Day1: Java Introduction

Prerequisites:

- Basics of any other programming language

Introduction of Java

- Java is one of the most popular programming languages in the world, it is an object-oriented programming language.
- The Java platform provides a complete environment for application development on desktops and servers and deployment in embedded environments.

- Java was developed by James Gosling in 1995, at Sun Microsystems which was later acquired by Oracle in 2010.
- It was originally called OAK after an OAK tree that stood outside Gosling's Office, later it was renamed "Green" and was finally renamed to "Java" inspired by Java Coffee, that's why its logo looks like a cup of coffee.

Java Edition

We have 3 editions of Java for building a different kinds of applications

1. Java Standard Edition (JSE)

This is the core Java platform, it is a specification, which contains the core libraries to develop standalone, networking, database, GUI, multithreaded types of applications.
In addition to the core API, the Java SE platform consists of the virtual machine,

development tools, deployment technologies and other class libraries and toolkit commonly used in Java application

2. Java Enterprise Edition (JEE)

The Java EE platform provides an API and runtime environment for developing and running large-scale, multi-tiered, scalable, reliable, and secure network applications.

3.Java Micro Edition (JME)

It is a subset of Java SE, designed used for microdevices and embedded development like mobile phones, sensors, micro-controller, TV set-top boxes etc.

> Note:- JSE is the basic foundation of the remaining 2 other editions.

> The latest version of Java is Java 17 which was released on 14-Sep 2021

Java has close to 10 million developers worldwide, currently, about 3 billion mobile phones run java, Now a day java is everywhere, which means more opportunities for us to get hired as a professional programmers.

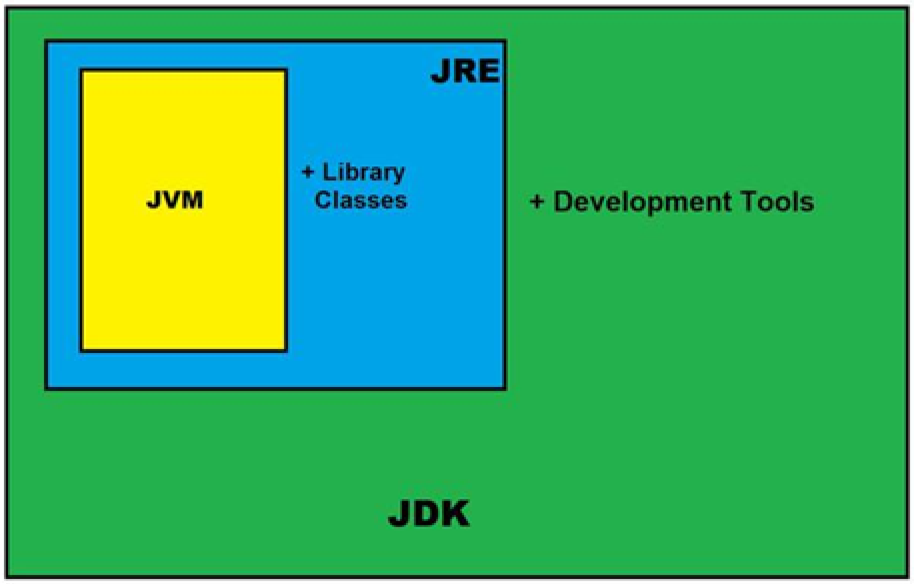
Features of Java

- **Simple:-** Java is easy to learn and its syntax is quite simple, clean and easy to understand.
- **Object-Oriented**:- Java is ***object-oriented***, it supports all the OOPS characteristics. This makes java applications easy to develop and maintain, compared to structured programming language.
- **Portable and Platform Independent:-** Java source code is compiled and converted into bytecode. this bytecode can run on multiple platforms i.e. Write Once and Run Anywhere(WORA), we can compile the java code in one Operating System and execute it on another Operating System.
- **Secure:-** JAVA has provided an implicit component inside JVM in the form of a "Security Manager" to provide implicit security against malicious code. Java has provided very good predefined implementations for almost all well-known network security. JAVA has provided a separate middleware service in JAAS [Java Authentication and Authorization Service], which provides web security. Auth, SSO.
- **Robust:-** Robust means strong. Java is having a very good memory management system in the form of a heap memory management system, it is a dynamic memory management system, it allocates and deallocates memory for the objects at runtime. JAVA is having very good Exception Handling mechanisms, because, Java has provided a very good predefined library to represent and handle almost all the frequently generated exceptions in java applications.
- **Multithreaded:-** Java supports multithreading to enhance performance. by using this we can execute multiple functionalities simultaneously.

