Project Report On

C++ Language Overview And Evaluation (3CS-306)



Submitted in Partial Fulfillment

For the Award of

Computer University (Pathein)

Third Year (Section-A)

Group (5)

Supervised By : Daw Khin Lay Thwin

Presented By:

Ma Yoon Nway Nway Ei (Leader) 3CS-6

Mg Thu Rein Lynn 3CS-11

Ma Wah Wah 3CS-13

Ma Khin Myo Aye 3CS-19

Ma Khin Htar Nwe 3CS-26

Mg Htet Ko 3CS-27

Mg Kaung Htet San Aung 3CS-30

# What is C++ (programming language)?

“C++ is a statically-typed, free-form, (usually) compiled, multi-paradigm, intermediate-level general-purpose middle-level programming language.”

In simple terms, C++ is a sophisticated, efficient and a general-purpose programming language based on C. It was developed by [Bjarne Stroustrup](http://www.stroustrup.com/) in 1979.

Many of today’s operating systems, system drivers, browsers and games use C++ as their core language. This makes C++ one of the most popular languages today.

Since it is an enhanced/extended version of [C programming](https://www.programiz.com/c-programming) language, C and C++ are often denoted together as C/C++.

### History of C++

While Bjarne Stroustrup was working in AT&T Bell Labs in 1979, he faced difficulties in analyzing UNIX kernel for distributed systems. The current languages were either too slow or too low level. So, he set forward to create a new language.

For building this language, he chose C. Why C? Because it is a general purpose language and is very efficient as well as fast in its operations.

He used his knowledge of object-oriented model from SIMULA and began working on class extensions to C. His aim was to create a language with far higher level of abstraction while retaining the efficiency of C.

This new programming language was named C withClasses, but was later renamed to C++ (++ refers to the increment operator in C).

Briefly :

#### C++98, C++03, C++11 (C++0x), C++14 (C++1y), C++17(C++1z)

### Features of C++

Being a general-purpose language, C++ is undoubtedly feature-rich. Going through all the features will take you some time but, as a beginner, below are the most important features you should know.

1. **C++ is fast**  
     
   Since, C++ is an extended version of C, the C part of it is very low level.  
     
   This offers a huge boost in speed that high level languages like Python, Java don’t give you.
2. **C++  is statically typed**  
     
   C++ is a statically typed programming language.  
     
   In simple terms, C++ doesn’t allow the compiler to make assumptions about the type of data e.g. 10 is different from “10” and you have to let C++ know which one you are talking about.This helps the compiler catch errors and bugs before execution of the program.
3. **C++ is a multi-paradigm programming language**  
     
   C++ supports at least 7 different styles of programming and gives developers the freedom to choose one at their will.You can choose the programming style that fits your use case.
4. **Object oriented programming with C++**  
     
   Object oriented programming helps you solve a complex problem intuitively.With its use in C++, you are able to divide these complex problems into smaller sets by creating objects.
5. **Power of standard library (Standard template library - STL)**  
     
   The power of C++ extends with the use of standard libraries contained in it.These libraries contain efficient algorithms that you use extensively while coding.

## 5 Reasons Why you should learn C++?

This is one of those questions you need to ask before starting any programming language. It helps you understand the scope of the language, the real world usability and how far you can get with it in terms of support. Here are 5 reasons why you should learn C++.

1. **C++ is irreplaceable**  
     
   With the use of C++ in development of modern games, operating systems, browsers, and much more, it is safe to say that C++ is irreplaceable.
2. **You learn the internal architecture of a computer**  
     
   Since, C++ is a middle level language, you will write code that interacts directly with the internal hardware of the computer.You’ll learn how the computer memory really works, how information is stored in them, how you can retrieve them and so on.
3. **Over 600,000 C++ repositories on Github**  
     
   Github, the leading open source collaboration platform, has over 600,000 repositories for C++ alone.This metric itself proves the worth of C++ in the open source community as well.And, You can always create your own.
4. **60% StackOverflow Answer rate and active community**  
     
   Likewise, with over 400,000 C++ questions asked on StackOverflow, the number one Q&A platform for developers, more than 60% questions have been answered.The number of questions asked and the percentage of them answered shows the interest and active support for C++ today.  
     
