Chapter 1

Introduction to Java

What is Java ?

Java is a general-purpose programming language which is class based and object-oriented, designed for building a wide range of applications, from desktop software to web applications, mobile apps, and enterprise systems.

Java follows the principles of object-oriented programming (OOP). It emphasizes the use of classes, objects, and inheritance for building modular and reusable code.

The primary goal of Java is to provide a platform-independent, robust, and secure programming language that allows developers to write code once and run it anywhere (WORA)

History of Java

- Java was developed by Sun Microsystems in the year 1995, by James Gosling.
- *James Gosling* is known as the father of Java.
- Sun Microsystems was acquired by Oracle in 2010
- It is Originally called ‘Oak’. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.
- Java 1.0 was launched in January 1996.

Features of Java

1. Simple :
Java is very easy to learn, and its syntax is simple, clean and easy to understand. Java Syntax is based on C++.

2. Platform Independence:
Platform independence refers to the ability of a programming language or software to run on different platforms (such as different operating systems or hardware architectures) without modification.

Java programs can run on any device or operating system that has a Java Virtual Machine (JVM), adhering to the principle of WORA (Write Once, Run Anywhere)

3. 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 incorporate both data and behavior.

4. Robust and Secure

Java programs are reliable and secure due to features like garbage collection, exception handling, and strict type checking.

5. Portable

Java is portable because it facilitates you to carry the Java bytecode to any platform. It doesn't require any implementation.

6. Multithreading

Java supports concurrent programming with built-in features for handling multiple tasks simultaneously. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area

7. Dynamic

Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand. Java supports dynamic compilation and automatic memory management (garbage collection).

8. High Performance

Java achieves high performance through just-in-time (JIT) compilation and optimized execution.

9. Architecture Neutral
Java bytecode (compiled Java code) can be executed on any system, providing flexibility in deployment.

10. Automatic Memory Management
Java handles memory management automatically through garbage collection, reducing the risk of memory leaks and crashes.

11. Community Support
Java has a large and active community that contributes to its ecosystem with frameworks, libraries, and tools, supporting rapid development and innovation.

Java Environment

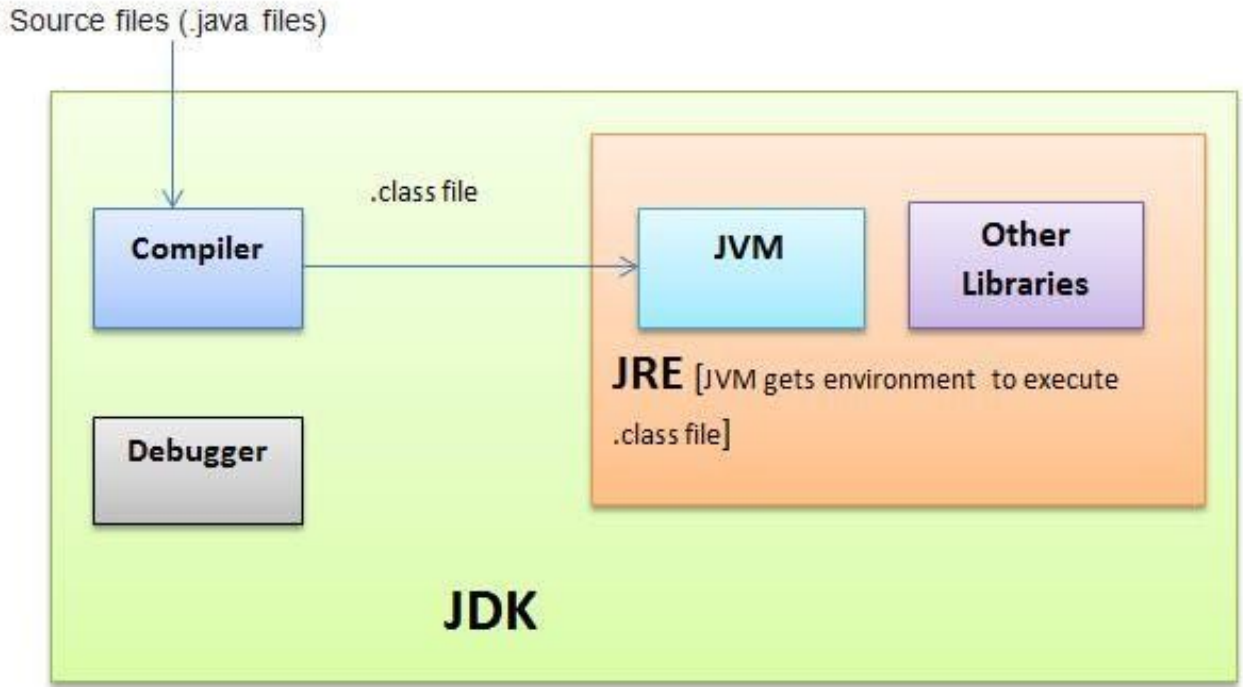
The Java environment consists of the tools and components necessary to develop and run Java applications.

