CSE310: Programming in Java

Topic: Java Platform Overview

Outlines

- Introduction
- Why Java?
- Where it is used?
- Characteristics of Java
- Java Platform Editions
- Evolution of Java
- Understanding JDK, JRE and JVM
- How Java is platform-independent
- Evaluating Java libraries, middle-ware, and database options

Introduction

- Java is a general purpose object-oriented programming language.
- Java is a high level, robust, secured and platform independent.
- Developed by Sun Microsystems (James Gosling).

Platform: *Any hardware or software environment in which a program runs, is known as a platform.*

Why Java?

- Java is the Internet programming language.
- Java enables users to develop and deploy applications on the Internet for servers, desktop computers, and small hand-held devices.

Where it is Used?

- According to Sun, 3 billion devices run java.

- 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 Applications
- Embedded System, Smart Card, Robotics, Games etc.

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- **Java Is Secure**
- **Java Is Architecture-Neutral**
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Java Platform Editions

- A Java Platform is the set of APIs, class libraries, and other programs used in developing Java programs for specific applications.

There are 3 Java Platform Editions

1. Java 2 Platform, Standard Edition (J2SE)
    - Core Java Platform targeting applications running on workstations

2. Java 2 Platform, Enterprise Edition (J2EE)
    - Component-based approach to developing distributed, multi-tier enterprise applications

3. Java 2 Platform, Micro Edition (J2ME)
    - Targeted at small, stand-alone or connectable consumer and embedded devices

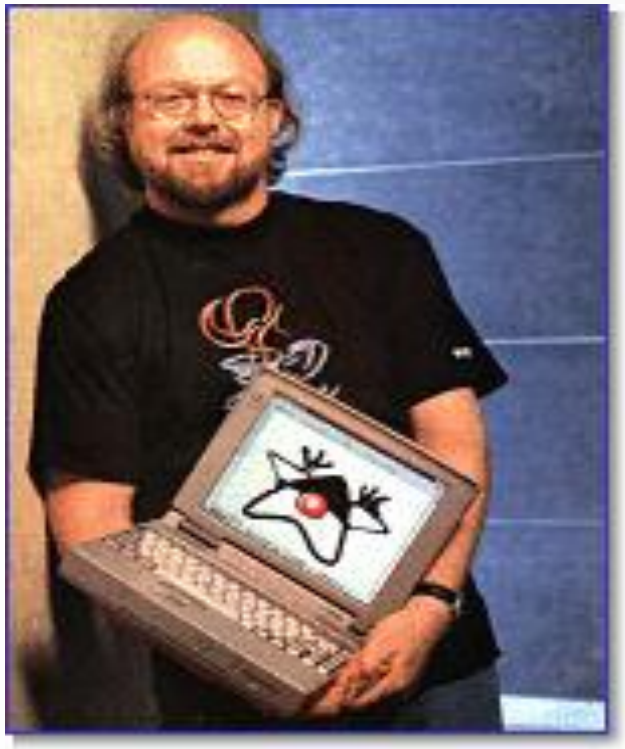
Origin of Java

James Gosling & Patrick Naughton at 1990

Goal : to develop distributed system which is applicable to electronic products(platform independent)

James Gosling

- James Gosling is generally credited as the inventor of the Java programming language



- He was the first designer of Java and implemented its original compiler and virtual machine
- He is also known as the Father of Java.

Brief History of Java

- In 1990, Sun Microsystems began an internal project known as the ***Green Project*** to work on a new technology.

- In 1992, the Green Project was spun off and its interest directed toward building highly interactive devices for the cable TV industry. This failed to materialize.

