

# What is Java?

Java is a **programming language** and a **platform**. Java is a high level, robust, 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* is known as the father of 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.

## Java Example

```
class Demo{  
    public static void main(String args[]){  
        System.out.println("JAVA");  
    }  
}
```

## Application

According to Sun, 3 billion devices run Java. There are many devices where Java is currently used. Some of them are as follows:

1. Desktop Applications such as acrobat reader, media player, antivirus, etc.
2. Web Applications such as irctc.co.in, javatpoint.com, etc.
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games, etc.

## Types of Java Applications

There are mainly 4 types of applications that can be created using Java programming:

### *1) Standalone Application*

Standalone applications are also known as desktop applications or window-based applications. These are traditional software that we need to install on every machine. Examples of standalone application are Media player, antivirus, etc. AWT and Swing are used in Java for creating standalone applications.

## 2) Web Application

An application that runs on the server side and creates a dynamic page is called a web application. Currently, [Servlet](#), [JSP](#), [Struts](#), [Spring](#), [Hibernate](#), [JSF](#), etc. technologies are used for creating web applications in Java.

## 3) Enterprise Application

An application that is distributed in nature, such as banking applications, etc. is called an enterprise application. It has advantages like high-level security, load balancing, and clustering. In Java, [EJB](#) is used for creating enterprise applications.

## 4) Mobile Application

An application which is created for mobile devices is called a mobile application. Currently, Android and Java ME are used for creating mobile applications.

# Java Platforms / Editions

There are 4 platforms or editions of Java:

### 1) Java SE (Java Standard Edition)

It is a Java programming platform. It includes Java programming APIs such as java.lang, java.io, java.net, java.util, java.sql, java.math etc. It includes core topics like OOPs, [String](#), Regex, Exception, Inner classes, Multithreading, I/O Stream, Networking, AWT, Swing, Reflection, Collection, etc.

### 2) Java EE (Java Enterprise Edition)

It is an enterprise platform that is mainly used to develop web and enterprise applications. It is built on top of the Java SE platform. It includes topics like Servlet, JSP, Web Services, EJB, [JPA](#), etc.

### 3) Java ME (Java Micro Edition)

It is a micro platform that is dedicated to mobile applications.

### 4) JavaFX

It is used to develop rich internet applications. It uses a lightweight user interface API.

# Prerequisite

To learn Java, you must have the basic knowledge of C/C++ programming language.

# Audience

Our Java programming tutorial is designed to help beginners and professionals.

# Problem

We assure that you will not find any problem in this Java tutorial. However, if there is any mistake, please post the problem in the contact form.

## Java Version History

Many java versions have been released till now. The current stable release of Java is Java SE 10.