It includes:

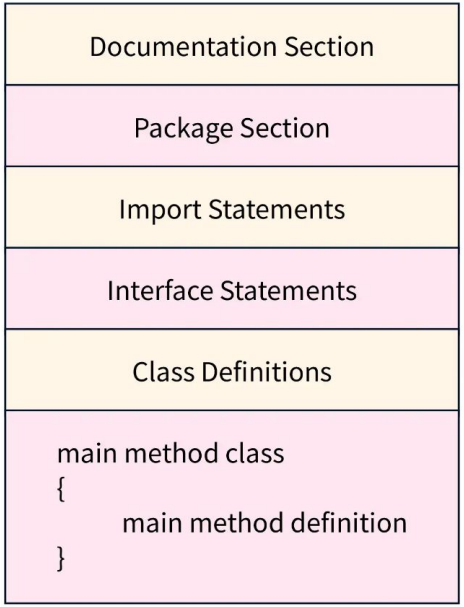
Java Development Kit (JDK): This is the core component for developing Java applications. It includes the Java compiler, which translates Java source code into bytecode, and other tools like the Java debugger and documentation.

Java Runtime Environment (JRE): The JRE is necessary for running Java applications. It includes the Java Virtual Machine (JVM), which executes Java bytecode. Users need the JRE installed to run Java programs.

Java Virtual Machine (JVM): The JVM is a crucial part of the Java environment. It interprets Java bytecode and translates it into machine-specific code, allowing Java applications to run on any device or operating system that has a compatible JVM.



Structure of Java Programs



Main method class is Essential

Simple Java Program

Hello.java

```
//This is a simple "Hello, World!" program in Java.

public class Hello

{

   public static void main(String args[])

   {

       //Printing Hello World

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

   }

}
```

To Compile:
Javac Hello.java

To Execute:
Java Hello

Output
Hello World

Understanding the Program

- The public keyword indicates that the class can be accessed from outside its package.
- The main method is the entry point of a Java application.
- String[] args is a parameter that is typically used to pass command-line arguments to the main method of a Java application
- Statements in Java end with a semicolon (;)

Comments in Java

Comments non-executable statement used to explain Java code, and to make it more readable.

Types of Comment in Java

1. Single-Line Comments:

    Single-line comments start with // and continue until the end of the line. They are used for short comments or annotations on a single line of code.

    Syntax:

    // This is a single-line comment.

2. Multi-Line Comments:

Multi-line comments start with /* and end with */. They can span multiple lines and are used for longer explanations or temporarily disabling blocks of code.

Syntax:

```
/* This is the example of

   Multi-line Comments. */
```

Java Naming Conventions

Classes follows PascalConvention

Example: MyFirstProgram

Functions & Variables follows camelCaseConventions

Example: addTwoNumbers(), myVariable

Constants follows UPPERCASE

Example: MAX_VALUE