# PROGRAMMING IN ADVANCE JAVA
# Introduction to JAVA

Lecture 1: overview of Java

By
Jitender Singh
Asst. Professor,
GITAM

# Contents

- What is JAVA?
- What is OOP?
- Why Java?
- Characteristics of JAVA
- Overview of Java.
- The difference between Java and C++?
- A Simple JAVA Program

# What is JAVA?

- Developed by Sun Microsystems (James Gosling)
- A general-purpose Object-Oriented language
- Based on C/C++
- Designed for easy Web/Internet applications
- Widespread acceptance

# What is Object Oriented Programming?

Object-oriented programming is a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of one or more hierarchy of classes united via inheritance relationships.

# Principles of OOP

- All object-oriented programming languages provide mechanisms that help you implement the object-oriented model.

- They are:
  - Encapsulation
  - Inheritance
  - Polymorphism

# Data Encapsulation

Encapsulation is the mechanism that binds together code and the data it manipulates, and keeps both safe from outside interference and misuse.

```
class Account {
                int x=10;
                float withdraw();
                void deposit(float amount);
                float balance;
               }
```

# Advantages of Encapsulation

- Protection
- Consistency
- Allows change

# Inheritance

- Inheritance is the process by which one object acquires the properties of another object.

- A class that is derived from another class is called a subclass (also a derived class, extended class, or child class).

- The class from which the subclass is derived is called a super class (also a base class or a parent class).

- A class which is a subtype of a more general class is said to be inherited from it.

# Inheritance cont'd

- The sub-class inherits the base class' data members and member functions.
- A sub-class has all data members of its base-class plus its own.
- A sub-class has all member functions of its base class (with changes) plus its own.

# Polymorphism

- *Polymorphism* refers to a programming language's ability to process objects differently depending on their data type or class.
- Generally, the ability to appear in many forms.

- More specifically, it is the ability to redefine *methods* for *derived classes.*

- For example, given a base class *shape,* polymorphism enables the programmer to define different *area* methods for any number of derived classes, such as circles, rectangles and triangles. No matter what shape an object is, applying the *area* method to it will return the correct results.

# 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.

# Java, Web, and Beyond

- Java Applets
- Java Web Applications
- Java can also be used to develop applications for hand-held devices such as Palm and cell phones

# 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

# 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 is partially modeled on C++, but greatly simplified and improved. Some people refer to Java as "C++--" because it is like C++ but with more functionality and fewer negative aspects.

# 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 is inherently object-oriented. Although many object-oriented languages began strictly as procedural languages, Java was designed from the start to be object-oriented. Object-oriented programming (OOP) is a popular programming approach that is replacing traditional procedural programming techniques.

One of the central issues in software development is how to reuse code. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through encapsulation, inheritance, and polymorphism.

# 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

Distributed computing involves several computers working together on a network. Java is designed to make distributed computing easy. Since networking capability is inherently integrated into Java, writing network programs is like sending and receiving data to and from a file.

# 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

You need an interpreter to run Java programs. The programs are compiled into the Java Virtual Machine code called bytecode. The bytecode is machine-independent and can run on any machine that has a Java interpreter, which is part of the Java Virtual Machine (JVM).

# 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 compilers can detect many problems that would first show up at execution time in other languages.

Java has eliminated certain types of error-prone programming constructs found in other languages.

Java has a runtime exception-handling feature to provide programming support for robustness.

# 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 implements several security mechanisms to protect your system against harm caused by stray programs.

# 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

Write once, run anywhere

With a Java Virtual Machine (JVM), you can write one program that will run on any platform.

# 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

Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

# 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's performance Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

# 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

Multithread programming is smoothly integrated in Java, whereas in other languages you have to call procedures specific to the operating system to enable multithreading.

# 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 was designed to adapt to an evolving environment. New code can be loaded on the fly without recompilation. There is no need for developers to create, and for users to install, major new software versions. New features can be incorporated transparently as needed.