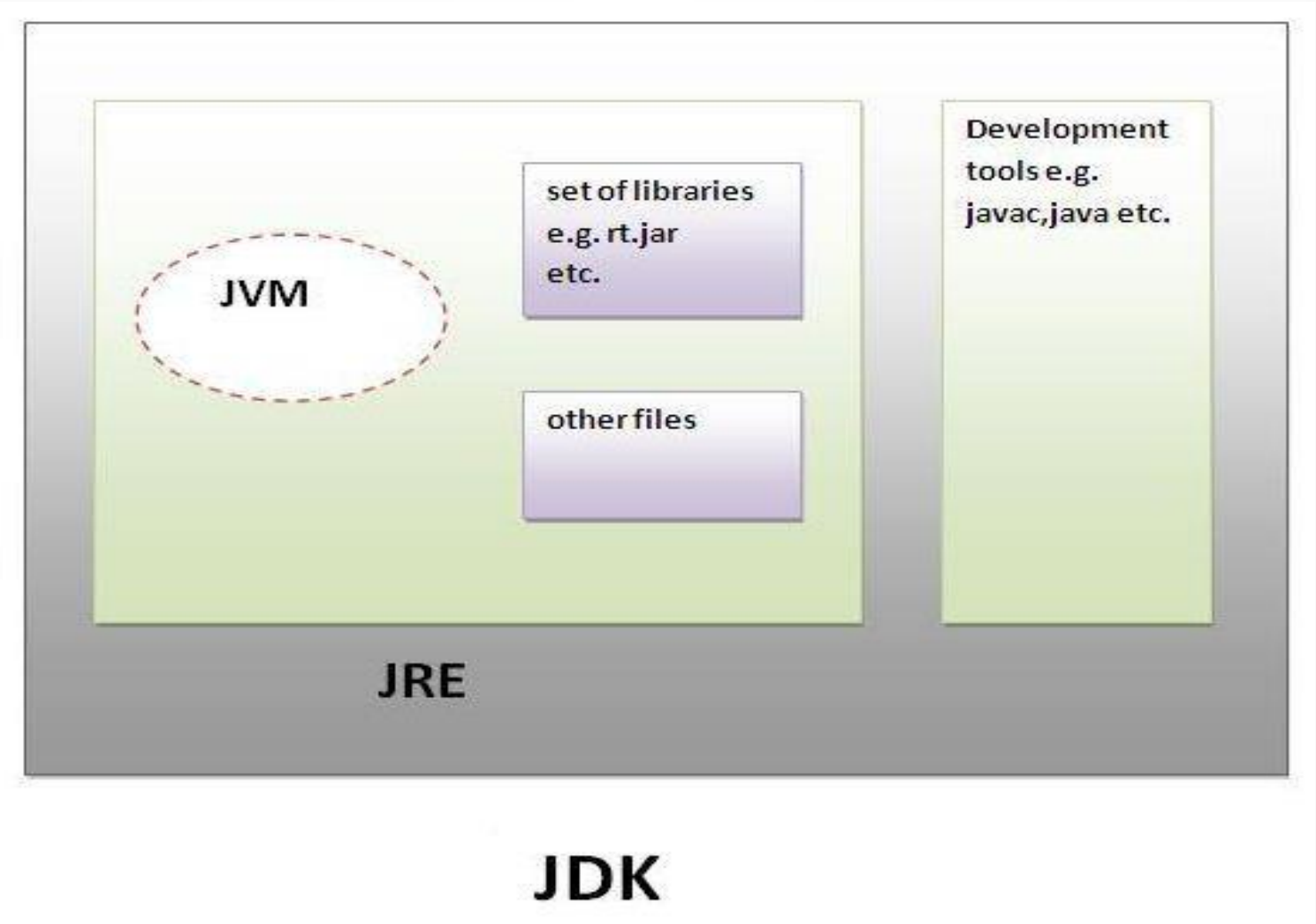
- In 1994, the focus of the original team was re-targeted, this time to the use of Internet technology. A small web browser called ***HotJava*** was written.
- Oak was renamed to ***Java*** after learning that Oak had already been trademarked.

- In 1995, Java was first publicly released.
- In 1996, Java Development Kit (***JDK***) 1.0 was released.
- In 2002, JDK 1.4 (codename *Merlin*) was released, the most widely used version.
- In 2004, JDK 5.0 (codename *Tiger*) was released.
- The latest version of java is jdk 8.0.

JDK Versions

- JDK Alpha and Beta (1995)
- JDK 1.0 (23rd Jan, 1996)
- JDK 1.1 (19th Feb, 1997)
- J2SE 1.2 (8th Dec, 1998)
- J2SE 1.3 (8th May, 2000)
- J2SE 1.4 (6th Feb, 2002)
- J2SE 5.0 (30th Sep, 2004)
- Java SE 6 (11th Dec, 2006)
- Java SE 7 (28th July, 2011)
- Java SE 8 (18th March, 2014)

Understanding JDK, JRE and JVM



Understanding JDK & JRE

JDK

- JDK is an acronym for *Java Development Kit*.
- It physically exists. It contains JRE and development tools.

