Introduction to Java

Agenda

- **What's java.**
- **Characteristics of Java.**
- **JDK, JRE, and JVM.**
- **The First Java Program.**

What is Java

What is java

- Java is a **high-level** programming language originally developed by **Sun Microsystems** and released in **1995**. Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX.

- Now it's owned by **Oracle**, and more than **3 billion** devices run Java.

What is java

- It's a Programming language used to develop
  - **Desktop** application
  - **Web** application
  - **Mobile** application

Java applications

- Java **Standard** Edition (**J2SE**)

  ✓ Develop client-side standalone applications.

- Java **Enterprise** Edition (**J2EE**)

  ✓ Develop server-side applications : **Servlets -JSP**.

- Java **Micro** Edition (**J2ME**).

  ✓ Develop mobile applications.

Characteristics of Java
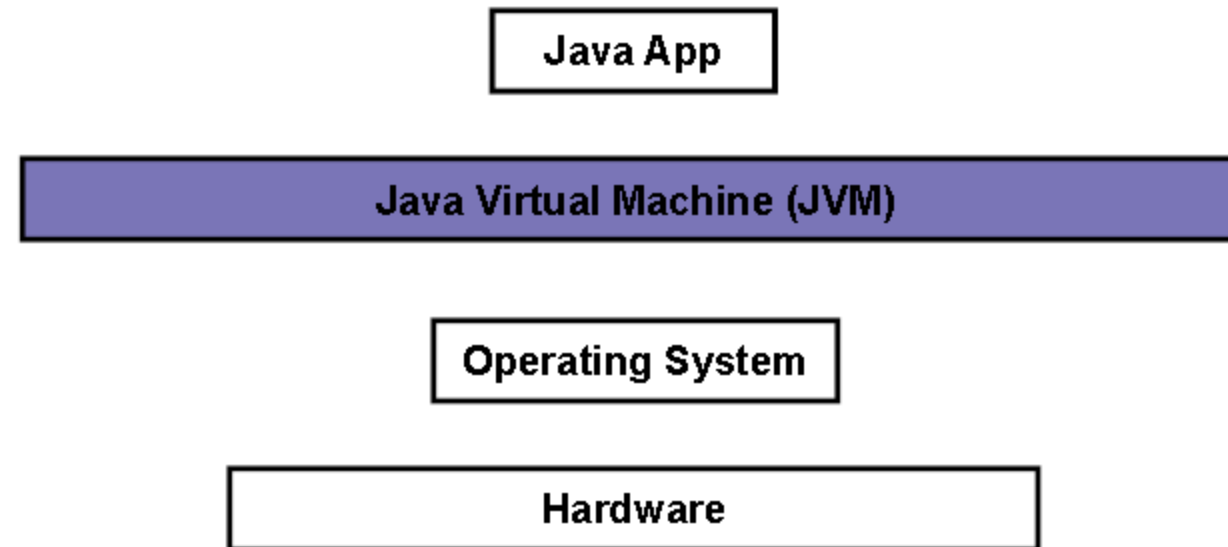
Characteristics of Java

