Khulna Khan Bahadur Ahsanullah University

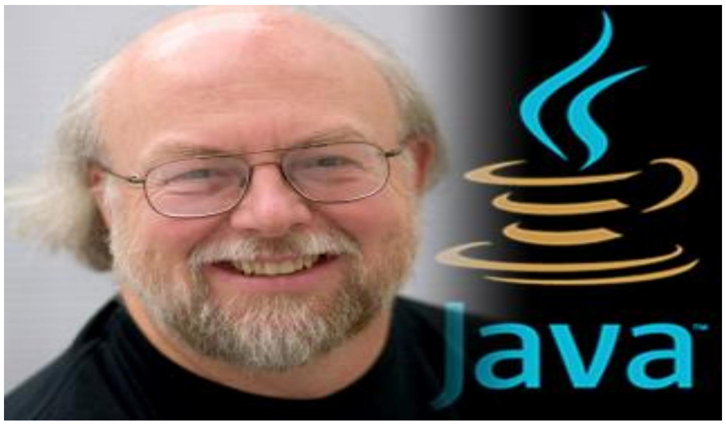
Object-oriented programming

CSE 1203

What is Java?

Java is a **programming language** and a **platform**. Java is a high level, robust, object-oriented and secure programming language.

Java was developed by **Sun Microsystems** (which is now the subsidiary of Oracle) in the year 1995. **James Gosling** is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java.



Platform: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

Why Java programming language is named JAVA?

- The initial name was OAK but it was renamed to Java.
- Java is the name of an island in Indonesia where the first coffee was produced.
- This name was chosen by James Gosling while having coffee near his office.

Why Java Language was created?

- Object oriented.
- A single representation of a program could be executed on multiple operating systems (Write once, run anywhere).
- Support network programing.
- Execute code from remote sources securely.
- Easy to use.

Application

1. Desktop Applications such as acrobat reader, media player, antivirus, etc.
2. Web Applications such as irctc.co.in, javatpoint.com, etc.
3. Enterprise Applications such as banking applications.
4. Mobile
5. Embedded System
6. Smart Card
7. Robotics
8. Games, etc.

Types of Java Applications

There are mainly **4 types of applications** that can be created using Java programming:

1) **Standalone Application**

Standalone applications are also known as desktop applications or window-based applications. These are traditional software that we need to install on every machine. Examples of standalone application are Media player, antivirus, etc. AWT and Swing are used in Java for creating standalone applications.

2) **Web Application**

An application that runs on the server side and creates a dynamic page is called a web application. Currently, Servlet, JSP, Struts, Spring, Hibernate, JSF, etc. technologies are used for creating web applications in Java.

3) Enterprise Application

An application that is distributed in nature, such as banking applications, etc. is called an enterprise application. It has advantages like high-level security, load balancing, and clustering. In Java, EJB is used for creating enterprise applications.

4) Mobile Application

An application which is created for mobile devices is called a mobile application. Currently, Android and Java ME are used for creating mobile applications.

Java Platforms / Editions

There are 4 platforms or editions of Java:

1) Java SE (Java Standard Edition)

It is a Java programming platform. It includes Java programming APIs such as java.lang, java.io, java.net, java.util, java.sql, java.math etc. It includes core topics like OOPs, String, Regex, Exception, Inner classes, Multithreading, I/O Stream, Networking, AWT, Swing, Reflection, Collection, etc.

2) Java EE (Java Enterprise Edition)

It is an enterprise platform that is mainly used to develop web and enterprise applications. It is built on top of the Java SE platform. It includes topics like Servlet, JSP, Web Services, EJB, JPA, etc.

3) Java ME (Java Micro Edition)

It is a micro platform that is dedicated to mobile applications.

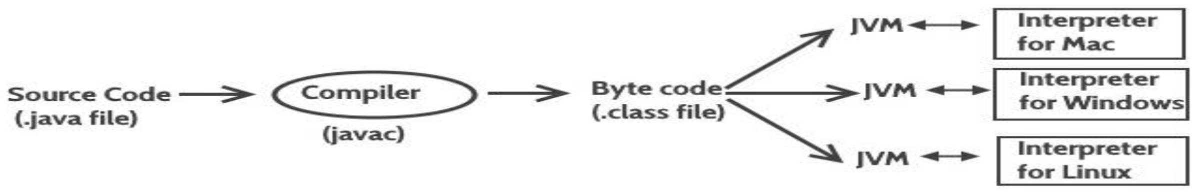
4) JavaFX

It is used to develop rich internet applications. It uses a lightweight user interface API.