# Java Disadvantages

- **Slower than compiled language such as C**
  - an experiment in 1999 showed that Java was 3 or 4 times slower than C or C++

# Origin of Java

- Generation
  - 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.

# JDK Versions

- JDK 1.02 (1995) … 250 classes
- JDK 1.1 (1996) … 500 classes
- JDK 1.2 (1998) … 2300 classes
- JDK 1.3 (2000) … 2300 classes
- JDK 1.4 (2002) … 2300 classes
- JDK 1.5 (2004) … 3500 classes
- JDK 1.6 (2006) … 3500 classes
- JDK 1.7 (2011) … 3500 classes

# 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

# Java Development Kit (JDK)

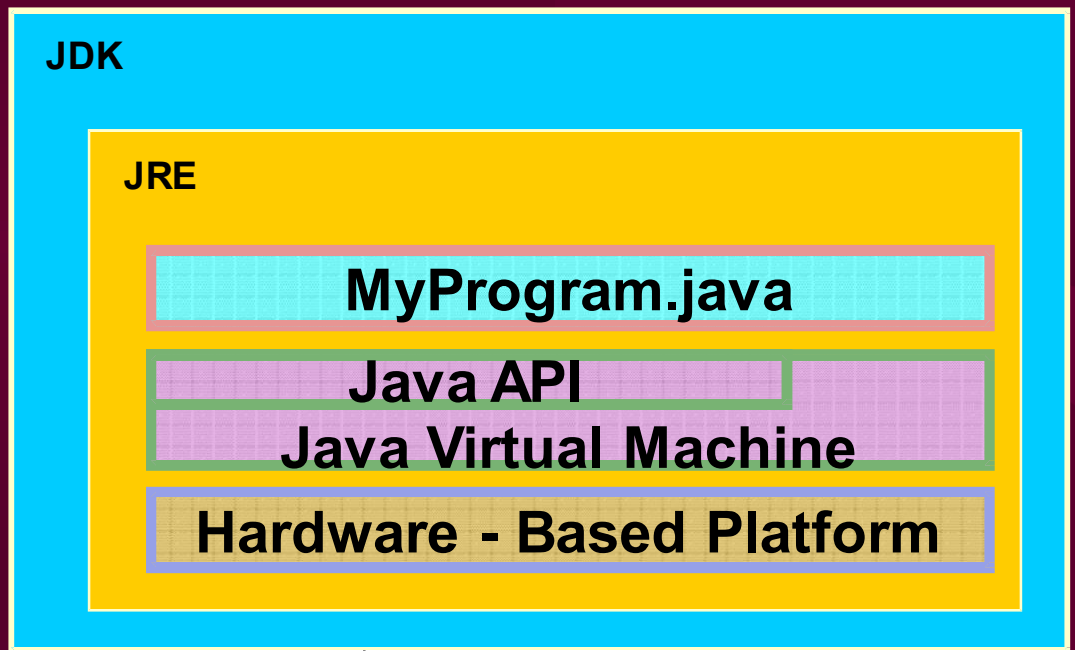
Java Development Kit (JDK)

- Is a set of Java tools for developing Java programs
- Consists of Java API, Java Compiler, and JVM