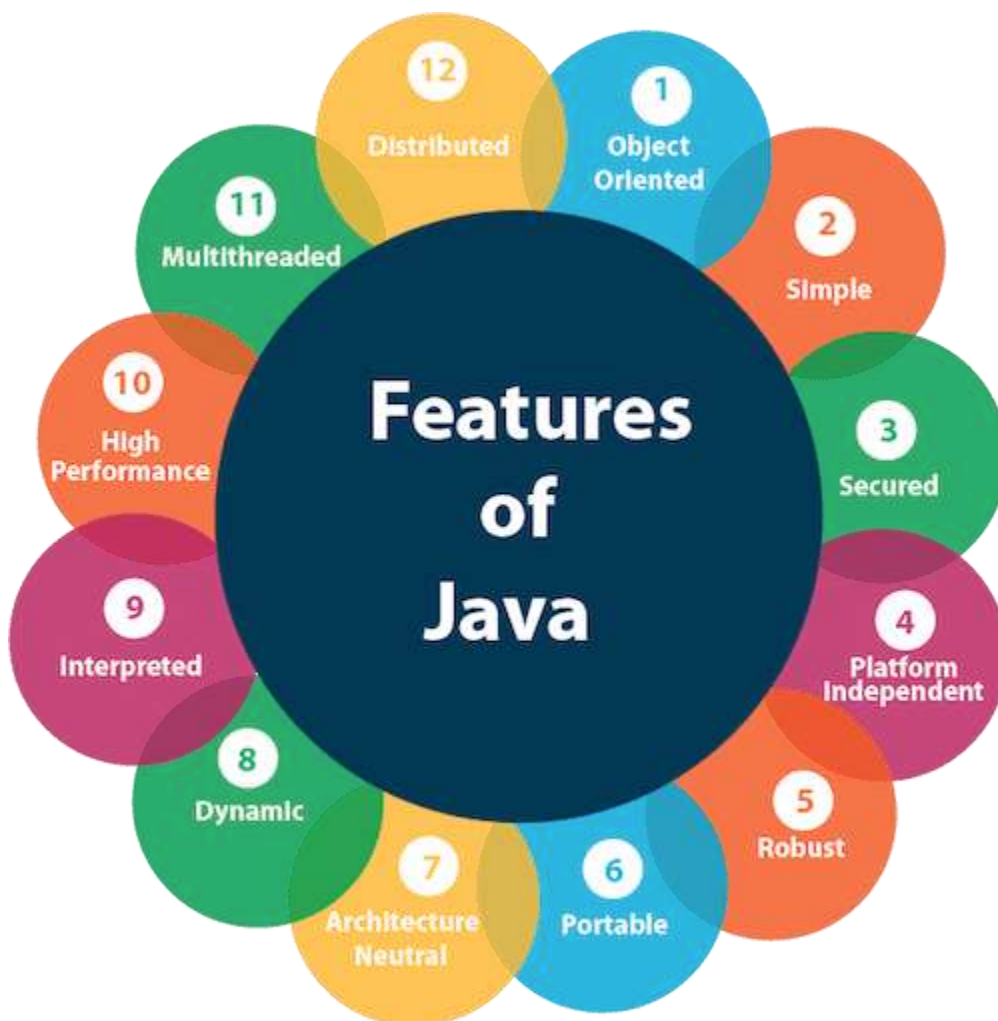
1. JDK Alpha and Beta (1995)
2. JDK 1.0 (23rd Jan 1996)
3. JDK 1.1 (19th Feb 1997)
4. J2SE 1.2 (8th Dec 1998)
5. J2SE 1.3 (8th May 2000)
6. J2SE 1.4 (6th Feb 2002)
7. J2SE 5.0 (30th Sep 2004)
8. Java SE 6 (11th Dec 2006)
9. Java SE 7 (28th July 2011)
10. Java SE 8 (18th Mar 2014)
11. Java SE 9 (21st Sep 2017)
12. Java SE 10 (20th Mar 2018)
13. Java SE 11 (September 2018)
14. Java SE 12 (March 2019)
15. Java SE 13 (September 2019)
16. Java SE 14 (Mar 2020)
17. Java SE 15 (September 2020)
18. Java SE 16 (Mar 2021)
19. Java SE 17 (September 2021)
20. Java SE 18 (to be released by March 2022)

Since Java SE 8 release, the Oracle corporation follows a pattern in which every even version is release in March month and an odd version released in September month.

# Features of Java

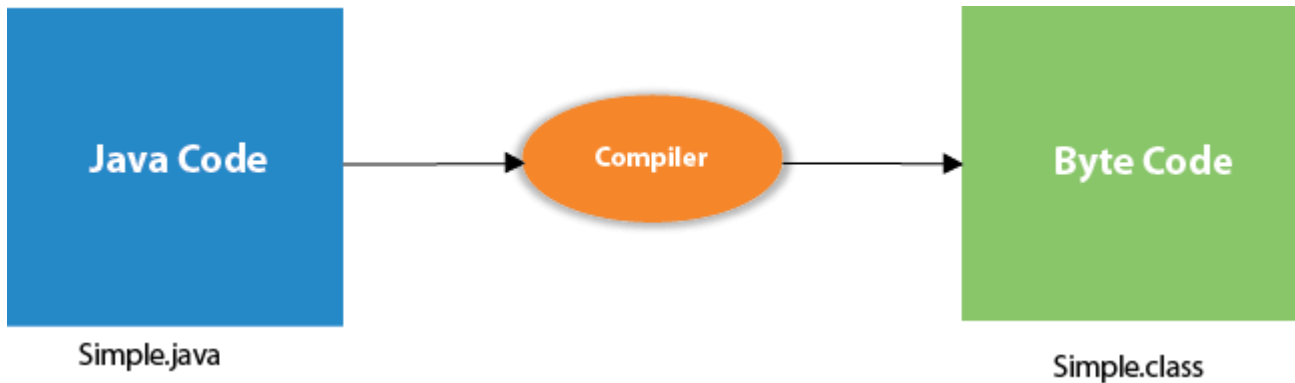
The primary objective of [Java programming](#) language creation was to make it portable, simple and secure programming language. Apart from this, there are also some excellent features which play an important role in the popularity of this language. The features of Java are also known as Java buzzwords.