- Java is simple .
- Java is easy to learn.
- Java is Object-Oriented .
- Java is multithreaded .
- Java is machine and platform independent.
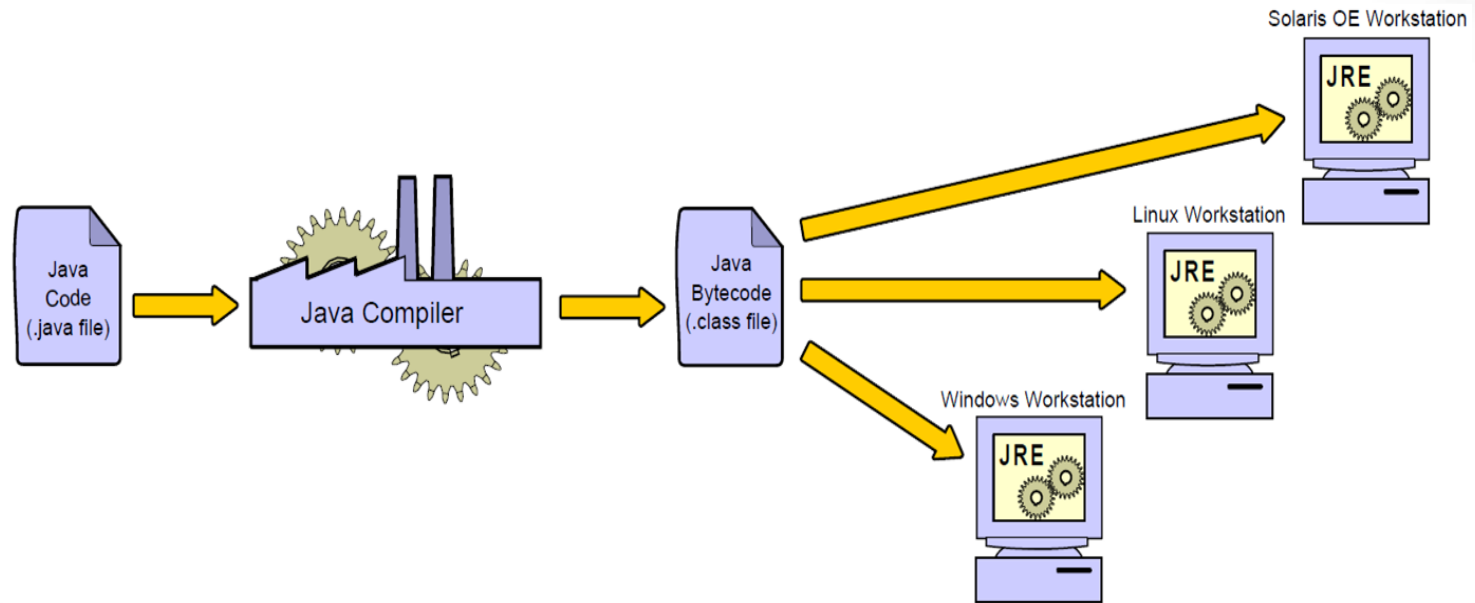 **"WRITE ONCE, RUN ANY WHERE!"**

Characteristics of Java
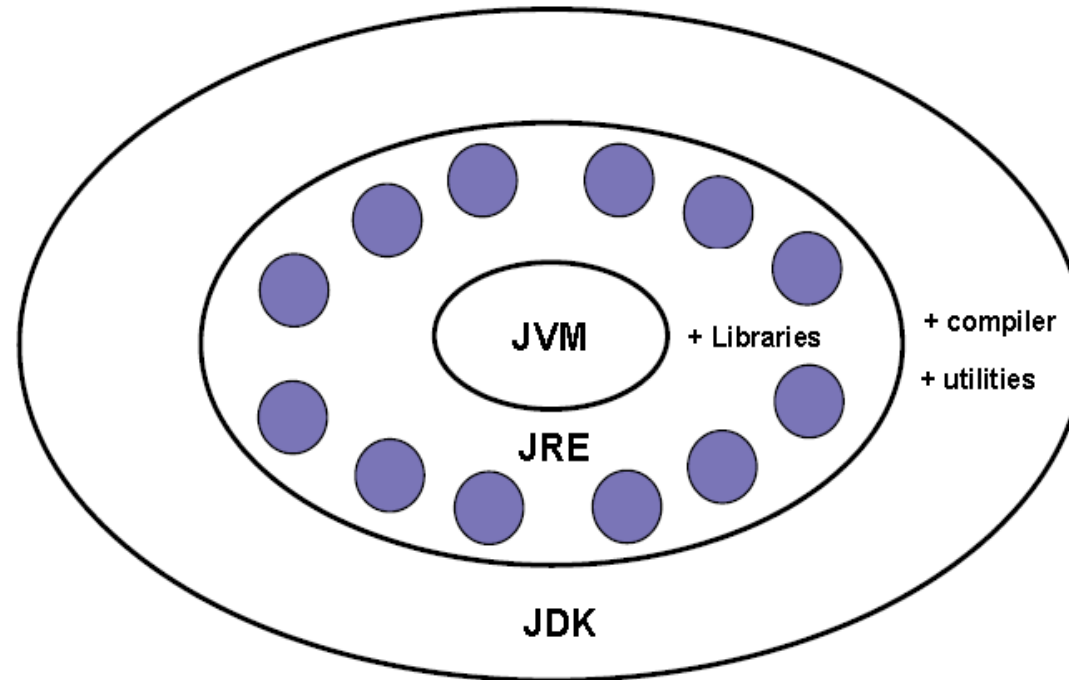
Java Platform-Independent

JDK, JRE and JVM

JDK, JRE and JVM

- Java depends on dynamic linking of libraries.