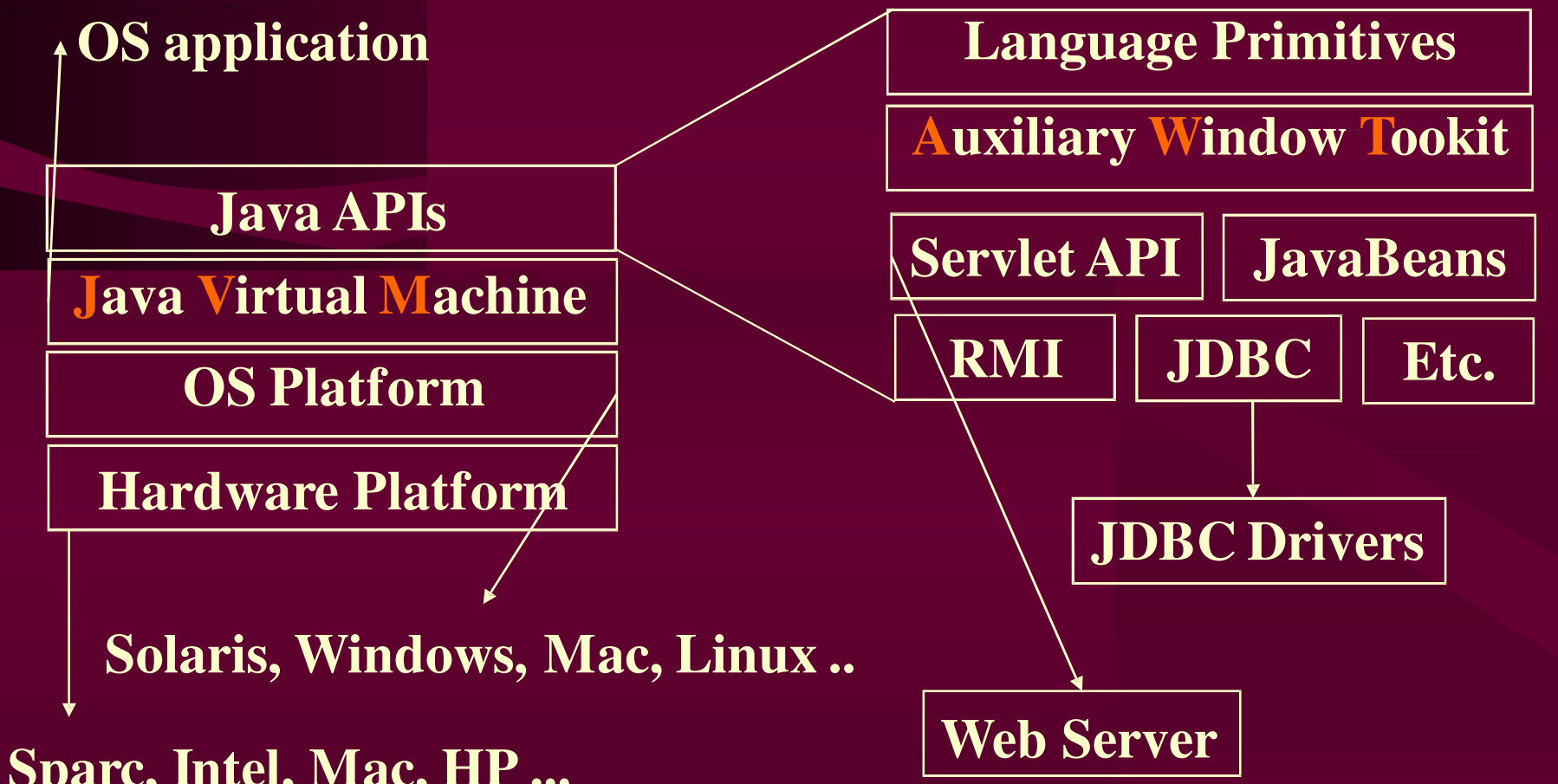
Java Application Programming Interface (API)

- Is prewritten code, organized into packages of similar topics

Java Virtual Machine (JVM) is an execution engine that runs compiled Java byte code.



# Structure Overview of Java



# Java life cycle

- Java programs normally undergo four phases
  - Edit
    - Programmer writes program (and stores program on disk)
  - Compile
    - Compiler creates *byte-codes* from program (.class)
  - Load
    - Class loader stores byte-codes in memory
  - Execute
    - Interpreter: translates *byte-codes* into machine language

# Java life cycle

- Source code (.java)