Java Terminology

What is JVM?

- Java Virtual Machine (JVM) is a part of the JDK and the foundation of the Java platform.
- It is a virtual machine that resides in the real machine and the machine language for JVM is byte code.
- JVM executes the bytecode generated by compiler and produce output. JVM is the one that makes java **platform independent**.



What is JDK?

- The Java Development Kit (JDK) is a **software package** that makes developing Java applications easier.
- Java development kit is a software development environment used for developing java applications includes JRE, Jar, JVM.

What is JRE?

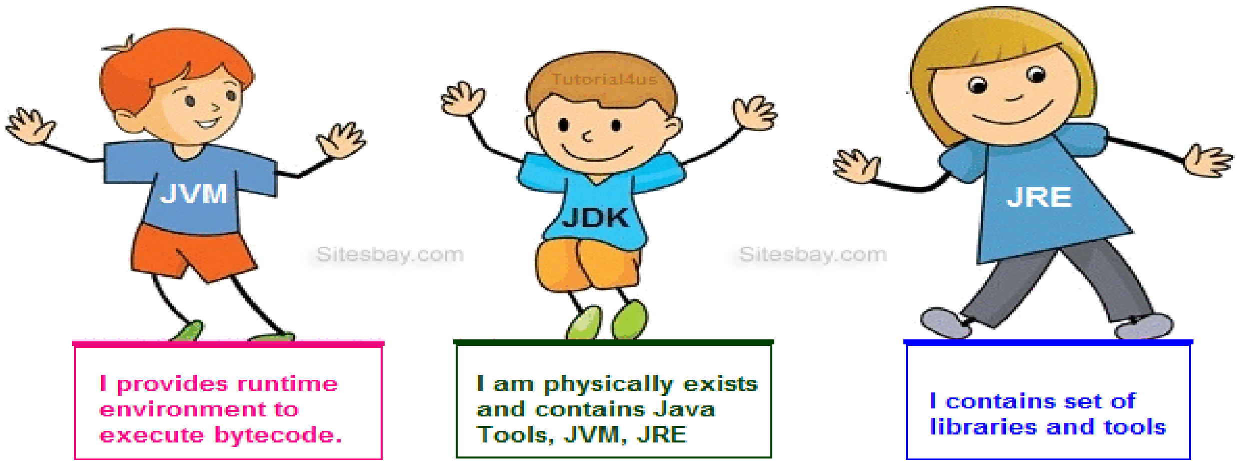
- The Java Runtime Environment (JRE) is part of the Java Development Kit (JDK).
- It contains **set of libraries and tools** for developing java application.
- The Java Runtime Environment provides the minimum requirements for executing a Java application.

What is JIT?