A list of the most important features of the Java language is given below.



**Compilation Flow:**

When we compile Java program using javac tool, the Java compiler converts the source code into byte code.



## Parameters used in First Java Program

Let's see what is the meaning of class, public, static, void, main, String[], System.out.println().

- **class** keyword is used to declare a class in Java.
- **public** keyword is an access modifier that represents visibility. It means it is visible to all.
- **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**. We will discuss it in coming section.
- **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. We will discuss the internal working of **System.out.println()** statement in the coming section.

## In how many ways we can write a Java program?

There are many ways to write a Java program. The modifications that can be done in a Java program are given below:

**1) By changing the sequence of the modifiers, method prototype is not changed in Java.**

Let's see the simple code of the main method.

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

**2) The subscript notation in the Java array can be used after type, before the variable or after the variable.**

Let's see the different codes to write the main method.

1. **public static void** main(String[] args)
2. **public static void** main(String []args)
3. **public static void** main(String args[])

### 3) You can provide var-args support to the main() method by passing 3 ellipses (dots)

Let's see the simple code of using var-args in the main() method. We will learn about var-args later in the Java New Features chapter.

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

### 4) Having a semicolon at the end of class is optional in Java.

Let's see the simple code.

1. **class** A{
2. **static public void** main(String... args){
3. System.out.println("hello java4");
4. }
5. };

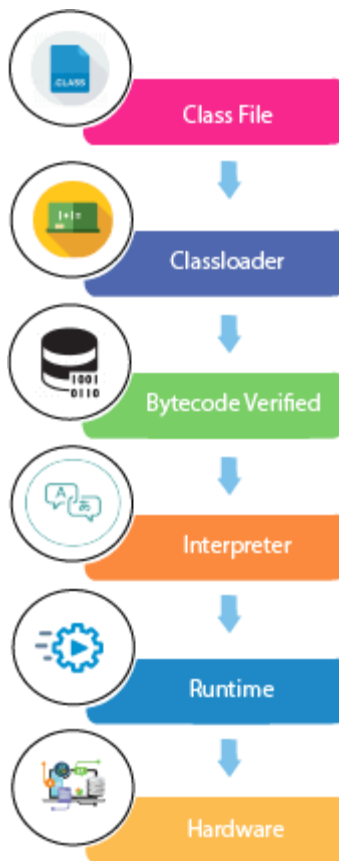
---

## Valid Java main() method signature

1. **public static void** main(String[] args)
2. **public static void** main(String []args)
3. **public static void** main(String args[])
4. **public static void** main(String... args)
5. **static public void** main(String[] args)
6. **public static final void** main(String[] args)
7. **final public static void** main(String[] args)
8. **final strictfp public static void** main(String[] args)

## What happens at runtime?

At runtime, the following steps are performed:



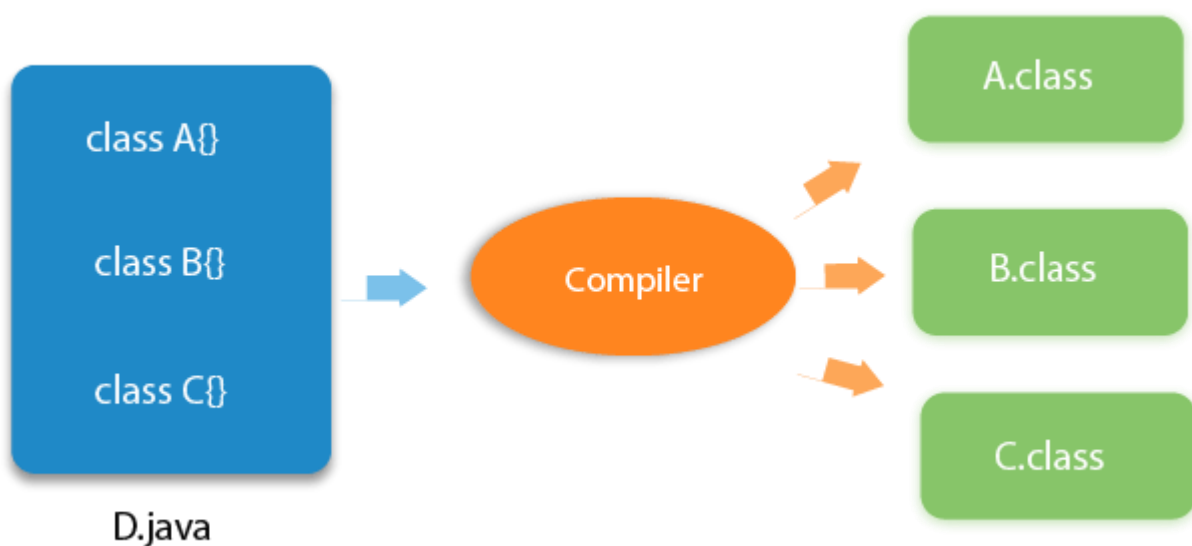
**ClassLoader:** It is the subsystem of JVM that is used to load class files.