JDK

- **Java Development Kit** contains tools needed to develop the *Java programs:*
  - **compiler** (javac.exe), Java application launcher (java.exe).
  - *JRE* to run the programs.

JRE

- Stands for **Java Runtime Environment**.
- Java Packages Classes:
  - util, math, lang, awt, swing etc.
- **JVM**.
- JRE connect JVM with java packages and libraries.

JVM

- Java programs executed in JVM.
- JVM is a virtual rather than a physical machine.
- JVM is the heart of Java programming language.
- Translate java byte code instructions into the machine specific code.

The First Java Program

The First Java Program

- To write this simple program you need **IDE**
  - **Netbeans**.
  - **Eclipse**.
  - **Jdeveloper**.
- To compile and run it we need **JDK**.

The First Java Program

```java
/* My First Program!!*/
public class HelloWorld {
    public static void main(String[] args){
        //prints the string Hello world on screen
        System.out.println("Hello World!");
    }
}
```

Command-line arguments

- public static void main(**String[] args**)
- if you run this line in command-line:
  - java **HelloWorld** **one two**
- Then **args** contains:
  - **["one", "two"]**

```java
public static void main(String [] args) {
  String one = args[0]; //=="one"
String two = args[1]; //=="two" }
```

Why is the Java main method static?

- Because they can then be **invoked** by the **runtime** engine without having to instantiate any objects then the code in the body of main() will do the rest.

The First Java Program

- ***println()*** places a newline character at the end of whatever is being printed out.

- The following lines:

  - ```
    System.out.println(" This is printed out");
    System.out.println(" on two lines.");
    ```

  results in 2 lines of output.