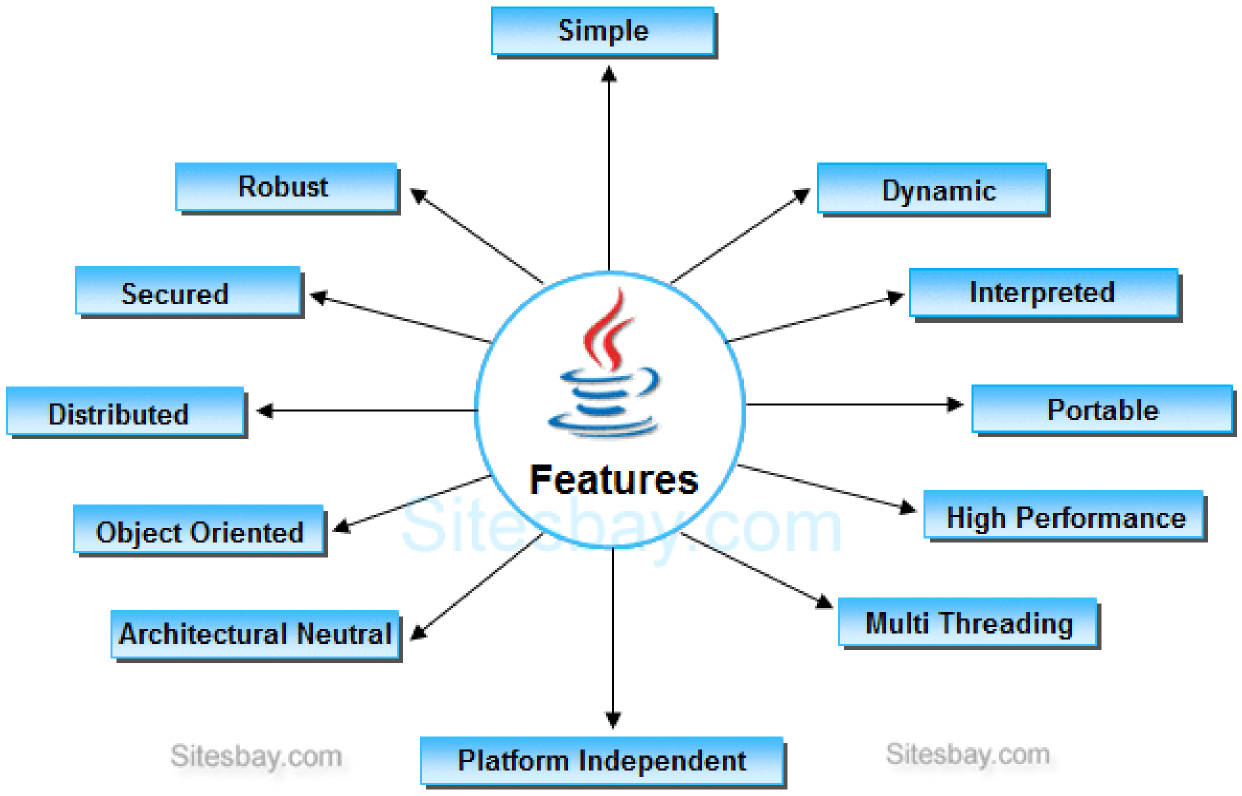
- The Just-In-Time (JIT) compiler is a component of the runtime environment that **improves the performance** of Java applications by compiling bytecodes to native machine code at run time.
- The main aim of JIT compiler is to **speed up the execution** providing machine level instruction to the process which are available in the buffer memory.

Difference between JVM, JDK, JRE

- JVM, JDK, JRE these all the backbone of java language.
- Each components have separate works.
- JDK and JRE physically exists but JVM are abstract machine it means it not physically exists.

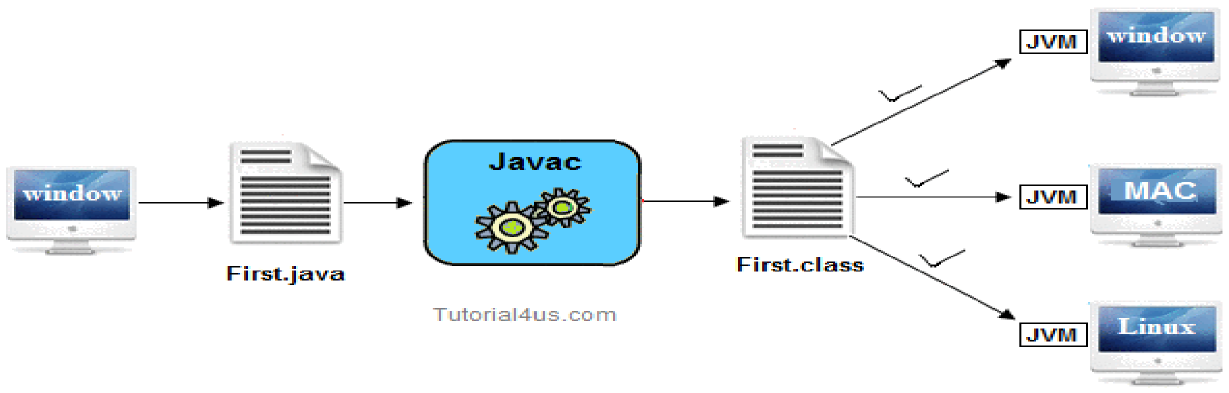

JVM
JDK
JRE
I provides runtime environment to execute bytecode.
I am physically exists and contains Java Tools, JVM, JRE
I contains set of libraries and tools
Sitesbay.com
Sitesbay.com
Tutorial4us

