TABLE OF CONTENTS

***Chapter 1: Getting Started with Java***

Indrocution to Java #

What is Java #

Java Virtual Machine #

History and Features of Java Versions #

Java Environment Setup for Windows 7 #

Difference Between JDK,JVM,JRE and JIT #

Garbage Collection in Java #

How Garbage Collector Work in Java #

Chapter 2: Java Basics

Object Oriented Programming Concepts #

Java Basic Syntax #

Hello World Java Example #

Basic Variables and Datatype in Java #

Basic Datatype in Java #

Basic Comments in Java #

Basic Operators in Java #

Control Statements in Java #

Array in Java #

String Class #

Objects and Classes #

Basic Modifier #

Methods #

Constructors #

String In Switch-Java 7 New Concept #

Chapter 3: OOPS Concept With respect To Java

Java Inheritance #

Java Overriding #

Java Polymorphism #

Java Abstraction #

Encapsulation in Java #

Java Interface #

Packages in Java #

Static Keyword in Java #

Final Keyword #

this Keyword #

super Keyword #

instance initializer block #

Chapter 4: Java Exception Handling

Exception Handling #

Try/Catch Block and Handling Exceptions #

Multiple catch Block Handling #

Nested try catch block handling #

Finally block in Java #

Throw Keyword #

Throws keyword #

Exception propagation #

Handle Exceptions in overriding methods #

User Defined Exception in java #

Multiple Exceptions In java 7 New Concept #

Different Exception Generate in Array in Java(7) #

Chapter 5: Basic IO

I/O Streams #

Byte Streams #

Character Streams #

Buffered Stremas #

Scanning and Formatting #

I/O from Command Line #

Data Stream #

Object Stremas #

File I/O(new in Java8) #

Chapter 6: Concurrency

Multithreading #

Life Cycle of a Thread #

Creating a thread in Java #

Thread Scheduling #

Thread Sleep #

Thread Join #

Naming a thread #

Priority of a thread #

Daemon Thread #

Synchronization #

Synchronization Block #

Static Synchronization #

Deadlock #

Inter-Thread Communication #

Thread Control #

Interrupting Java Threads #

Difference Between Wait() and Sleep() #

Chapter 7: Collections Framework

Introduction to Collections Framework #

Life Cycle of a Thread #

Set #

List #

Map #

Difference Between Set, List and Map #

Sorting in Collection Framework #

Iterator #

Comparator and Comparable #

Hash code and Equals method #

**Chapter 1**

**Introduction to Java**

Java is an object oriented programming language developed by James Gosling and Colleagues at Sun Microsystems in the Early 1990’s.

Sun Systems Formally announced Java to outer world in 1995. Sun Microsystems is acquired by Oracle Corp in 2010.

**What is Java?**

Java is

1. Objected Oriented Programming language
2. Platform(machine) independent programming language

**The Java Platform has three basic Editions:**

1. **Java 2 Standard Edition (J2SE):** J2SE is used primarily for writing applets and other Java-based applications.  
     
   **2. Java 2 Enterprise Edition (J2EE):** J2EE is the Java architecture for developing multitier enterprise applications. As part of J2EE, JSP pages have access to all J2EE components, including JavaBeans and Enterprise JavaBeans components and Java servlets. JSP pages are actually compiled into servlets, so they have all of the benefits of these flexible, server-side Java applications.  
     
   **3. Java 2 Micro Edition (J2ME):** J2ME is a technology that allows programmers to use the java programming language and related tools to develop programs for mobile wireless information devices such as cellular phones and personal digital assistants (PDAs).

**Where is it used?**

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, magicbricks.com etc.
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games etc.

**Java Application Types:**

There are different types of Java applications before moving on to Servlets.  
Java can be used to develop different types of applications:

1. Standalone Applications
2. Applets
3. Web Applications
4. Distributed Applications

**History and Features of Java:**

### Java Version History:

There are many java versions that have been released. Current stable release of Java is Java SE 8.

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 March, 2014)

**Features of Java:**

There is given many features of java. They are also known as java buzzwords. The Java Features given below are simple and easy to understand.

1. Simple
2. Object-Oriented
3. Platform independent
4. Secured
5. Robust
6. Architecture neutral
7. Portable
8. Dynamic
9. Interpreted
10. High Performance
11. Multithreaded
12. Distributed

**Java Virtual Machine:**

JVM (Java Virtual Machine) is an abstract machine. It is a specification that provides runtime environment in which java bytecode can be executed.

JVMs are available for many hardware and software platforms (i.e JVM is platform dependent).

**What it does?**

**The JVM performs following operation:**

* Loads code
* Verifies code
* Executes code
* Provides runtime environment

**JVM provides definitions for the:**

* Memory area
* Class file format
* Register set
* Garbage-collected heap
* Fatal error reporting etc.

