## What is Java?

Java is a **programming language** and a **platform**. Java is a high level, object-oriented and secure programming language.

Java was developed by Sun Microsystems (which is now the subsidiary of Oracle) in the year 1995. James Gosling along with his team has introduced Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.

**Platform**: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

**class** Hello{

**public** **static** **void** main(String args[]){

     System.out.println("Hello world");

    }

}

* **class** keyword is used to declare a class in Java.
* **public** keyword is an access modifier that represents visibility.
* **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is no need to create an object to invoke the static method. The main() method is executed by the JVM, so it doesn't require creating an object to invoke the main() method. So, it saves memory.
* **void** is the return type of the method. It means it doesn't return any value.
* **main** represents the starting point of the program.
* **String[] args** or **String args[]** is used for [command line argument](https://www.javatpoint.com/command-line-argument)
* **System.out.println()** is used to print statement. Here, System is a class, out is an object of the PrintStream class, println() is a method of the PrintStream class.

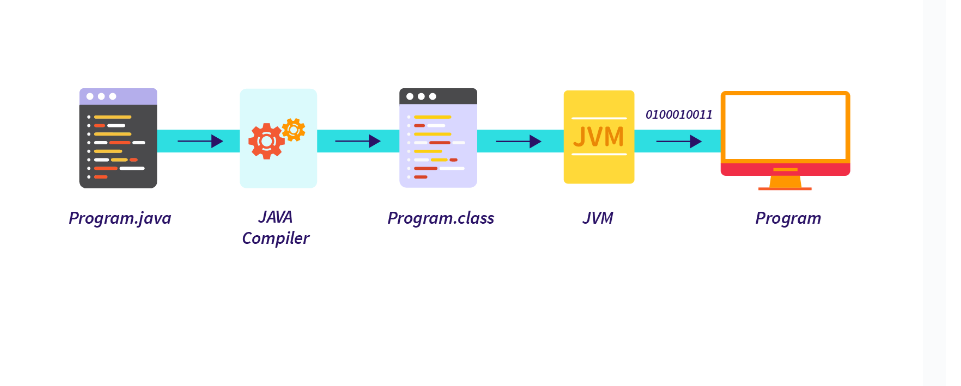
To compile :

javac Hello.java

To Excecute :

java Hello

Give the same class name and file name.



Main method:

The main() is the starting point for JVM to start execution of a Java program. Without the main() method, JVM will not execute the program

Eg: Main

* The no of .class files generated after the compilation will be equal to the of classes in the program.
* In a program if there are 2 classes, then two .class files will be generated after the compilation.

Eg: Main\_Without\_String\_Args

The program will compile, but we can’t run the program. because JVM will not recognize the main() method. Remember JVM always looks for the main() method with a string type array as a parameter.

Eg: Program\_Without\_Main\_Method\_With\_Static\_Block

we can [execute a java program](https://www.tutorialspoint.com/How-to-run-a-java-program) without a main method by using a static block (works until Java 1.6 version). Java 7 and newer versions don’t allow this.

In jdk 17 Program with static block and without main method will compile and .class will be generated but cannot be executed.

Program without static block and main method will not even compile ( “.class” file will not be generated. As it is generated only after compilation)

// we will learn static block further. For now just understand the above thing.

// we will learn method overloading further. For now just understand the below thing.

## Different Ways of Method Overloading in Java

* Changing the Number of Parameters.
* Changing Data Types of the Arguments.
* Changing the Order of the Parameters of Methods

main method can be overloaded but jvm takes only the one with regular format

(public static void main(String []args)

Eg: Main\_Method\_Overloading

Default prototype

**public** **static** **void** main(String[] args)

### Order of Modifiers

The static and public modifiers in the main() method declaration can be interchanged

**static** **public** **void** main(String[] args)

### Variants of String array arguments

In Java, it is possible to place the square brackets at different positions

**public** **static** **void** main(String[] args)

**public** **static** **void** main(String []args)

**public** **static** **void** main(String args[])

### Args or anything

 you can use any valid Java identifier instead of args. Identifier means a name

**public** **static** **void** main(String[] HelloWorld)

Command line arguments :

Eg: Command\_Line\_Arguments \_1

Eg: Command\_Line\_Arguments\_2

Command line Arguments starts from 0 index number. And arguments we are passing are treated as string type data by default.

Eg: Command\_Line\_Arguments\_3

// program flow

1. First compiler will do type checking whether the arguments given is of String type even if the arguments given are of boolean type data, number type, character type data they are treated as String only.
2. Here in args array at location 1 we stored a string “day”, and during the runtime we have given the argument

Java

1. Now the jvm shouts “hey I have already allocated space for “java” now I can’t allocate space for day in program
2. So jvm throws ArrayIndexOutOfBounds Exception

Give the command line arguments only at run time. command line arguments can’t be given directly in the code

Eg: Command\_Line\_Arguments\_4

// if the command line arguments are given without double quotes, and space is given between each word , then for each word index is counted .

Syntax : java filename hello world how are you

0 1 2 3 4

If the command line arguments are given with double quotes then the whole sentence is treated as a single index

Syntax:

java filename “hello world how are you” “ world : ni sangati nv chusko bey”

1. 1

Eg: Command\_Line\_Arguments\_Eg5

args is the array and since array you can use length property on it. and args is an array of string, so data present in array is always a string

we can apply equals method on string objects so there is no compile time error.

here we have given command line arguments only at 0th index but tried to access 1st index so array index out of bounds.

Eg: Command\_Line\_Arguments\_Eg6

// go through the code

Eg: Command\_Line\_Arguments\_Eg7

x is intialized in the if block , there is no guarantee that block would be executed by compiler , so it will throw compile time error . if there is no else block

Here the compiler understands since x is also initialized in else ,incase if block does not executes ,else block will execute compulsory and x will be initialized.

So there we be no compile time error.

Eg: Command\_Line\_Arguments\_8

// go through the code

### Var-args instead of String array

you can use a varargs parameter (denoted by three dots ... instead of square brackets ( [] ) as a replacement.

**public** **static** **void** main(String... args)

### The final Modifier String argument

ensuring that the parameter's value cannot be changed.

**public** **static** **void** main(**final** String[] args)

### Final main method

It is possible to declare the main() method with the final keyword.

**final** **public** **static** **void** main(String[] args)

### Synchronized Keyword to Static Main Method

**public** **static** **synchronized** **void** main(String[] args)

### strictfp keyword to the static main method

 strictfp keyword ensures consistent floating-point arithmetic across different platforms.

**final** **strictfp** **public** **static** **void** main(String[] args)

* There should be only one public class.
* Class name and file name can be different.
* Give file name for compilation.
* Class name during run time
* Don’t use public for class name in this program it will show error. If the class name and file name is different.

JVM executes the static block, then it executes static methods, and then it creates the object needed by the program. Finally, it executes the instance methods. JVM executes a static block on the highest priority basis. It means JVM first goes to static block even before it looks for the main() method in the program.

Eg: Excuetion\_Process

Eg: SemiColon\_Example

// check Documentation in program

Eg: Main\_Method

// only main method with regular signature is executed by jvm.