# FAST

# National University of Computer and Emerging Sciences Peshawar

Lecture # 01

# Software Construction and Development (Java Programming)

**Instructor:** Muhammad Abdullah Orakzai

DEPARTMENT OF COMPUTER SCIENCE


الذی علم بالقلم۔ علم الانسان ما لم یعلم۔



# Introduction to Java

# Contents

# Java Introduction

- Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language.
- Java is an object oriented, class-based, concurrent, secured and general-purpose computer-programming language. It is a widely used robust technology.

**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.

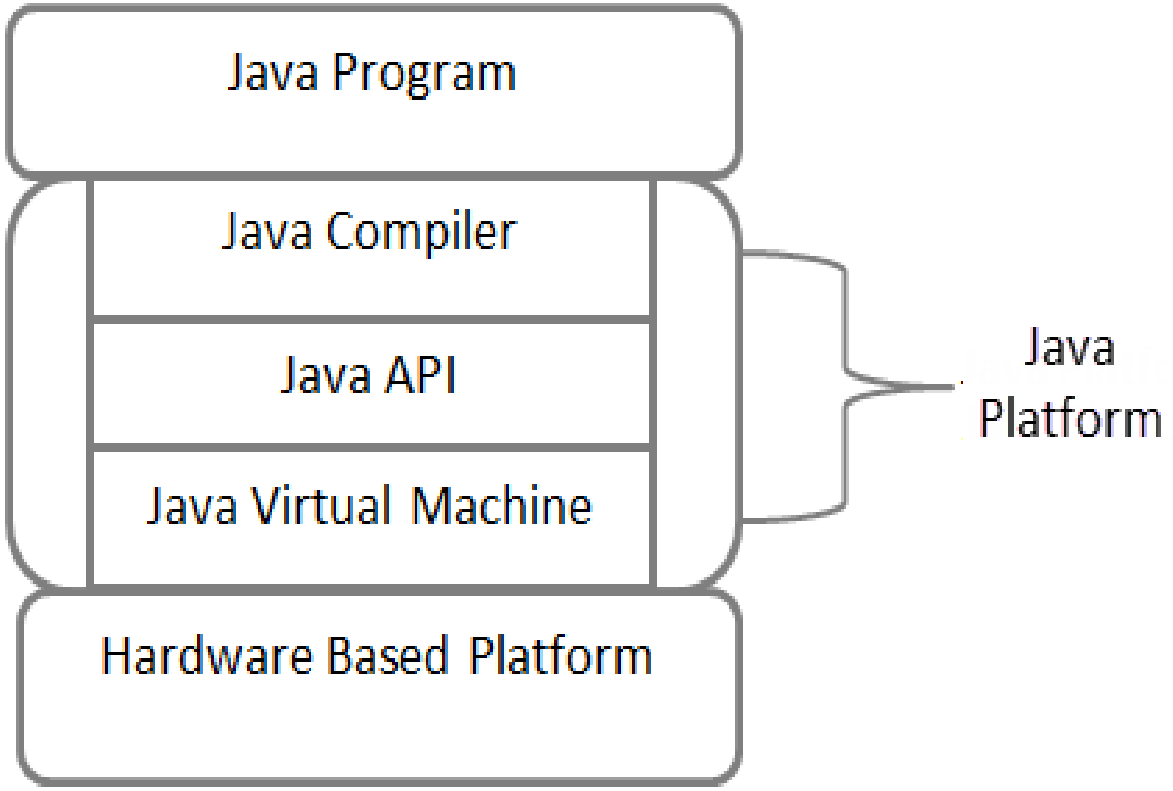
# Java API

- Java Application Programming Interface (API) is the area of Java development kit (JDK).
- An API includes classes, interfaces, packages and also their methods, fields, and constructors.
- All these built-in classes give benefits to the programmer.
- Only programmers understand how to apply these classes.

# Java API…

- A user interface offers the basic user interaction among user and computer, in the same manner, the API works as an application program interface which gives connection amongst the software as well as the consumer.
- API includes classes and packages which usually assist a programmer to minimize the lines of a program.

# Java API...


Java Program
Java Compiler
Java API
Java Virtual Machine
Hardware Based Platform
Java
Platform

# Short History

- 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 Oak name to Java.

# Java Application Types

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

- Desktop Applications such as acrobat reader, media player, antivirus, etc.
- Web Applications such as irctc.co.in, javatpoint.com, etc.
- Enterprise Applications such as banking applications.
- Mobile
- Embedded System
- 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.

**2) Web Application**

An application that runs on the server side and creates a dynamic page is called a web application.

# Types of Java Applications...

**3) Enterprise Application**

An application that is distributed in nature, such as banking applications, etc. is called enterprise application.