**List of Few Implementations of JVM available in market:**

* OpenJDK
* Java For HP-UX
* JRockit
* Mac OS Runtime for Java
* Micro JVM

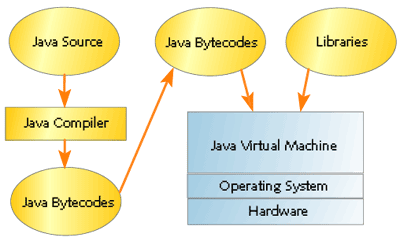


Figure1. Java development and execution infrastructure

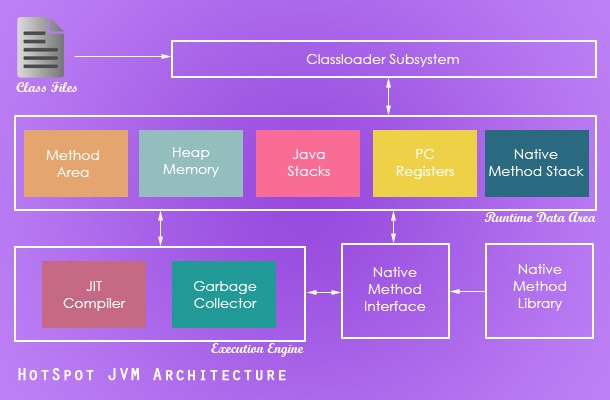


Figure 2: JVM Architecture

**Below is description of JVM components:**

1. **Class loader:** Class loader is a subsystem of JVM that is used to load class files.
2. **Class (Method) Area:** Class (Method) Area stores per-class structures such as the runtime constant pool, field and method data, the code for methods.
3. **Heap:** It is the runtime data area in which objects are allocated.
4. **Stack:** It is used to hold local variables and partial results, and plays a part in method invocation and return. Each thread has a private JVM stack, created at the same time as thread. A new frame is created each time a method is invoked. A frame is destroyed when its method invocation completes.
5. **Program Counter Register: PC** (program counter) registers. It contains the address of the Java virtual machine instruction currently being executed.
6. **Native Method Stack:** It contains all the native methods used in the application.
7. **Execution Engine:** It Contains:
   1. **Virtual Processor**
   2. **Interpreter:** Read bytecode stream then execute the instructions.
   3. **Just-In-Time (JIT) Compiler:** It is used to improve the performance. In [computing](https://en.wikipedia.org/wiki/Computing), just-in-time (JIT) compilation, also known as dynamic translation, is [compilation](https://en.wikipedia.org/wiki/Compiler) done during execution of a program – at [run time](https://en.wikipedia.org/wiki/Run_time_%28program_lifecycle_phase%29) – rather than prior to execution.[[1]](https://en.wikipedia.org/wiki/Just-in-time_compilation#cite_note-FOOTNOTEAycock2003-1) Most often this consists of translation to [machine code](https://en.wikipedia.org/wiki/Machine_code), which is then executed directly, but can also refer to translation to another format.

**Java Environment Setup for Windows:**

* **Install the JDK software.**

1. Go to <http://www.oracle.com/technetwork/java/javase/downloads/index.html>
2. Select the appropriate JDK software and click Download. The JDK software is installed on your computer, for example, at C:\Program Files\Java\jdk1.8.0\_02. You can move the JDK software to another location if desired.

* **Set JAVA\_HOME:**
  1. Right click My Computer and select Properties.
  2. On the Advanced tab, select Environment Variables, and then edit JAVA\_HOME to point to where the JDK software is located, for example, C:\Program Files\Java\jdk1.8.0\_02.
* **Set PATH Variable**
  1. To set the PATH variable permanently, add the full path of the jdk1.7.0\bin directory to the PATH variable. Typically, this full path looks something like C:\Program Files\Java\jdk1.8.0\bin. Set the PATH variable as follows on Microsoft Windows:
  2. Click **Start**, then **Control Panel**, then **System**.
  3. Click **Advanced**, then **Environment Variables**.
  4. Add the location of the bin folder of the JDK installation for the PATH variable in **System Variables**. The following is a typical value for the PATH variable:

C:\WINDOWS\system32;C:\WINDOWS;**C:\Program Files\Java\jdk1.8.0\bin**

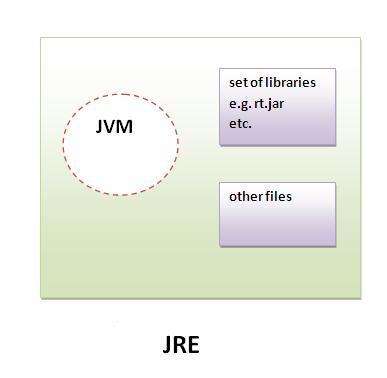
Open Command Prompt and type where Java. You should see your JDK path. Type javac in it should show some options. You should not encounter with errors.

**Difference between JVM, JRE, JDK:**

**JVM:**

* Loads Code
* Verifies Code
* Executes code
* Provides runtime environment

**JRE:** Acronym for Java Runtime Environment. It provide runtime environment. It contains set of Libraries +other files that JVM uses at runtime.



**JDK**:Acronym for Java Development Kit. It contains JRE and development tools.



**END OF Chapter 1**

**Chapter2: Java Basics**

**Object Oriented Programming Concepts (OOPS):**

Object means a real word entity such as pen, chair, table etc. Object-Oriented Programming is a methodology or paradigm to design a program using classes and objects. It simplifies the software development and maintenance by providing some concepts:

* **Object** – Any named entity in real time world is called object. It has state and behavior. EG: house, dog, human, bike etc.
* **Class** – Blue print of an Object is called Class.
* **Abstraction**: Hiding internal details and showing meaningful functionality is known as abstraction. Eg: hiding internal organs of humans and showing only meaningful functionality to outer world.
* **Polymorphism:** Ability to use same interface for deferring datatypes. In a simpler words, when one task is performed in different ways. Eg: Traveler checks can be used as Dollars and Rupees.
* **Inheritance:** An object Acquiring all properties and behavior of parent object.
* **Encapsulation:** Binding functionality (code) and data together in a single unit.

NOTE: remember APIE