JRE

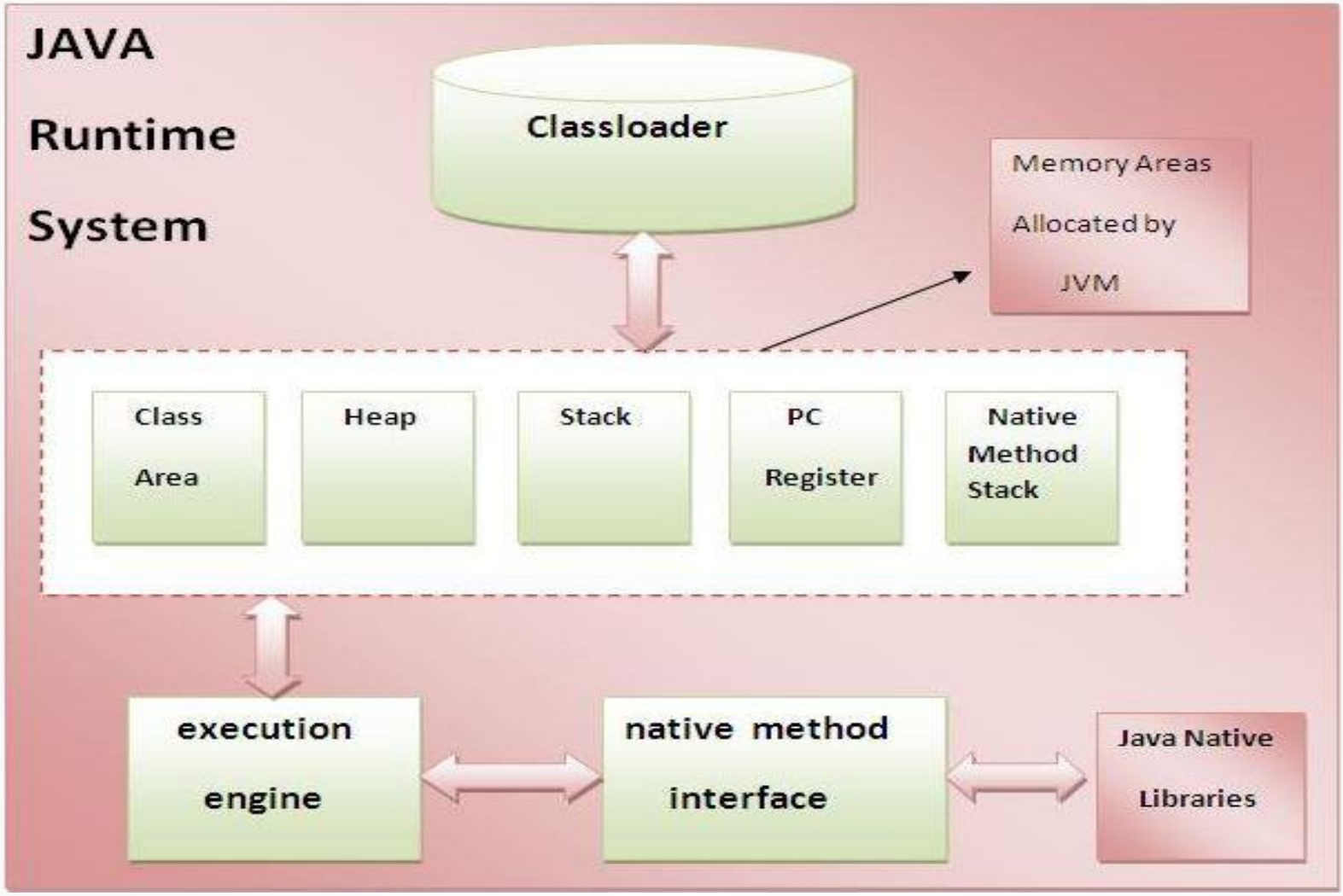
- JRE is an acronym for *Java Runtime Environment*.
- It is the implementation of JVM and used to provide runtime environment.
- It contains set of libraries and other files that JVM uses at runtime.