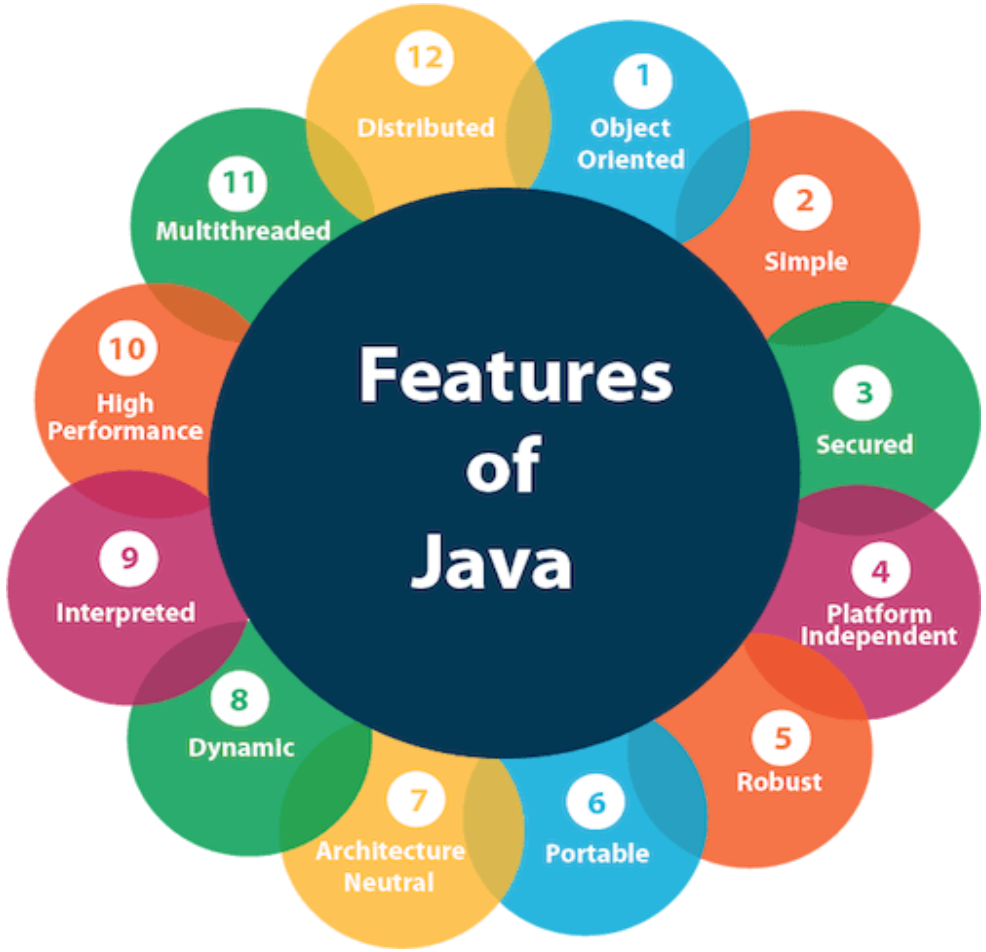
**4) Mobile Application**

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 features

- Simple
- Object-Oriented
- Portable
- Platform independent
- Secured
- High Performance
- Multithreaded
- Distributed
- Dynamic
- Robust  etc



# 1) Simple

Java is very easy to learn, and its syntax is simple, clean and easy to understand. According to Sun, Java language is a simple programming language because:

- Java syntax is based on C++ (so easier for programmers to learn it after C++).
- Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
- There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

# 2) Object-Oriented

- Java is an object-oriented programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.
- Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

# 3) Java is Platform Independent

- Java is platform independent because it is different from other languages like **C, C++,** etc. which are compiled into platform specific machines while Java is a write once, run anywhere language.
- A **platform** is the hardware or software environment in which a program runs.
- There are two types of platforms software-based and hardware-based.
- Java provides a software-based platform.
- The Java platform differs from most other platforms in the sense that it is a software-based platform that runs on the top of other hardware-based platforms. It has two components:

# 3) Java is Platform Independent…

**1)** Runtime Environment

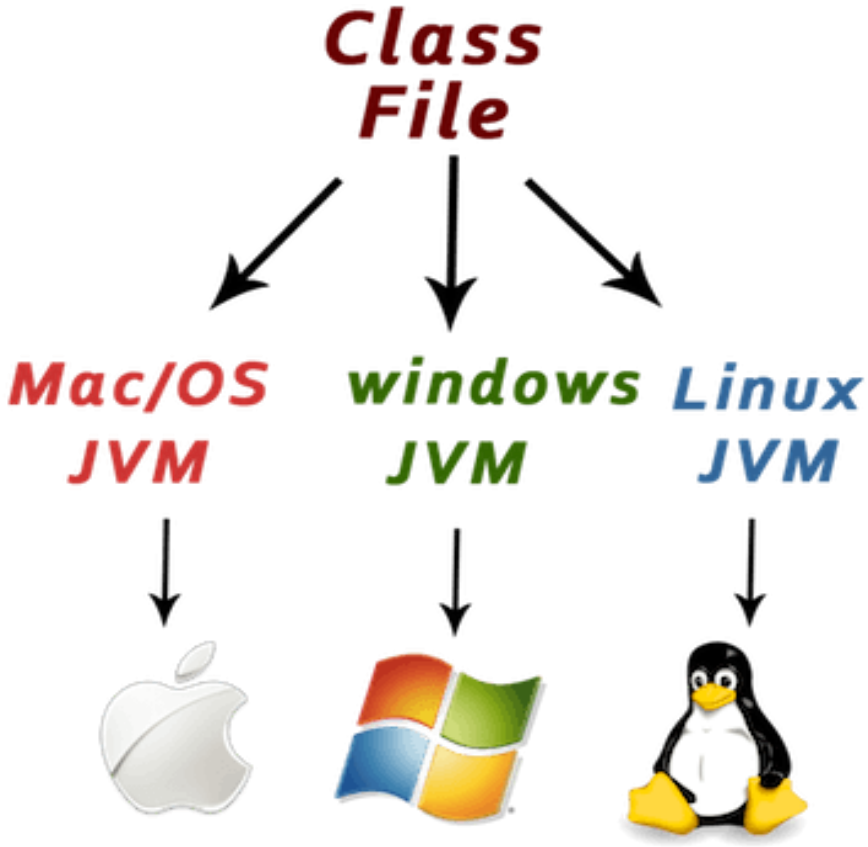
**2)** API (Application Programming Interface)

Java code can be run on multiple platforms, **for example**, Windows, Linux, Sun Solaris, Mac/OS, etc. Java code is compiled by the compiler and converted into bytecode or class file.

This bytecode is a platform-independent code because it can be run on multiple platforms,

**i.e.** Write Once and Run Anywhere(WORA).

# 3) Java is Platform Independent…



# 4) Secured

Java is best known for its security. With Java, we can develop virus-free systems.

Java is secured because:

**No explicit pointer**

**Java Programs run inside a virtual machine sandbox**

**Classloader:** Classloader in Java is a part of the Java Runtime Environment(JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.

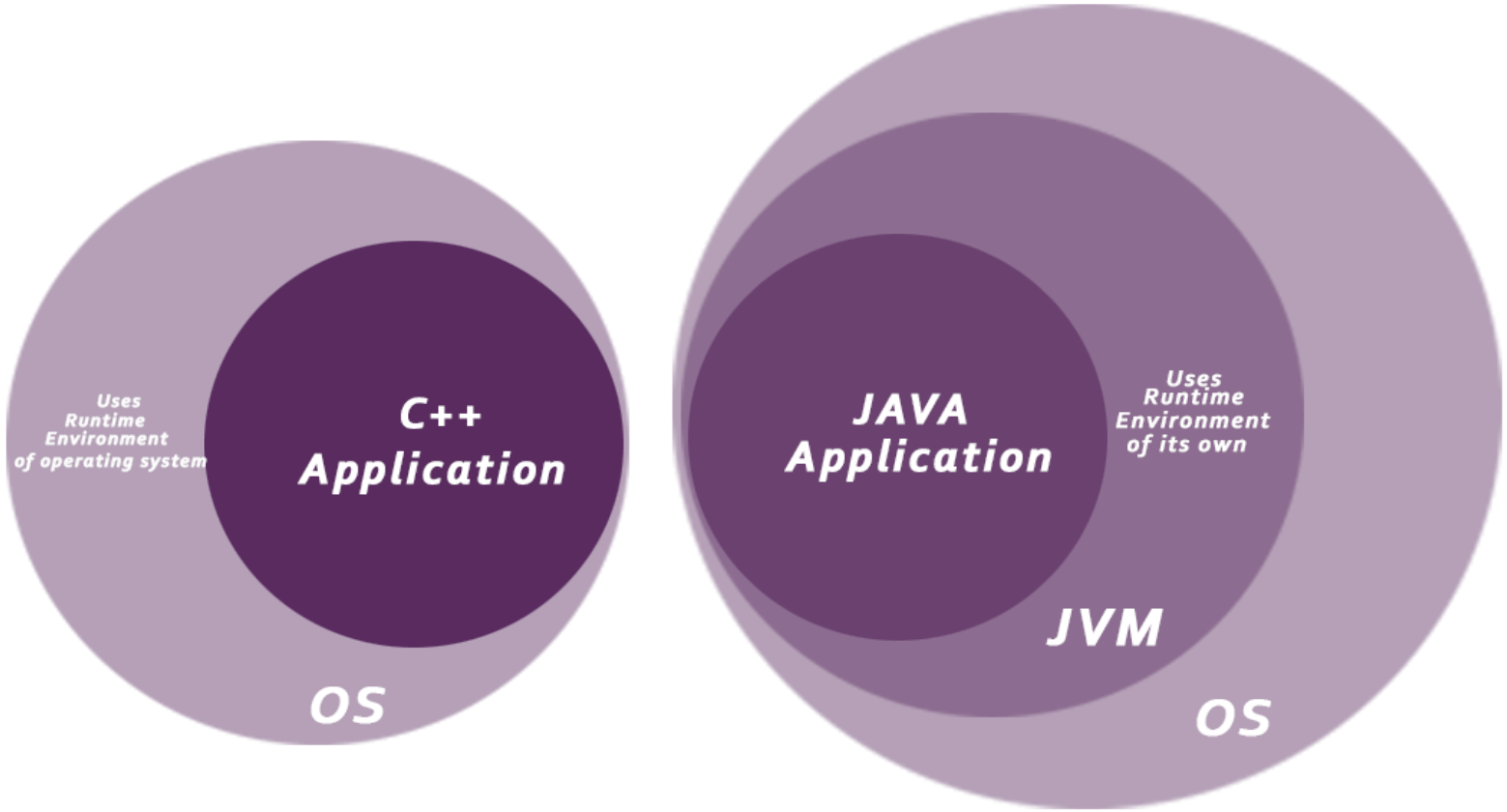
# 4) Secured...

