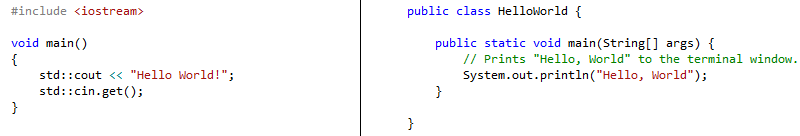
**Introduction to C++ Syntax**

**Main**

The main function is the starting point of every C++ program. You have to have it in your code somewhere, or the compiler will not know where to start the program from. Inside our HelloWorld project, we have a single file which contains this function. Unlike Java or C#, it is not inside a class, but sits in what we can call ‘*Global Namespace*’. It is basically a function that is on its own in a program.

Here we have a comparison of the HelloWorld program in C++ vs its Java counterpart.



Notice the similarities between the two. Both make use of *void main* to define their starting point, and both have similar construction of functions (*type* -> *name* -> *arguments* -> *code*).

The main difference is that Java has its main function inside of a class, in this case called HelloWorld, whereas C++ has it in Global Namespace. There is also the method each language calls to print the message into the console window, but we will get to that later.

**Breaking Down our HelloWorld Program**

We are going to break down each line of the HelloWorld program and see what each line of code does.



The first line asks the compiler to include the library iostream which stands for input output stream. This library gives us functions and classes to retrieve data from the console and output it to the console. It is the primary method we can use to print the “Hello World!” message.



This line tells the compiler that we are defining the main function and what code goes inside that. We have stated that the main is going to return void (no type or returns nothing), that it takes no arguments (the empty parenthesis).



Now that we have included the necessary library and started our main function, the next step is to ask the program to print the message. To do this, we use an item called cout which stands for console output. Now the cout item does not exist inside global namespace like the main function, but inside a namespace called std. Namespaces are sections of code encapsulated by an identifier, usually a unique name. A lot of the standard C++ libraries make use of the namespace std. If you see any code that starts with std::, then you can safely bet it’s part of a standard C++ library. To access this item, we need to call the namespace, then its name inside that namespace.



Namespace Item Name

To push the message to the console, the cout item makes use of the << operator. By adding this on to the code along with our message, we can print whatever we want to the console. Remember as well, as with Java, every line of code in C++ needs semi-colon (;) at the end to signify the end of a command.



The final line in out code does the opposite of cout. After we have printed the message, we ask the program to wait for input from the console before we can continue. It doesn’t matter what input we receive; we just need a button press to make sure that the program does not close before we have had a chance to view the message.

To do this we use the std namespace again, but this time use the item cin (standing for console input). We then call the .get() function that is part of cin, which asks the program to wait for any console input.

**Expanding the Messages**

**Chain Link with Cout**

Being able to output a single message is fine but there are a couple more useful features we can use to have better control over the cin and cout items in our program.



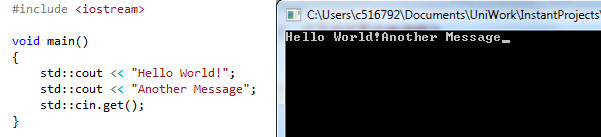
First we can add on more parts to the output by affixing << to the end of each message or command. This allows us to chain link items we want to output to the console. The above example will still print out the message “Hello World!”, but we have split it into multiple parts.



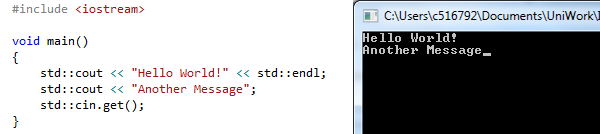
The above example does the same; we have just split the entire message into twelve single characters that we print in one command.

**End Line**

It is very useful to be able to print one message on one line of the console, then another message on the next line. Currently, if you used the following code, you would get the following output:



To print the second message on a new line, we can use an item inside the std namespace, std::endl (stands for end line). It is affixed to the message the same way we have seen above.



**Getting User Input**

Sometimes we want to get input from the user, to do that uses the cin mentioned earlier but with a small modification.



**Exercises**

Play about with the HelloWorld program to make sure you are familiar with the console output functions. Remember you can always start a new project and try again if you change something and it no longer compiles. For example:

* Change the “Hello World!” message to something else. (difficulty: 1)
* Create a program that takes two int inputs and prints the result. (difficulty: 1)
* Create a program that can print different paragraphs depending on what int is given by the user as input. (difficulty: 2)
* Create a program that prints the following riddle along with the answer: (difficulty: 3)

As I was going to St. Ives,   
I met a man with seven wives.   
Each wife had seven sacks,   
Each sack had seven cats,   
Each cat had seven kits.   
Kits, cats, sacks, and wives,   
How many were going to St. Ives?

Summary:

1. You must have the main function somewhere in your C++ code. It cannot be inside a class.
2. Make sure you have the correct includes in your code files. For example, without including the iostream library, we cannot use cout or cin.
3. Use std::cout to output a message to the console window.
4. Use std::cin to get input from the console window to your program.
5. Use std::endl to move to the next line in the console window.