   So, you can expect many great developers to help you solve real-life problems using C++.
5. **C++ job opportunities and salary**  
     
   C++ developers can expect an [average of yearly $100,000 salary](https://gooroo.io/analytics/skill/Cpp/#.V8-qYfl96Uk) with over 7,700 jobs advertised every month.The requirement of jobs comes mostly from game development, rendering engines and the windows applications.

## 4 Things to Know Before you Code in C++

Now that you know what C++ is and how vast its scope ranges to, it’s time to get started with it.

But, before you start, there are a couple of important things you should know.

Below are the 4 most important things you need to know.

1. **C++ cannot be learnt in a day**  
     
   Learning any language takes time and that holds even more truth for C++.If you are here to learn C++ in a day, then you’re going to end up facing failure.  
     
   To be honest, there’s no definite time to complete learning C++ and someone who says they can, are simply lying.  
     
   You only start learning with regular practice and dedication. So, I suggest you to invest valuable time learning C++.
2. **Learning C++ can be hard.**  
     
   Since it’s not a high level language, learning C++ can get overwhelming when you start and you’d need to be prepared to put thoughtful hours to learn the basics.  
     
   But, there’s no need to panic.  
     
   We offer plenty of resources and easy C++ tutorials available on Programiz to get started for beginners. Also, there are numerous support communities that will help you when you are stuck.
3. **No, you don’t need to learn C before C++**  
     
   People have different theories whether one should learn C before C++ or not. If you ask me, it isn’t a must. You can easily start with C++ and that’s what I did myself.If you already know C, you will have a head start in learning C++ as they have similar attributes like syntax and semantics.
4. **Don’t wait for the next C++ release**  
     
   Since, a new iteration of C++ is due late 2017, a lot of people ask whether it would be better it they wait until the next release before learning C++ or not.  
     
   The answer is **NO**.  
     
   Though there are a lot of additions and improvements planned for the next releases, the core principles are the same. So, it would be wise to invest your time now.

## Compile and run C++ programming on your OS

C++ is completely free and readily available on all platforms.

Follow the tutorial below for running C++ on your computer.

There are multiple compilers and text editors you can use to run C++ programming. These may differ from system to system.

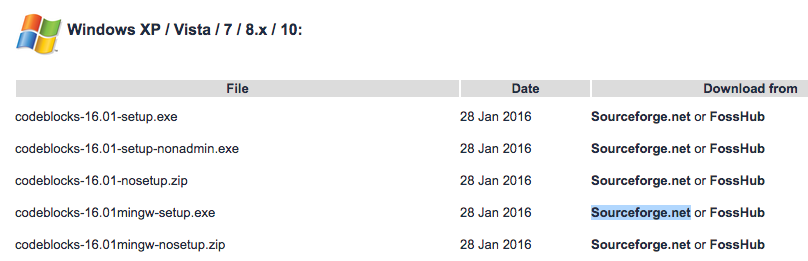
If you want a quick start, you can also run C++ program online.

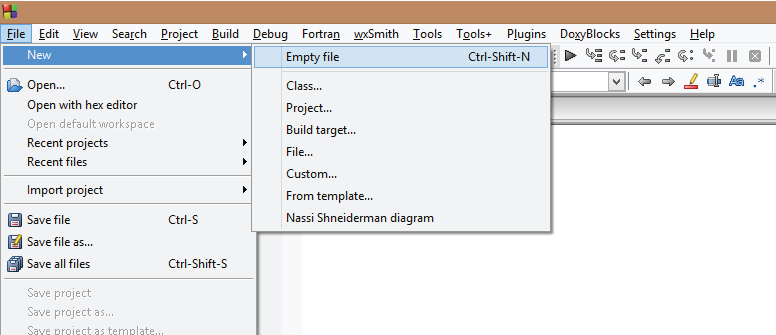
To run C++ Programming in Windows, you’d need to download Code::Blocks.