**Bytecode Verifier:** Checks the code fragments for illegal code that can violate access rights to objects.

**Interpreter:** Read bytecode stream then execute the instructions.

**Q) Can you have multiple classes in a java source file?**

Yes, like the figure given below illustrates:



# Difference between JDK, JRE, and JVM

1. [A summary of JVM](#)
2. [Java Runtime Environment \(JRE\)](#)
3. [Java Development Kit \(JDK\)](#)

We must understand the differences between JDK, JRE, and JVM before proceeding further to [Java](#). See the brief overview of JVM here.

If you want to get the detailed knowledge of Java Virtual Machine, move to the next page. Firstly, let's see the differences between the JDK, JRE, and JVM.

---

## JVM

JVM (Java Virtual Machine) is an abstract machine. It is called a virtual machine because it doesn't physically exist. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode.

JVMs are available for many hardware and software platforms. JVM, JRE, and JDK are platform dependent because the configuration of each [OS](#) is different from each other. However, Java is platform independent. There are three notions of the JVM: *specification*, *implementation*, and *instance*.

The JVM performs the following main tasks:

- Loads code
- Verifies code
- Executes code
- Provides runtime environment

[More Details.](#)

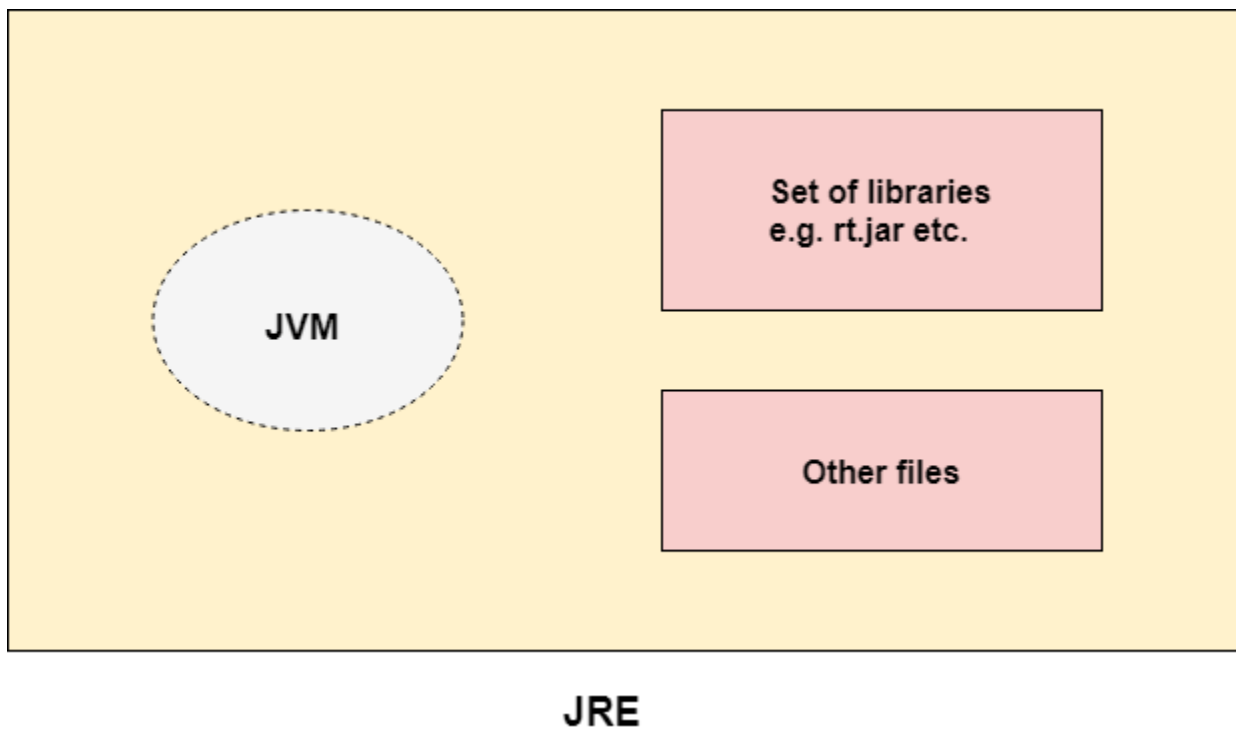
---

## JRE

JRE is an acronym for Java Runtime Environment. It is also written as Java RTE. The Java Runtime Environment is a set of software tools which are used for developing Java applications. It is used to provide the runtime environment. It is the implementation of JVM. It physically exists. It contains a set of libraries + other files that JVM uses at runtime.

The implementation of JVM is also actively released by other companies besides Sun Micro Systems.





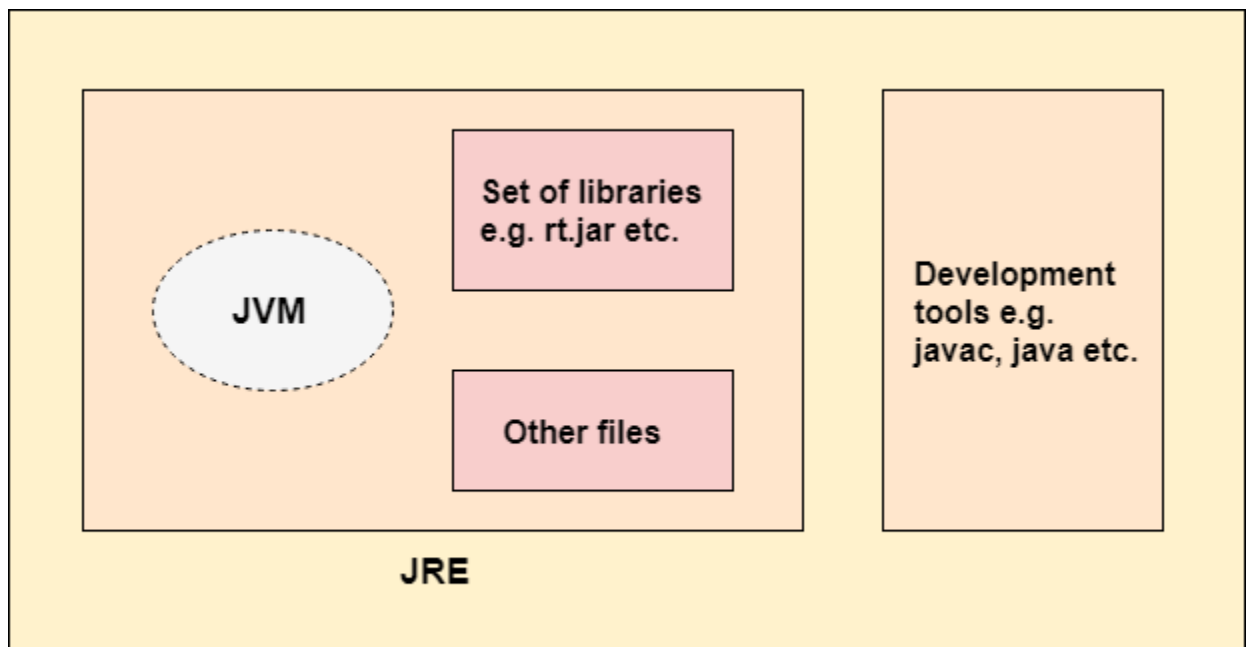
---

## JDK

JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop Java applications and [applets](#). It physically exists. It contains JRE + development tools.

JDK is an implementation of any one of the below given Java Platforms released by Oracle Corporation:

- Standard Edition Java Platform
- Enterprise Edition Java Platform
- Micro Edition Java Platform
- The JDK contains a private Java Virtual Machine (JVM) and a few other resources such as an interpreter/loader (java), a compiler (javac), an archiver (jar), a documentation generator (Javadoc), etc. to complete the development of a Java Application.



**JDK**

- 
-