

# Anatomy of a "Hello World!" Program

Let's get our hands dirty on a "Hello World!" program in C++.

## We'll cover the following



- “Hello World!” program
- Explanation
  - Preprocessor directives
  - White space
  - Namespace
  - main() function
  - Display text on the console
  - “Hello, World!”
  - <<
  - cout
  - ;

## “Hello World!” program #

Here is the source code for a program that prints “Hello World!” on the screen. First, see the program, and then we will discuss every component of the program in detail.

Run the code below and see the output!

```
#include <iostream>

using namespace std;

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



## Explanation #

When we run the code above, it prints “Hello World!” on the screen. This means we can modify the code given above to print anything on the screen. Sounds interesting!

But first, let’s understand the meaning of each line in a “Hello World!” program.

## Preprocessor directives #

**Line No. 1:** Lines that start with `#` are known as **preprocessor directives**.

Preprocessor directives tell the compiler to preprocess some information before starting the compilation. The **Header file** contains the declarations of predefined functions in C++.

`#include` is a preprocessor directive. It tells the preprocessor to add the content of the `iostream` header file in our source code before compilation. `iostream` file enables the user to take input from the keyboard and display output on the console.



## White space #

**Line No. 2:** Adds one line space. C++ compiler ignores the white spaces



## Namespace #

**Line No. 3:** This line allows the use of functions and variables from the standard library.

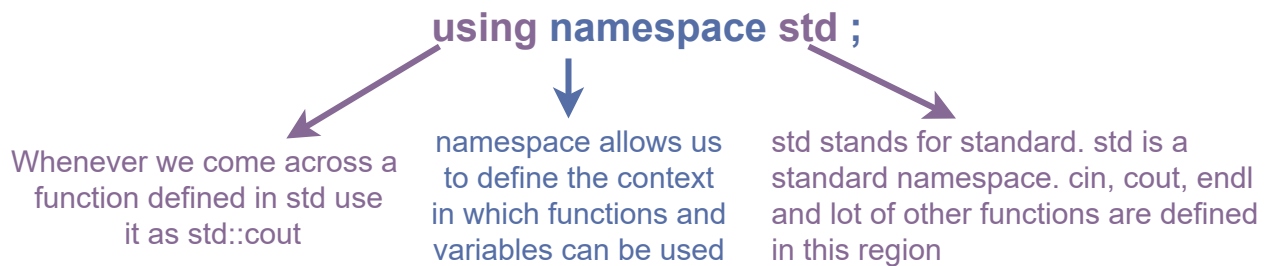
Suppose you have two employees named **John** in your office. You must need some extra context along with their first names to differentiate between these two.

Similarly, in the C++ program, you might define a variable `number`, and the same

variable `number` also exists in the external library. In order to use the variable and


functions from the external library, the compiler needs some extra context. Here, namespace comes in handy!

A **namespace** is a region that defines the context in which variables and functions can be used. All the elements of the C++ standard library are declared within a region called an `std`.



## `main()` function #

`main( )` is the function. It is the point from where all the C++ program starts its execution. Whenever the C++ program is executed, the operating system gives control to the `main` function. Therefore, every program in C++ must have a `main` function.


 **Function definition:** In computer language, a **function** is a block of code that performs a particular task, and it is given a name. Functions in programming are just like mathematical functions. They take something as input, perform some operation on it, and return something in the output.

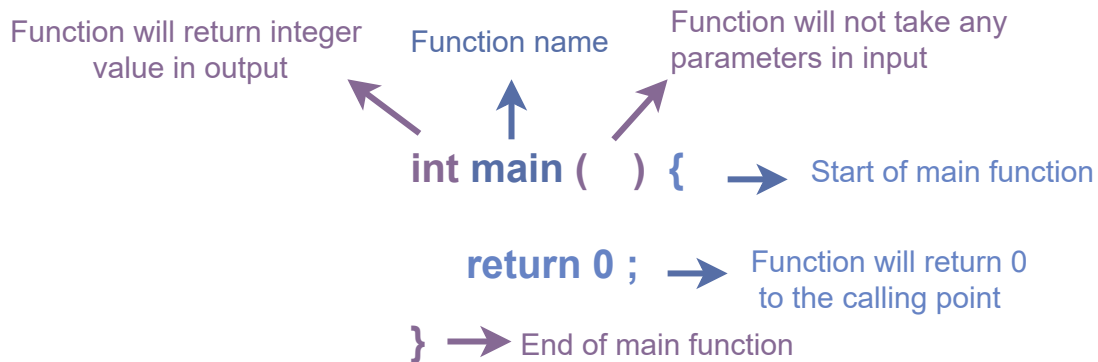
**Line No. 5:** `int` specifies that the `main` function will return an integer value in the output. `{` indicates the beginning of the `main` function.

**Line No. 7:** `return 0` returns 0 to the calling point on the successful execution of the program. It is a signal to the operating system that everything went great.

 **Note:** Adding a `return 0` statement in a program is not mandatory.

**Line No. 8:** `}` indicates the end of the `main` function.

 **Note:** {} The two curly braces represent the start and end of the main function. Everything written inside the curly braces is what the function does when it is called.



## Display text on the console #

**Line No. 6:** We can use `cout` along with insertion operator `<<` to display text on the console.

### "Hello, World!" #

In C++, we write our content inside the double-quotes. Anything written inside double quotes is known as a `string`. Here, `Hello, World!` is a `string`.

`<< #`

`<<` is called the insertion or output operator. It takes the content written on its right-hand side and inserts it into the `cout`.

`cout #`

`cout` knows that it should print everything on the console that is sent via an insertion operator.

`;` #

A statement is a command that the programmer gives to the computer. Here, **Line No.6** is a statement. It instructs the machine to display **Hello, World!** on the console. Every statement in the C++ program ends with a semicolon, which indicates the end of the current statement, and the next one is ready to execute.

---

In the next lesson, you will see how to compile the C++ program.