Features of Java



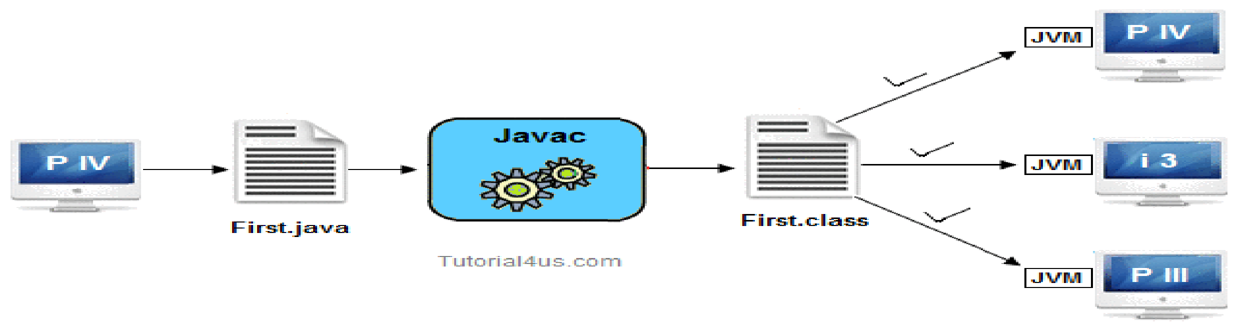
1. Platform Independent

A program or technology is said to be platform independent if and only if which can run on all available **operating systems**



2.Architectural Neutral

A Program or Technology is said to be Architectural neutral which can run on any available **processors** in the real world.



3.Object Oriented

- Object Oriented means we organize our software as a combination of **different types of objects** that incorporates both data and behavior.
- Object Oriented Programming is a methodology or paradigm to design a program using **classes and objects**.

Basic concepts of OOP:

- Object
- Class
- Inheritance
- Polymorphism
- Encapsulation
- Abstraction

What is Object?

- Object is the **physical** as well as **logical entity** whereas class is the only logical entity.
- An entity that has state and behavior is known as an object.
- Such as, chair, table, bike, marker, pen, car etc.
- An object has three characteristics:
  - **state**: represents data (**value**) of an object.
  - **behavior**: represents the behavior (**functionality**) of an object such as deposit, withdraw etc.
  - **identity**: Object identity is typically implemented via a unique **ID**. The value of the ID is not visible to the external user. But it is used internally by the JVM to identify each object uniquely

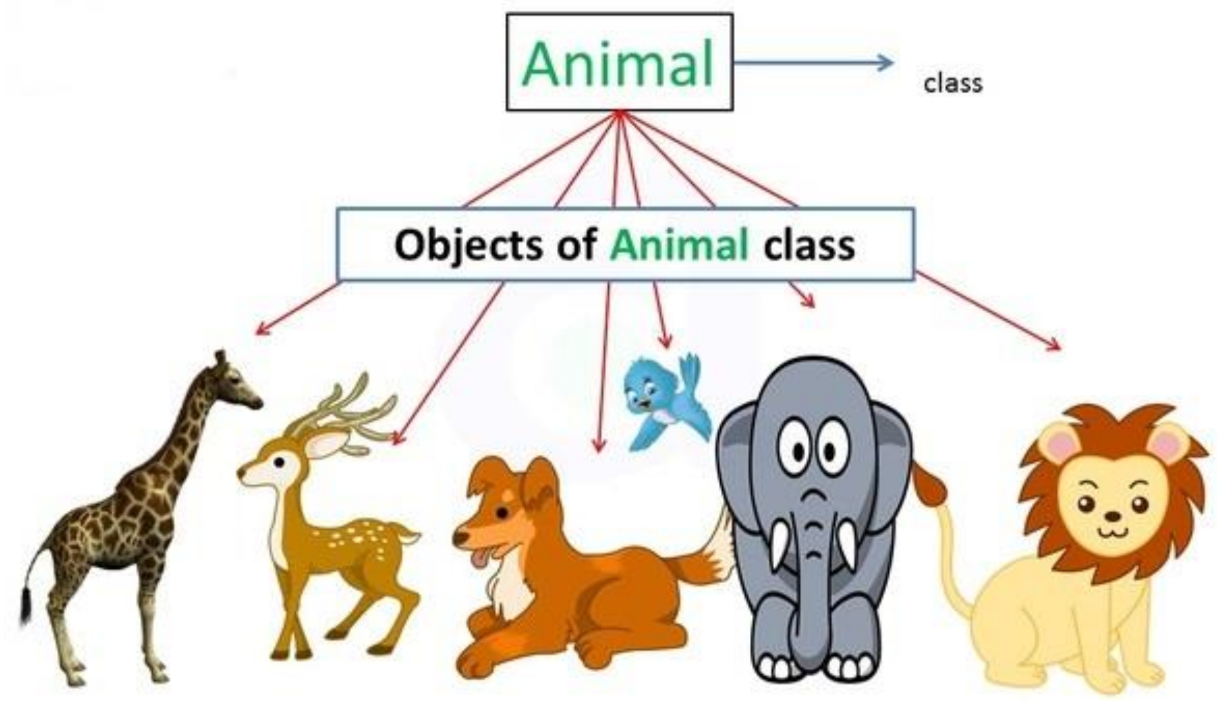
- **Object is an instance of a class**. Class is a template or blueprint from which objects are created. So, object is the instance(result) of a class.
- **Examples**:
  - **Object**: – Car
  - **State**: Color
  - **Behavior**: Accelerate, Slow Down