**Bytecode Verifier:** It checks the code fragments for illegal code that can violate access right to objects.

**Security Manager:** It determines what resources a class can access such as reading and writing to the local disk.

Java language provides these securities by default. Some security can also be provided by an application developer explicitly through SSL, JAAS, Cryptography, etc.

# 4) Secured…


Uses
Runtime
Environment
of operating system
C++
Application
OS
JAVA
Application
Uses
Runtime
Environment
of its own
JVM
OS

# 5) Robust

Robust simply means strong. Java is robust because:

- It uses strong memory management.
- There is a lack of pointers that avoids security problems.
- There is automatic garbage collection in java which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.
- There are exception handling and the type checking mechanism in Java.

**Note:** All these points make Java robust.

# 6) Portable

- Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.
- “Write Once Run Anywhere” and for the most part this work.

# 7) High-performance

- Java is faster than other traditional interpreted programming languages because Java bytecode is "**close**" to native code.
- It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

# 8) Distributed

- Java is distributed because it facilitates users to create distributed applications in Java.
- RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

# 9) Multi-threaded

- A thread is like a separate program, executing concurrently.
- We can write Java programs that deal with many tasks at once by defining multiple threads.
- The main advantage of multi-threading is that it doesn't occupy memory for each thread.
- It shares a common memory area. Threads are important for multi-media, Web applications, etc.

# 10) Dynamic

- Java is a dynamic language. It supports dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++.
- Java supports dynamic compilation and automatic memory management (garbage collection).

# Object Oriented Programming (OOP)

OOP is a particular style of programming which involves a particular way of designing solutions to particular problems. Most modern programming languages, including Java, support this paradigm. When speaking about OOP one has to mention:

- Inheritance
- Modularity
- Polymorphism
- Encapsulation (binding code and its data)

# Tools to be Used for Java Development

- Notepad and cmd
- Eclipse
- NetBeans
- JDK+JRE
- DrJava

# How to set path in Java

The path is required to be set for using tools such as javac, java, etc.

If you are saving the Java source file inside the JDK/bin directory, the path is not required to be set because all the tools will be available in the current directory.

However, if you have your Java file outside the JDK/bin folder, it is necessary to set the path of JDK.

There are two ways to set the path in Java:

1. Temporary
2. Permanent

# First Java Program | Hello World Example

In this page, we will learn how to write the simple program of java. We can write a simple hello java program easily after installing the JDK.

To create a simple java program, you need to create a class that contains the main method. Let's understand the requirement first.

## The requirement for Java Hello World Example

- For executing any java program, you need to Install the JDK if you don't have installed it, download the JDK and install it.
- Set path of the jdk/bin directory. http://www.javatpoint.com/how-to-set-path-in-java
- Create the java program
- Compile and run the java program

# First Java Program | Hello World Example…

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

**To compile:** javac Simple.java

**To execute:** java Simple

# 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 which 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 to create an object to invoke the main method. So it saves memory.

# Parameters used in First Java Program

**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** is used for command line argument. Java main method accepts a single argument of type String array. We will learn it later.

**System.out.println()** is used to print statement. Here, **System** is a class, **out** is the object of PrintStream class, println() is the method of PrintStream class. We will learn about the internal working of System.out.println statement later.