There are others available as well but Code::Blocks makes installation a piece of cake.

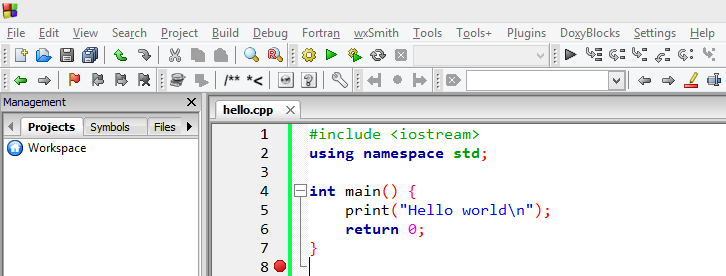
It’s easy, simple and developer friendly.

To make this procedure even easier, follow this step by step guide.

1. Go to the binary release download page of Code:Blocks official site.  
   
2. Under **Windows XP / Vista / 7 / 8.x / 10**section, click the link with **mingw-setup**highlighted row either from Sourceforge.net or FossHub.
3. Open the Code::Blocks Setup file and follow the instructions (**Next**> **I agree** > **Next** > **Install**); you don’t need to change anything. This installs the Code::Blocks with gnu gcc compiler, which is the best compiler to start with for beginners.
4. Now, open Code::Blocks and go to **File > New > Empty file**(Shortcut: **Ctrl+Shift+N**)



1. Write the C++ code and save the file with **.cpp** extension. To save the file, go to **File > Save**(Shortcut: **Ctrl+S**). **Important**: The filename should end with .cpp extension, like: hello**.cpp,**your-program-name**.cpp**

****

1. To run the program, go to **Build**> **Build and Run**(Shortcut: **F9**). This will build the executable file and run it.

If your program doesn’t run and if you see error message "can't find compiler executable in your search path(GNU GCC compiler)",  go to **Settings > Compiler > Toolchain executables**  and click **Auto-detect**. This should solve the issue in most cases

## Your first C++ program

Now you have installed the compilerbased on your OS, it’s time to write your first C++ program.

### “Hello World!”

Your first C++ program will be a “Hello World!” program.

You might have noticed “Hello World!” being the first program while starting out with any programming language.

This is because:

* It is a standard check to see whether everything is working fine or not.
* There will be very less code to start with.
* The less code makes it intuitive for the beginners to get acquainted with the language.
* The code is enough to learn the basic syntax and semantics of the language.

So, let’s get coding.

#include<iostream>

usingnamespace std;

int.main()

{

cout<<"Hello World!";

return0;

}

The program prints Hello World! in the output screen.

### How the program works?

Now, let’s dissect the above code. The code is divided into six major parts:

* #include <iostream>
* using namespace std
* ;
* int main() { }
* cout << “Hello World!”;
* return 0;

1. **What is #include <iostream>?**  
     
   If you’ve already written code in C language before, you might seen this line of code before. If you haven’t, don’t worry we’ll cover it now.  
     
   This statement includes the header file into the application so that you are able to use the operations included in them. Also, you can create your own header files and include them in your program using the #include.  
     
   **What is iostream?**

iostream is what you call the header file. It is a standard C++ input/output library file. It comes packaged with the compiler/IDE and contain mechanisms to get the information from the user and print same or added information to a file, screen or any other media.

**What is #include?**

The #include iostream file, into the program. This ensures that now you’re able to use the operations, iostream operations (like: taking input from user, displaying output on the screen), in the program.

1. **What is using namespace std;”?**  
     
   The statement is intuitive in itself, you are “using” the “namespace” “std” in your file.  
   We use the namespace std to make it easier to reference operations included in that namespace.  
   If we hadn’t used the namespace, we’d have written **std::cout** instead of **cout**. This tells the compiler that every **cout** is actually **std::cout**.  
     