```
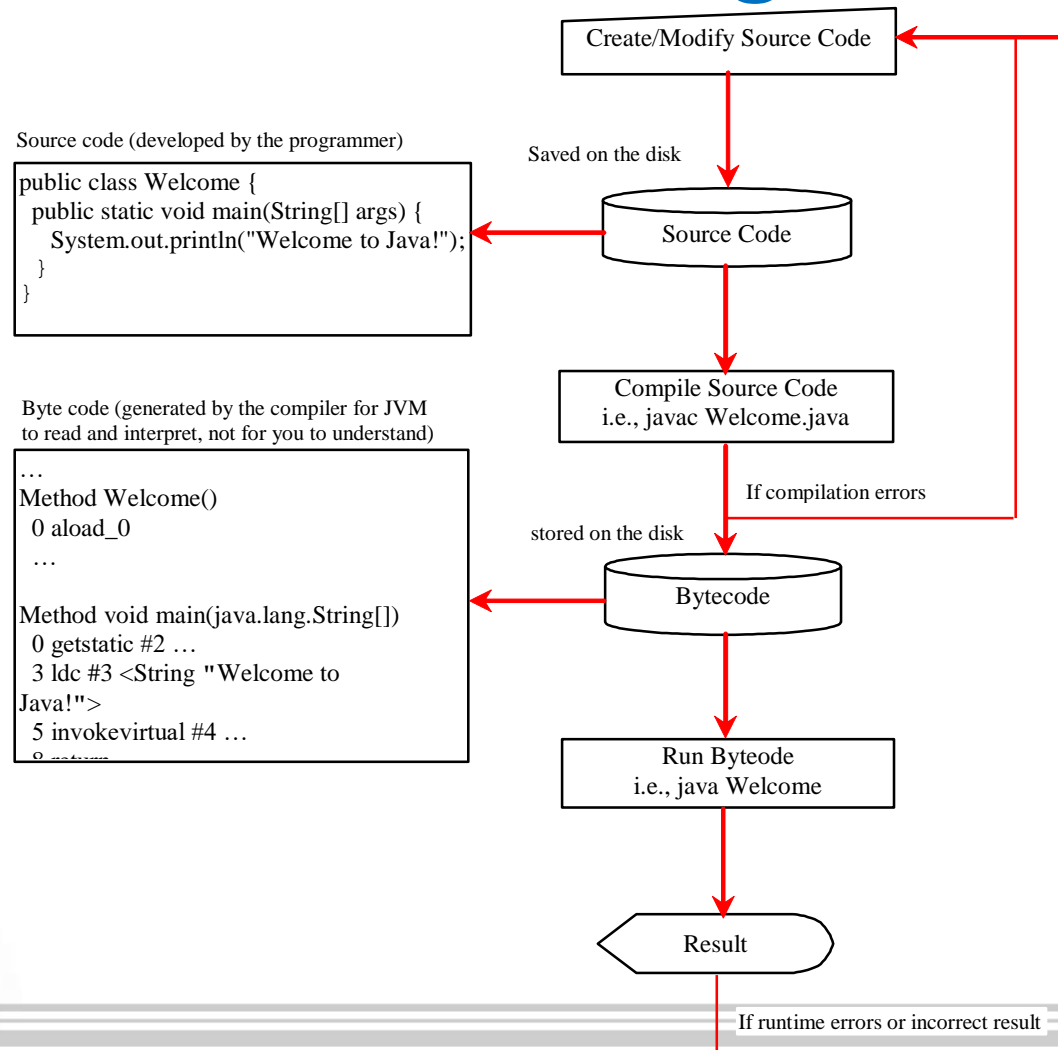Output :
This is printed out
On two lines.
```

The First Java Program

- ***Print()*** result in one line :
  - ```
    System.out.print(" These lines will be");
    System.out.print(" on the same line");
    ```

> **Output :**
> **These lines will be on the same line**

The First Java Program

Create/Modify Source Code

Source code (developed by the programmer)

Saved on the disk

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

Source Code

Byte code (generated by the compiler for JVM to read and interpret, not for you to understand)

Compile Source Code
i.e., javac Welcome.java

```
…
Method Welcome()
 0 aload_0
 …

Method void main(java.lang.String[])
 0 getstatic #2 …
 3 ldc #3 <String "Welcome to Java!">
 5 invokevirtual #4 …
```

If compilation errors

stored on the disk

Bytecode

Run Byteode
i.e., java Welcome

Result

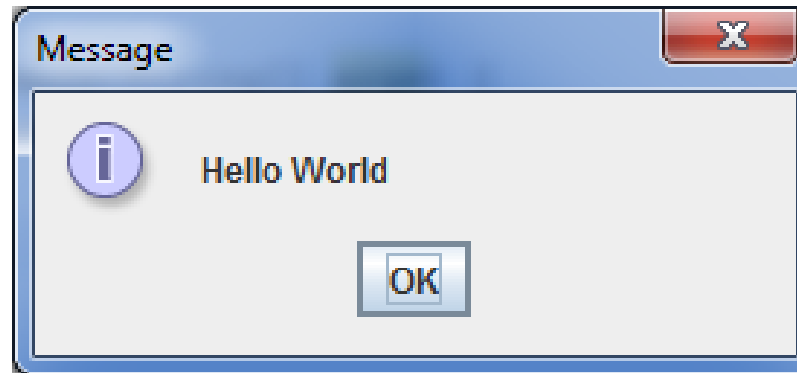If runtime errors or incorrect result

Coding Guidelines

- Your Java programs should always end with the .**java** extension.
- Filenames should match the name of your public class.
- You should write comments in your code explaining what a certain class does, or what a certain method does.
- Naming Conventions

Message Dialog

- You can rewrite the program to display the text in a message dialog box.
- Use the **JOptionPane** class:
  - JoptionPane.showMessageDialog(null,"Hello ");

Links

- **NetBeans With JDK**
  - ➢ Netbeans - Google Drive

Thanks