# String args[]

```
public class Test {

public static void main(String[] args)

{

for(String s : args)

{

System.out.println(s);

}

 }

 }
```

# String args[]

Run this program from command line prompt:

javac Test.java

java Test 123 // one element passed to method

java Test 1 2 3 // three elements passed to method

java Test Ali Khan Asad Amin //four elements passed

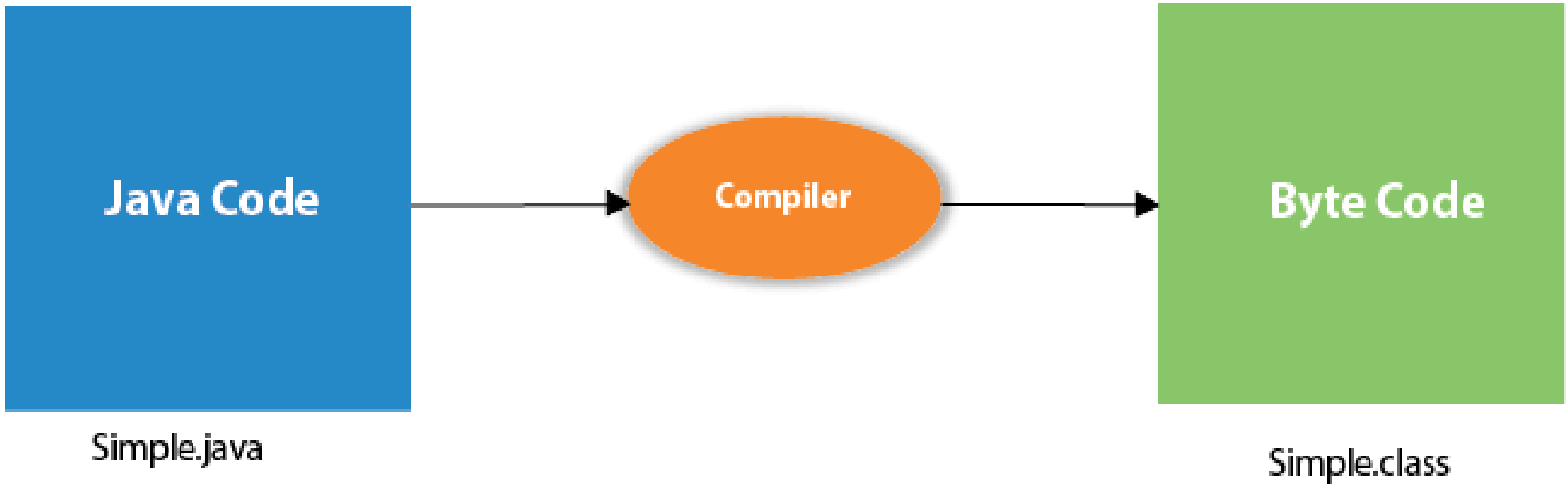
# Valid java main method signature

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

# First Java Program | Hello World Example…

**Compilation Flow:**

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

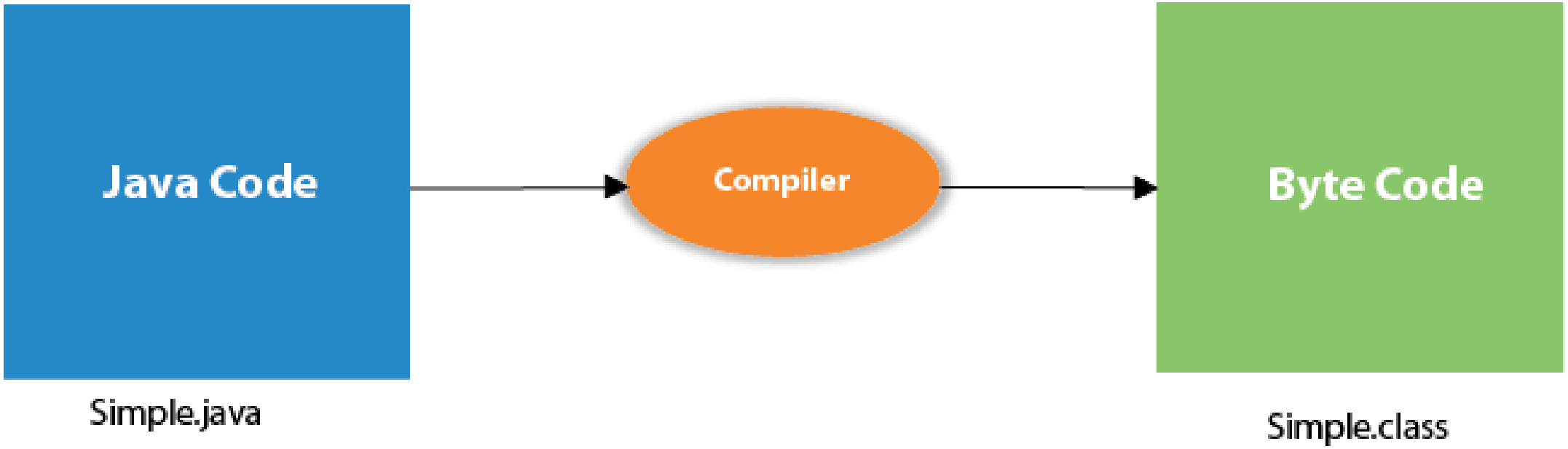


# Internal Details of Hello Java Program

In the previous page, we have learnt about the first program, how to compile and run the first java program. Here, we are going to learn, what happens while compiling and running the java program. Moreover, we will see some question based on the first program.