   **What’s a namespace?**  
     
   It’s a region where your code resides. It limits or expands the scope of your code to one or more files.  
     
   **Why do you use namespace?**  
     
   Like two persons can have the same name, variables and functions in C++ can have same names as well. The use of namespace is to avoid the confusion of which variables/functions you are referencing to.  
     
   **What is std?**  
     
   std is a standard namespace used in C++.
2. **Semicolon ”;”**  
     
   Ask any C++ programmer and they will tell you at least one horror story related to the semicolon ; .  
   The semicolon is a terminal. It terminates a statement. When missed or incorrectly used, it will cause a lot of issues.
3. **int main() { }**  
     
   As the name suggests, it is the main function of the program. The code inside { } is called the body and is executed first when you run your C++ program.  
     
   It is one code that is mandatory in a C++ program. If you just have this line of code alone, your program will be valid.
4. **cout << “Hello World!”;**  
     
   This statement prints “Hello World!” onto the output screen.  
     
   The cout is an object of standard output stream. What this means is, it outputs/prints the data after *<<* , i.e. Hello World! into a stream (in this case, the output screen).  
     
   **What is a stream?**  
     
   Stream is basically a sequence of objects, usually bytes. It can describe files, input/output terminal, sockets, etc.  
   **What is <<?**  
     
   << is the insertion operator used to write formatted data into the stream.
5. **What is return 0;?**  
     
   This statement returns 0 ‘zero’.  
   This is called a return statement. It isn’t mandatory to return anything from the main() function but is rather a convention. If not return, the compiler returns a status automatically.  
     
   **Why zero in return statement?**  
     
   It denotes Exit status of the application that basically the tells system “The program worked fine.”

## 5 Tips to better yourself in C++ Programming

As you consume more content on C++, you’ll gradually understand there is no one way to get something done.

1. **Learning by doing**

Whether you follow our tutorials or from any other media, practice what you’ve just gone through. Only practice will make you a better programmer.  
  
Don’t just copy code and run it. Take some time to think what the code actually does. Replicate it on your system and see what errors occur and most importantly, learn from them.  
  
Only then will you better yourself as a developer.

1. **Follow C++ standards**

It is really important to follow a good C++ standard when you start programming. This provides a set of easy rules to follow for a particular purpose in a particular setting. For beginners, it gives you one less freedom when you start out.  
  
Take a look at [IsoCPP’s coding standard FAQ](https://isocpp.org/wiki/faq/coding-standards) and also follow the [C++ Core Guidelines](http://isocpp.github.io/CppCoreGuidelines/CppCoreGuidelines).

1. **Read others C++ code**

Join [Github’s open source projects](https://github.com/trending/cpp) and read others code. This can be overwhelming at first when you see all the code in the project. You can use [Code Whittling](http://patrick.lioi.net/2013/01/03/code-whittling/) to start small and only focus on one thing at a moment.  
  
You’ll not only learn other’s style of coding but you’ll also understand how they think.

1. **Break things**

Don’t be afraid to break things the way it is. You’ll be amazed to find how much you can learn from the broken code filled with errors.  
  
Errors are developers best friend. Understand what the error is about. Follow the error trail that takes you to the root of the issue, fix them and learn from them.

1. **Join C++ communities**

Get help from others. There are tons of great C++ communities that will help you solve real life problems and most importantly, become a better developer.  
  
Some of them are:

* + [StackOverflow](http://stackoverflow.com/questions/tagged/c%2b%2b) - Most Popular programming Q&A site on the web
  + [Codechef](https://www.codechef.com/) - Practice questions, challenges and a large community of programmers
  + [CodeProject](http://www.codeproject.com/) - For those who code, with in-depth articles and huge community of coders

## Final Words

C++ is a great language to start your programming journey with. The experience will take you a long way in becoming a great developer. So why wait. Get started by visiting any of the tutorials below.