JDK or Java SDK

Java Standard Edition comes in the form of Specification, and the implementation of this specification is the JDK software, also known as Java SDK (Java Standard Development Kit).

This JDK software is used for developing and executing Java applications.



JDK = (JRE + Development tools like java compiler, debugger, etc.)
JRE = (JVM + Predefined Library classes)

We Problem:

Downloading and Installing the JDK software:

We can download the JDK software from the Official website of Oracle.

https://www.oracle.com/java/technologies/downloads

After downloading we need to install this software as normal/ordinary software on our computer.

Select Java Editor:

We can develop our java application inside a normal text editor like a notepad and can execute

the application on the command prompt or terminal also. but in real-time to develop a java application we need to use IDE software (Integrated Development Environment).

Inside an IDE software, all the application development environments like editor, terminal, code intelligence, etc. are there in one place, with the help of an IDE software our application development speed and productivity will be improved.

There are many IDE software are there like:

IntelliJ IDEA

Eclipse

NetBeans

Downloading and Installing the IDE software (IntelliJ IDEA)

We can download the IntelliJ IDEA community edition from the following website:

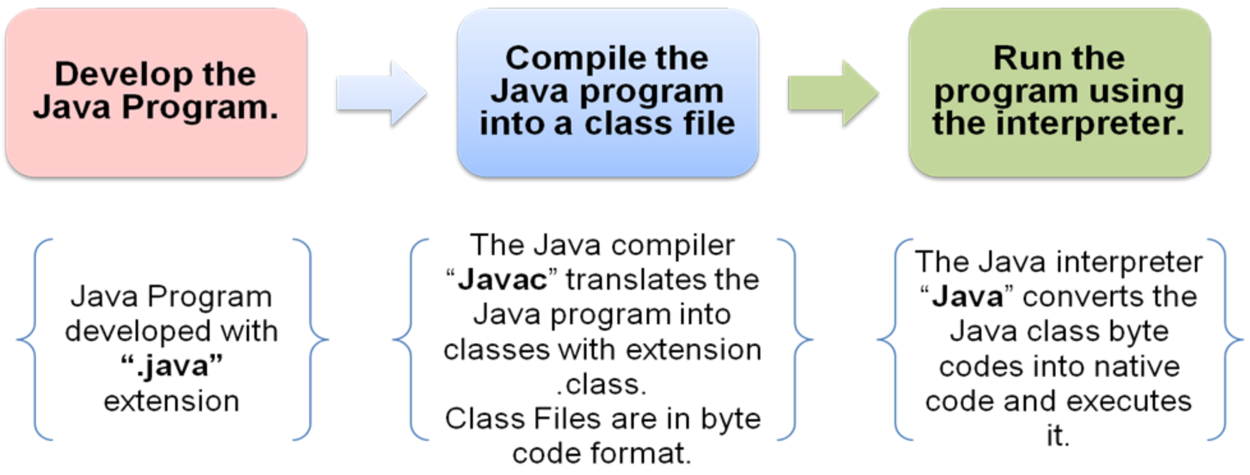
https://www.jetbrains.com/idea/

From here we can download the community edition of the IntelliJ IDEA (for Windows OS, we can download either a .exe file and install it as a normal software application or we can download a zip file and just unzip this file to make use of IntelliJ IDEA software).

To develop the Java application step by step in IntelliJ IDEA, you can refer to the following link:

https://www.jetbrains.com/help/idea/creating-and-running-your-first-java-application.html

Basic Steps To Develop a Java Program:


Develop the Java Program.
Compile the Java program into a class file
Run the program using the interpreter.
Java Program developed with ".java" extension
The Java compiler "Javac" translates the Java program into classes with extension .class. Class Files are in byte code format.
The Java interpreter "Java" converts the Java class byte codes into native code and executes it.