- Compiled into Byte codes (.class) , as (.exe) in c++

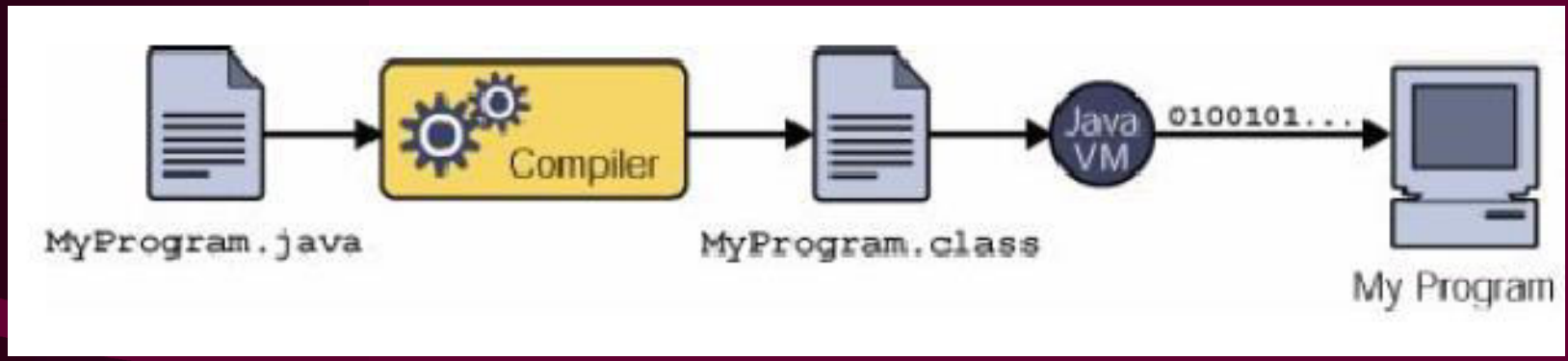
–The Java Application Programming Interface (API)

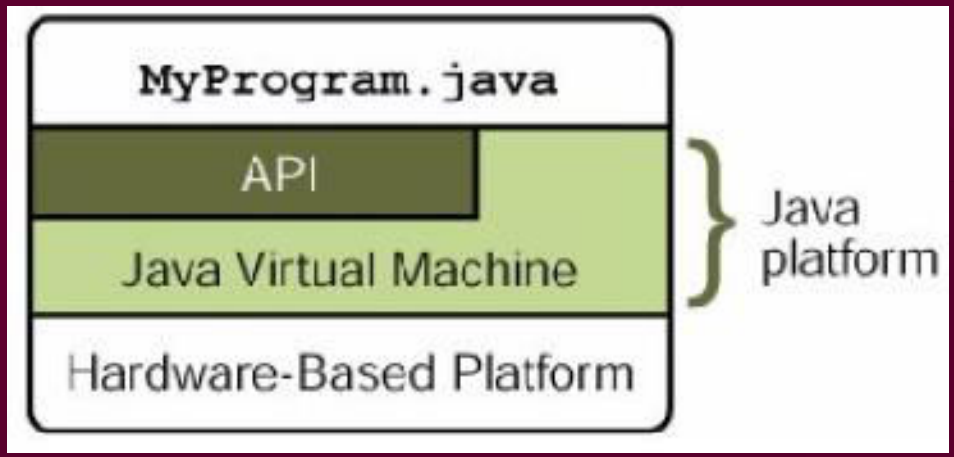
  - a large collection of ready-made software components. It is grouped into libraries of related classes and interfaces; these libraries are known as packages.

–Java Virtual Machine (JVM)