Understanding JVM

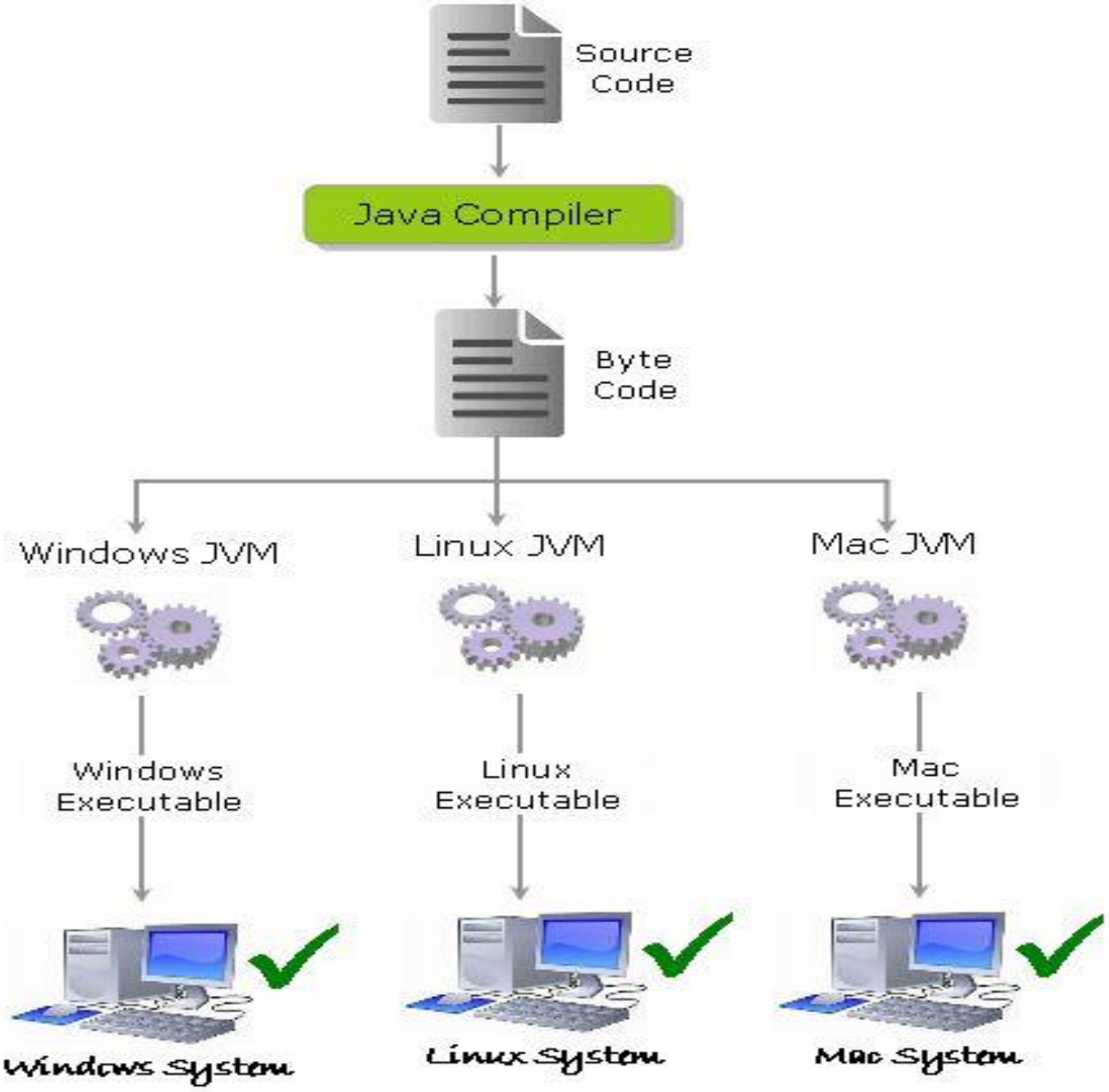
- JVM (Java Virtual Machine) is an abstract machine.
- It is a specification that provides runtime environment in which java byte code can be executed.
- JVMs are available for many hardware and software platforms.

- The JVM performs following main tasks:
  - Loads code
  - Verifies code
  - Executes code
  - Provides runtime environment

Internal Architecture of JVM



How Java is Platform-independent?


Source Code
Java Compiler
Byte Code
Windows JVM
Linux JVM
Mac JVM
Windows Executable
Linux Executable
Mac Executable
Windows System
Linux System
Mac System

How Java is Platform-independent?

- The source code (program) written in java is saved as a file with .java extension.

- The java compiler “javac” compiles the source code and produces the platform independent intermediate code called BYTE CODE. It is a highly optimized set of instructions designed to be executed by the JVM.

How Java is Platform-independent?

- The byte code is not native to any platform because java compiler doesn't interact with the local platform while generating byte code.

- It means that the Byte code generated on Windows is same as the byte code generated on Linux for the same java code.

- The Byte code generated by the compiler would be saved as a file with .class extension. As it is not generated for any platform, can't be directly executed on any CPU.

Java Libraries, Middle-ware, and Database options

- java.lang
- java.util
- java.sql
- java.io
- java.nio
- java.awt
- javax.swing