There are two phases are involved in Java application execution:

1. Compilation phase
2. Execution phase

In the Compilation phase, IntelliJ uses the java compiler to compile our code into a different format called Java **byte-code.**

This java compiler comes with the JDK software.

This java byte-code, is platform-independent, which means it can run on the Window, Mac, Linux, or any Operating-system that has **JRE (Java Runtime Environment).**

We can also download this JRE, for various Operating-system separately.

This JRE has a software component called **JVM (Java Virtual Machine)**, this JVM takes our java byte-code and translates it to the native code for the underlying OS.

If we are in the Window Operating-system, this JVM will convert our java byte-code to the window Operating-system understandable native code, and if we are in the Linux environment, then this JVM will convert our java byte-code to the Linux OS understandable native code. with this architecture only, our java applications are portable or platform-independent.

We can write a java program on a Window machine and can execute it in a Linux or macOS or any other OS that have JRE.

Basics of a Java Application

The smallest building block in java application is function/method.

A method is a block of code that performs a well-defined task. example: a method for adding 2 numbers, a method for validating user input, a method for calculating interest, etc.

the basic syntax of a method in Java application is:

```
returnType methodName(){
  --instructions--
}
```

Some method returns some value, whereas some method does not return any value.

the method which does not return any value we apply **void** as a return type.

void is a reserved keyword in java, the method name should be proper and descriptive.

Example:

```
void calculateInterest(){

}
```

We can pass some parameters to this method also, we use parameters to pass the value to our method. for example, amount, rate of interest, duration, etc.

and inside the pair of parenthesis { }, we write the actual implementation java code.

Every java program should have at least one method. and that method is called the main method.

The main method is an entry point of our java application.

Whenever we execute a java program, the main function gets called and the code inside this main method gets executed.

These methods don't exist as their own, they always belong to a class.

A class is like a container, of one or more methods.

We use the class to organize our code in the java application.

Every java program should have at least one class, which contains the main method.

I Problem

Our first java program:

Main.java

```
package com.masai;

public class Main{

  public static void main(String[] args){

    System.out.println("Hello World");
  }

}
```

Here class is a keyword, by which we define a class in java. we should give a proper descriptive name to the class.

In Java, all the classes and methods should have an access modifier.

An access modifier is a special keyword that determines if other classes and methods in this program can access these classes and methods.

We have various access modifiers are there like public, private, and so on. most of the time we use public access modifier.

So, the basic structure of a java program contains a class and inside the class, we have a main method.

To name our classes we use **P**ascal**N**aming**C**onvention(first letter of every word in uppercase
and to name our methods we use **c**amel**N**aming**C**onvenion.

Inside java programming, we have a concept called a package. we use the package to group related classes, so as our application grows, we are going to end up with many classes, so we should properly organize these classes inside different packages.

By convention, the base package for a java project is the domain name of your company in reverse.
it does not mean we should have an actual domain registered. this is just a way to create a namespace for our classes.

example: **com.massai;**

So every class we create in our java application should belong to a package. we are going to talk about packages in more detail in our upcoming session.

In the above example, **com.masai** is the base package of our project.

Note: All java files should have a **.java** extension. and every statement in the java application should be terminated with a semicolon ;

The **static** is a keyword, we will talk about this keyword later, for now, just remember the main method in our program should always be static. and the return type of this method is void, which means this method is not going to return any value.

In the main method, we have one parameter, we can use this parameter to pass values to our program. we will talk about this parameter in our upcoming session.

```
System.out.println("Hello World");
```

Here **System** is a predefined java class, which belongs to **java.lang** package. inside this class, we have various members, **out** is a member (field) who belongs to this System class.

the type of this **out** field is the **PrintStream class.** this PrintStream is another predefined class in java. the **println** method belongs to this PrintStream class.

So here we are calling or executing the println method inside our main method.

Inside the parenthesis of this println method, we can pass any value, which we want to print on the terminal or console.

Here “Hello World” is textual data, in java whenever we deal with textual data we should always surround them with double-quotes. which is known as a string.

In java, a string is a sequence of characters.

You Problem:

Write a java application that will print Your Name and Address in the following pattern:

```
This is My First Java Application
==================================
My Name is: Your Name
My Address is: Your Address
```