–Machine code

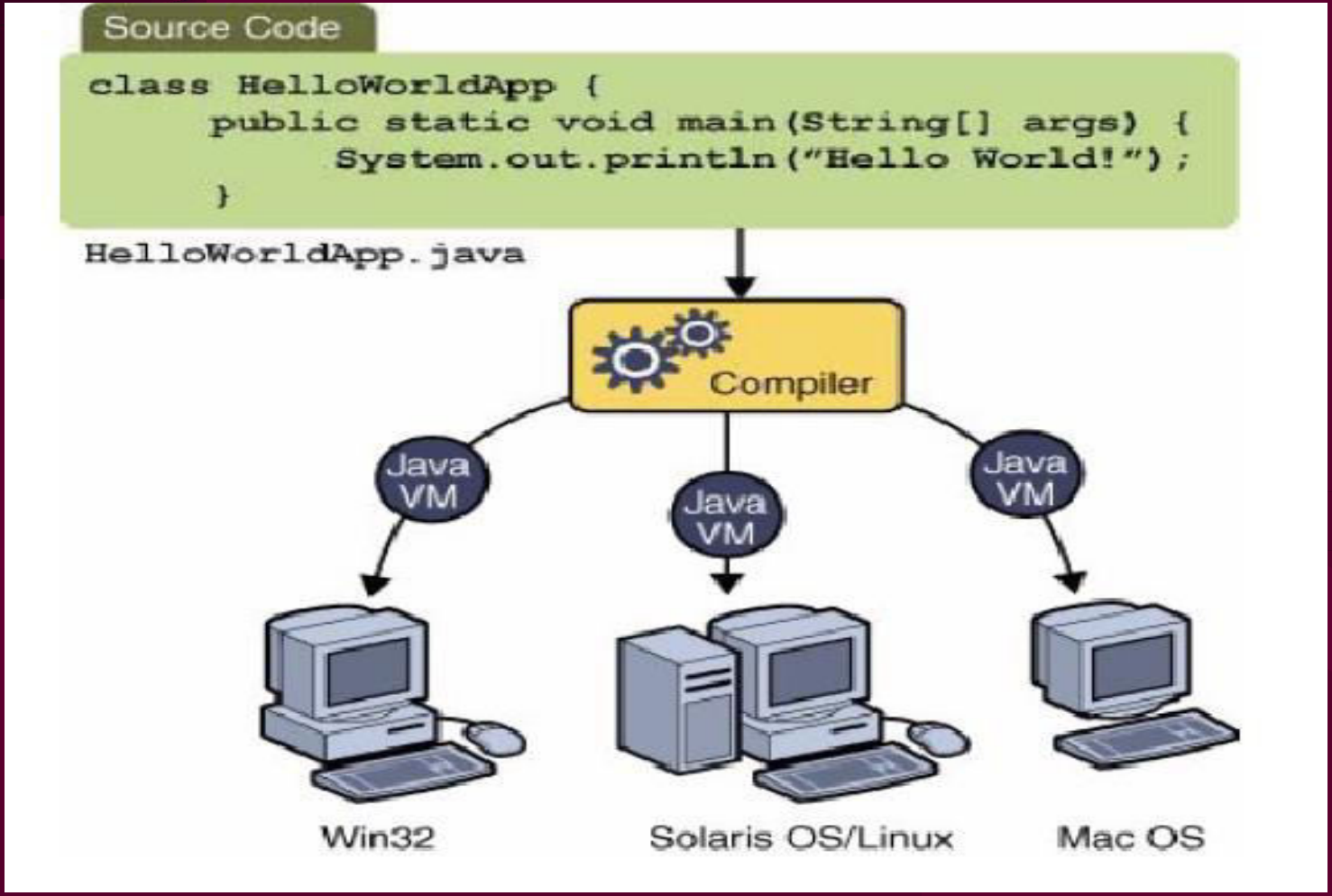
# Java life, cont..



# JVM an Portability

- Through the Java VM, the same application is capable of **running on multiple platforms**.



# A Simple Java Program

```
//This program prints Welcome to Java!
        public class Welcome
                {
                 public static void main(String[] args)
                        {
                         System.out.println("Welcome  to Java!");
                         }
                 }
```

# Compiling Java Source Code

Java was designed to run object programs on any platform.

With Java, you write the program once, and compile the source program into a special type of object code, known as *bytecode*.

The byte-code can then run on any computer with a Java Virtual Machine, as shown below.

Java Virtual Machine is a software that interprets Java byte-code.