# What happens at compile time?

At compile time, java file is compiled by Java Compiler (It does not interact with OS) and converts the java code into bytecode.



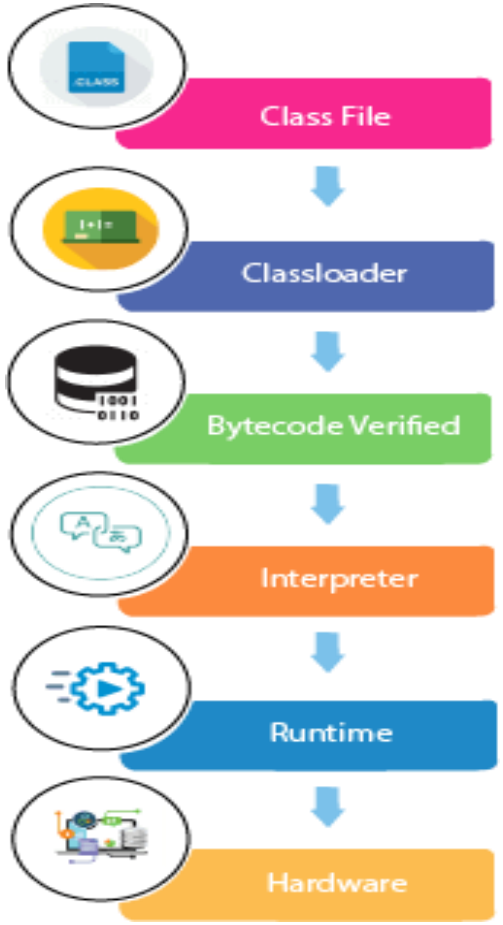
# What happens at runtime?

At runtime, following steps are performed:

**Classloader:** 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 right to objects.

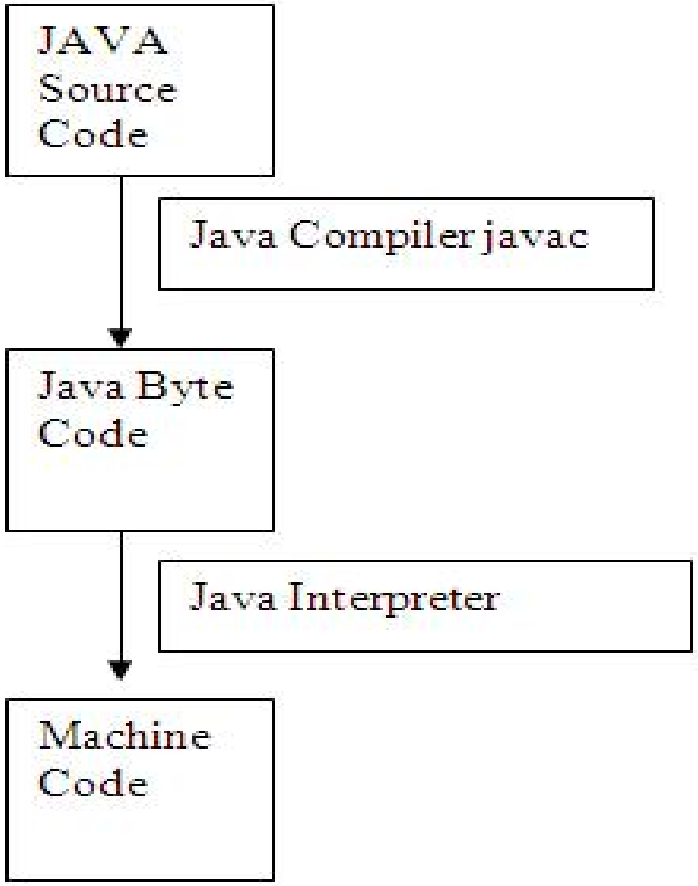
**Interpreter:** read bytecode stream then execute the instructions.