What is Class?

- A class is a **group of objects** that has common properties. It is a template or blueprint from which objects are created.

- A class in java can contain:
  - data member
  - method
  - constructor
  - block
- A class in declared using **class** keyword in java.

Class and Object in Java



Difference between C++ and Java is given below:

Comparison	**C++**	**Java**
Platform-independent	C++ is platform-dependent.	Java is platform-independent.
Mainly used for	C++ is mainly used for system programming.	Java is mainly used for application programming. It is widely used in Windows-based, web-based, enterprise, and mobile applications.
Design Goal	C++ was designed for systems and applications programming. It was an extension of the C programming language.	Java was designed and created as an interpreter for printing systems but later extended as a support network computing. It was designed to be easy to use and accessible to a broader audience.
Goto	C++ supports the goto statement.	Java doesn't support the goto statement.
Multiple inheritance	C++ supports multiple inheritance.	Java doesn't support multiple inheritance through class. It can be achieved by using interfaces in java.

Operator Overloading	C++ supports operator overloading.	Java doesn't support operator overloading.
Pointers	C++ supports pointers. You can write a pointer program in C++.	Java supports pointer internally. However, you can't write the pointer program in java. It means java has restricted pointer support in java.
Compiler and Interpreter	C++ uses compiler only. C++ is compiled and run using the compiler which converts source code into machine code so, C++ is platform dependent.	Java uses both compiler and interpreter. Java source code is converted into bytecode at compilation time. The interpreter executes this bytecode at runtime and produces output. Java is interpreted that is why it is platform-independent.
Call by Value and Call by reference	C++ supports both call by value and call by reference.	Java supports call by value only. There is no call by reference in java.
Structure and Union	C++ supports structures and unions.	Java doesn't support structures and unions.

Thread Support	C++ doesn't have built-in support for threads. It relies on third-party libraries for thread support.	Java has built-in thread support.
Documentation comment	C++ doesn't support documentation comments.	Java supports documentation comment (/** ... */) to create documentation for java source code.
Virtual Keyword	C++ supports virtual keyword so that we can decide whether or not to override a function.	Java has no virtual keyword. We can override all non-static methods by default. In other words, non-static methods are virtual by default.
unsigned right shift >>>	C++ doesn't support >>> operator.	Java supports unsigned right shift >>> operator that fills zero at the top for the negative numbers. For positive numbers, it works same like >> operator.
Inheritance Tree	C++ always creates a new inheritance tree.	Java always uses a single inheritance tree because all classes are the child of the Object class in

		Java. The Object class is the root of the inheritance tree in java.
Hardware	C++ is nearer to hardware.	Java is not so interactive with hardware.
Object-oriented	C++ is an object-oriented language. However, in the C language, a single root hierarchy is not possible.	Java is also an object-oriented language. However, everything (except fundamental types) is an object in Java. It is a single root hierarchy as everything gets derived from java.lang.Object.

Note

- Java doesn't support default arguments like C++.
- Java does not support header files like C++. Java uses the import keyword to include different classes and methods.

C++ Program Example

File: main.cpp

```
#include <iostream>
using namespace std;
int main() {
   cout << "Hello C++ Programming";
   return 0;
}
```

Output:

```
Hello C++ Programming
```

Java Program Example

File: Simple.java

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

Output:

```
Hello Java
```