| 00000 | 10011110 |

A particular computer's machine language program that allows a user to input two numbers, adds the two numbers together, and displays the total could include these machine code instructions:

|  |  |
| --- | --- |
| 00000 | 10011110 |
| 00001 | 11110100 |
| 00010 | 10011110 |
| 00011 | 11010100 |
| 00100 | 10111111 |
| 00101 | 00000000 |

As you can imagine, programming a computer directly in machine language using only ones and zeros is very tedious and error prone. To make programming easier, high level languages have been developed. High level programs also make it easier for programmers to inspect and understand each other's programs easier.  
  
This is a portion of code written in C++ that accomplishes the exact same purpose:

|  |  |  |
| --- | --- | --- |
| 1 2 3 4 5 6 7 | int a, b, sum;    cin >> a;  cin >> b;    sum = a + b;  cout << sum << endl; |  |

Even if you cannot really understand the code above, you should be able to appreciate how much easier it will be to program in the C++ language as opposed to machine language.  
  
Because a computer can only understand machine language and humans wish to write in high level languages high level languages have to be re-written (translated) into machine language at some point. This is done by special programs called compilers, interpreters, or assemblers that are built into the various programming applications.  
  
C++ is designed to be a compiled language, meaning that it is generally translated into machine language that can be understood directly by the system, making the generated program highly efficient. For that, a set of tools are needed, known as the development toolchain, whose core are a compiler and its linker.

**Console programs**

Console programs are programs that use text to communicate with the user and the environment, such as printing text to the screen or reading input from a keyboard.  
  
Console programs are easy to interact with, and generally have a predictable behavior that is identical across all platforms. They are also simple to implement and thus are very useful to learn the basics of a programming language: The examples in these tutorials are all console programs.  
  
The way to compile console programs depends on the particular tool you are using.  
  
The easiest way for beginners to compile C++ programs is by using an Integrated Development Environment (IDE). An IDE generally integrates several development tools, including a text editor and tools to compile programs directly from it.  
  
Here you have instructions on how to compile and run console programs using different free Integrated Development Interfaces (IDEs):

|  |  |  |
| --- | --- | --- |
| **IDE** | **Platform** | **Console programs** |
| **Code::blocks** | Windows/Linux/MacOS | [Compile console programs using Code::blocks](http://www.cplusplus.com/doc/tutorial/introduction/codeblocks/) |
| **Visual Studio Express** | Windows | [Compile console programs using VS Express 2013](http://www.cplusplus.com/doc/tutorial/introduction/visualstudio/) |
| **Dev-C++** | Windows | [Compile console programs using Dev-C++](http://www.cplusplus.com/doc/tutorial/introduction/devcpp/) |

If you happen to have a Linux or Mac environment with development features, you should be able to compile any of the examples directly from a terminal just by including C++11 flags in the command for the compiler:

|  |  |  |
| --- | --- | --- |
| **Compiler** | **Platform** | **Command** |
| **GCC** | Linux, among others... | g++ -std=c++0x example.cpp -o example\_program |
| **Clang** | OS X, among others... | clang++ -std=c++11 -stdlib=libc++ example.cpp -o example\_program |