# Compilation and interpretation process

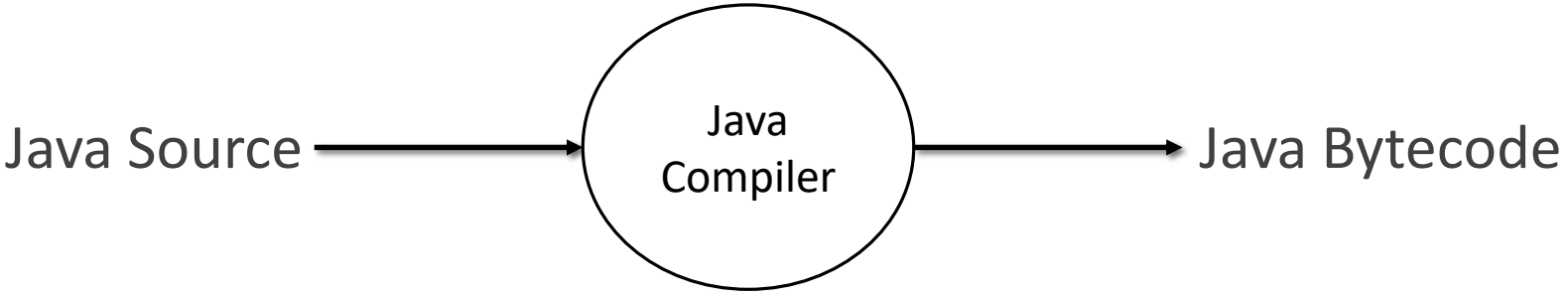

**prog.class:** Intermediate code or class file or bytecode.

# Compilation and interpretation process…



# Bytecode

- Java program (source code) are compiled into a form called java bytecode.
- The java reads java language source (.java) files, translates the source into java bytecodes, and places the bytecode into (.class) files.
- The compiler generates one class file for each class contained in java source file.



# 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*.

# JVM…

The JVM performs the following main tasks:

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

More Details.

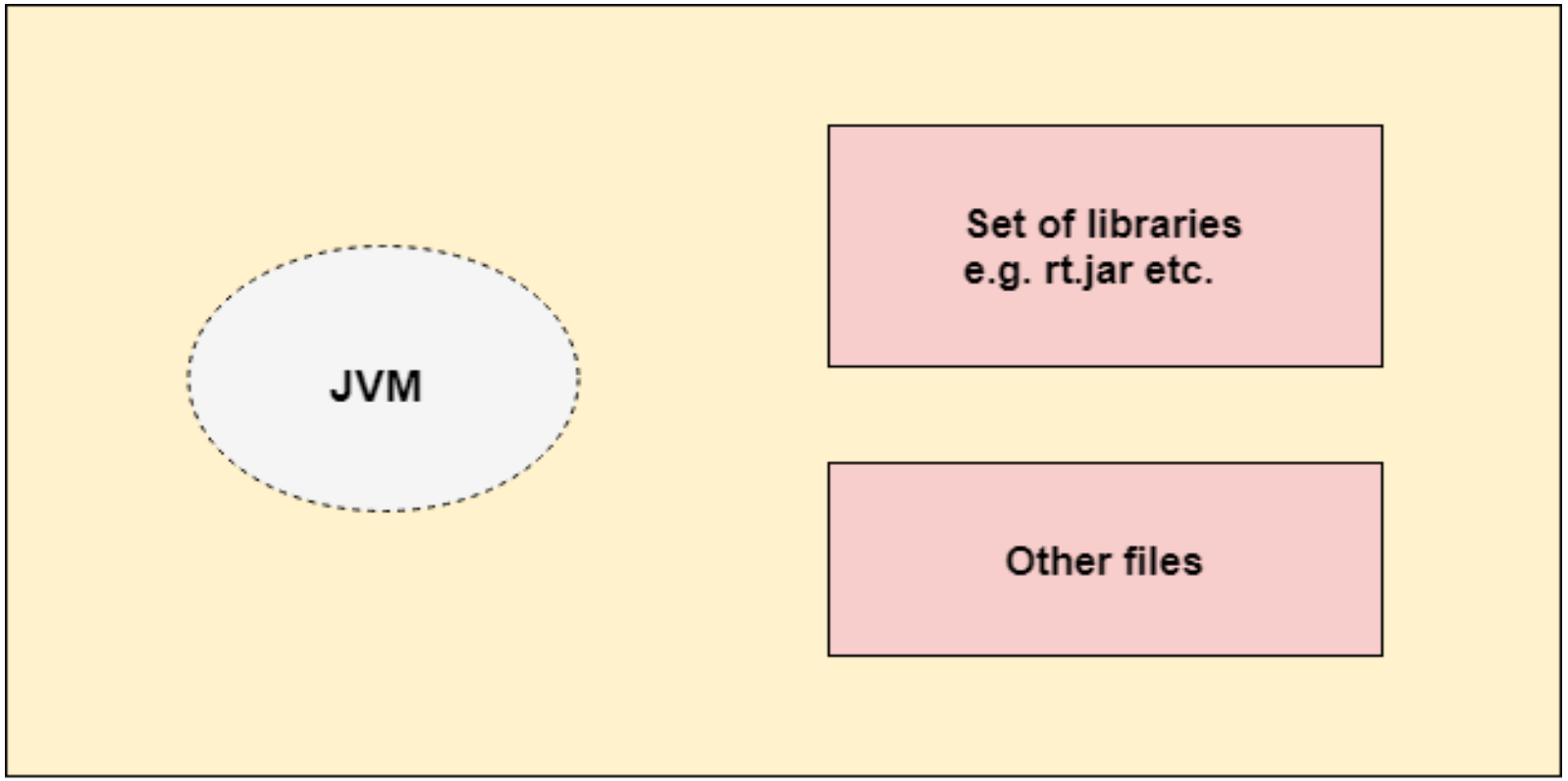
# JRE

- 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.

