Getting Started with Java

Java Terminology

We should know some terms in Java before we begin with Java programming. Below is a brief description of terminologies(given in figure 1.1) often used in Java.

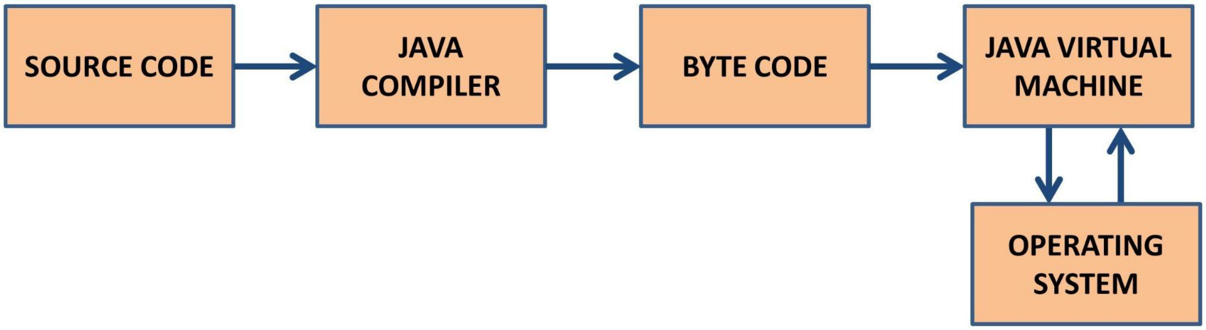

Figure 1.1.Java Terminologies

a. **Source code :** In programming, source code, or simply code, is any collection of code written in a plain text file, with or without comments.

b. **Java Compiler:** A program that takes the text file and compiles it into a platform-independent Java file.

c. **Byte Code:** It is a form of instruction set designed for efficient execution by a software interpreter

d. **JVM (*Java virtual machine*):** It enables a computer to run Java programs (other programs also) written in different languages that are also compiled into Java bytecode.

e. **Operating System:** An operating system is system software that manages computer hardware and software resources and provides standard services for computer programs.

Java Development Kit (JDK)

It is a kit with a compiler, Java Run-time Environment(JRE), Debuggers, and Java documents. JDK should be installed on our computers for execution (creation, compilation, and running) of the java program. We use the JRE Java Run-time Environment, which provides the conditions to execute the Java program. It provides the supporting files, JVM, and core classes.

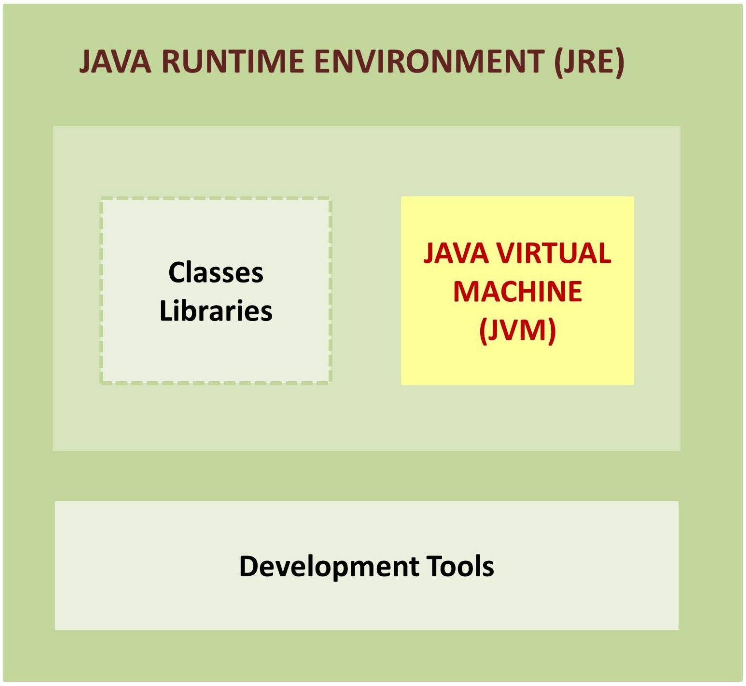

Figure 1.2. Java Development Kit (JDK)

Java Run-time Environment (JRE)

JDK includes JRE, which allows the Java program to run after installation. However, we still can not compile it. It has a browser, applet support, and a few plugins. So, JRE is required to run a java program on our respective computers.

JRE consists of following

- Java class libraries
- Java class loader
- Java virtual machine (JVM)

When our software executes a particular program, it requires some environment to run in. Usually, it is any operating system, for example, Unix, Linux, Microsoft Windows, or macOS. Here JRE acts as a translator and supporter between the operating system and the java program.

Java Virtual Machine (JVM)

It is a significant part of JRE (Java Run-time Environment). When a program is run using JDK and JRE, it also goes to the JVM. It contains three phases:

1. **Compile the Code:** JDK (Java Development Kit) provides a JAVAC compiler to get through this step. JDK converts source code into compiled code (a specific format) that can be easily interpreted by the JRE(Java Run-time Environment).

2. **Run the Code:** The bytecode provided by the compiler is run using JVM. JAVAC compiler produces compiled code which is converted to machine code using platform-specific JVMs. Different platforms have different JVMs. JVMs convert the bytecode into platform-specific machine code.

3. **Bytecode in the development process:** We know JAVAC is present in JDK and helps compile source code into bytecode which a JVM can execute. The compiler saves it as a .class file. It can be viewed as a bytecode using a disassembler like javap.

Top features of Java

1. Object-Oriented Programming Language:

- Managing the program in terms of the collection of objects provides a clear structure for the programs.
- Code maintenance and modification become easier.
- The main concepts of Object-Oriented programming are Abstraction, Encapsulation, Inheritance, and Polymorphism.
- Being an object-oriented language, Java supports methods (referred to as native methods) written in other languages like C and C++.

2. **Platform Independent:**

- Java is platform-independent because it uses JVM, which compiles source code into bytecodes.

- Bytecodes can run on any operating system, be it Linux, Windows, or macOS, which means if we compile a program on Linux OS, then we can run it on Windows OS and vice-versa.
- Every OS has its own JVM (Java Virtual Machine), but the output produced by all the OS is the same after the execution of the bytecode.
- This feature makes Java programs more simple, more secure, and more portable.
- Writing code once and running it anywhere is an alluring feature of being platform independent language.

3. Robust:

- Java is reliable because it is developed in such a way that the Java compiler detects even those errors that are not easily detected by other programming languages.
- Features like robust garbage collection, memory allocation, and Exception Handling makes it more reliable.
- Java uses a JIT (Just In Time) compiler where the code is executed on demand by the compiler.
- This way, only those methods are completed that are called making applications execute faster.

4. Multithreading:

- Java program allows execution of more than one part of a program at one instance for maximum utilization of the CPU.

5. Distributed:

- Java facilitates users to create distributed applications.
- RMI (Remote Method Invocation) and EJB(Enterprise Java Beans) 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.

Applications of Java

Java programming language provides features like High Performance, Dynamic, Multithreaded, and Distributed programming. Therefore, it is fit for multiple platforms, i.e., once code is written in Java, it can be run on any platform. There are various types of applications of the Java programming language.

- Mobile applications.
- Standalone applications.
- Web applications.
- Enterprise applications.