# JRE

- The implementation of JVM is also actively released by other companies besides Sun Micro Systems.
- The JRE consists of set of built in classes as well as JVM. Without an available JRE for given environment, it is impossible to run java software.

# JRE...



# JDK

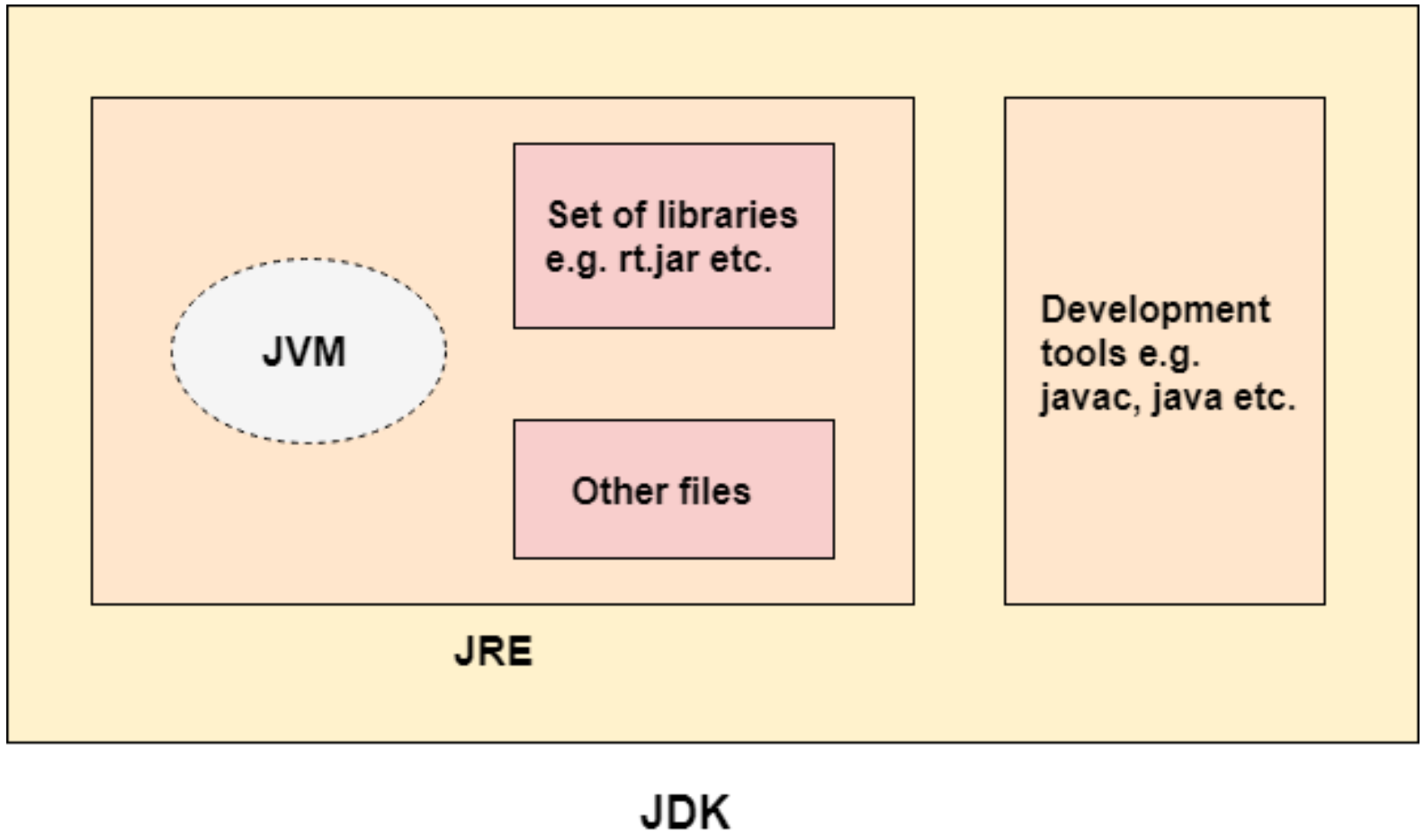
JDK is an acronym for Java Development Kit. The Java Development Kit (JDK) is a software development environment which is used to develop **Java applications** and **applets**. It physically exists. It contains JRE + development tools.

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...


Set of libraries
e.g. rt.jar etc.
JVM
Other files
Development
tools e.g.
javac, java etc.
JRE
JDK

# Java Applets

- An **applet** is a **Java** program that can be embedded into a web page. It runs inside the web browser and works at client side.

- An **applet** is embedded in an HTML page using the **APPLET** or **OBJECT tag** and hosted on a web server.

- **Applets** are used to make the web site more dynamic and entertaining.

# THANK YOU