# Introduction to Java

## What is Java?

- Java is an Object Oriented Programming Language and a Computing Platform

## Object Oriented

Java uses the concept of class and object to create and maintain a software application

### OOP Building Blocks

- Object Oriented Programming is a type (paradigm/concept) of programming
- This concept consists of the following building blocks:
    - Class
    - Object
    - Encapsulation
    - Data hiding/protection
    - Inheritance
    - Association
    - Polymorphism
    - Abstraction

### Programming Language

It is a formal language that follows some [rules] and [conventions] to achieve an outcome.

### Computing Platform

Java provides a Java Runtime environment to run and manage java based applications.

## Java History

- Java was developed by Sir James Gosling at Sun Microsystems.
- Initially it was developed to solve a problem with multi-platform consumer devices.
- Work on Java began in 1991.
- Java was released in 1995.
- Oracle acquired Sun on 20th April, 2009
- As Java is Open Source, many companies apart from Oracle are creating their own flavors of Java.

## Java Editions & Versions

There are 3 editions of Java: JSE, JEE and JME

### JSE

- JSE stands for Java Platform, Standard Edition
- It is the core language.
- It is in the core of all other editions

- Solely it can be used to develop desktop applications
- The latest version of JSE is JDK 23.x.x
- The latest LTS version of JSE is JDK 21.x.x

JEE
---
There are 2 JEE platforms
- Java Platform, Enterprise Edition
- Jakarta Platform, Enterprise Edition

Java EE was maintained by Oracle under the Java Community Process.

On September 12, 2017, Oracle Corporation announced that it would submit Java EE to the Eclipse Foundation.

From here on Java EE was renamed to Jakarta EE.
- The last version of Java EE was Java EE 8.
- The first version of Jakarta EE was Jakarta EE 8.
- The latest version of Jakarta EE is Jakarta EE 10.

- It is used to develop web and enterprise applications
- Spring Framework uses JEE under the hood

JME
---
- JME stands for Java Platform, Micro Edition
- It is used for the development and deployment of portable code for embedded and mobile devices (micro-controllers, sensors, gateways, mobile phones, personal digital assistants, TV set-top boxes, printers).
- The latest version is JME 8

Java is Open Source
==========
Open Source Software:
1. Is free to download and use
2. Is developed by contributors for free
3. The code is shared with license
-----X-----X-----X-----X-----
Dev Environment Setup
==========
We need:
1. Any JDK version (preferably > JDK 8)
2. Eclipse
3. Notepad++

First Java Program
==========
Rules to create a basic Java program

1. In Java all the code is written inside a class
2. The class name should be in upper camel case. Ex: FirstJavaProgram
3. First define the scope of class then write the code inside {}
4. The file name SHOULD be same as the class name.
5. The extension of the file should be .java

```
6. Inside the class define a method named main(){}
7. The signature MUST be: public static void main(String[] args)
8. The execution of code starts from main()
9. We have to look out for following 3 things:
            1. Syntax : Rules of the language
            2. Semantics : Logic of the program
            3. Conventions : Best Practices of programming

Additional Rules
================
- The file name MUST be same as the name of the class if the class is
public
- One file can contain multiple classes, but
- One file can contain only one public class

Conclusion:
1. One file can contain multiple claases, but it is a good practice to
have only one class per file.
2. The name of the file and that of the class will be same.

Code Execution
==============
Every Java programs runs in 2 steps: compilation and execution

Step1: Compile the source code
------------------------------
- The program that we write as a programmer is known as Source code
(.java)

To compile:
- Command: javac <name of file>.java
- Example: javac Ex01FirstJavaProgram.java

- Compilation is done by Compiler.
- Compiler converts java source code (.java) into bytecode (.class)
- It checks the syntax of source code.

Step2: Run the executable file (.class)
---------------------------------------
- The file that we get post compilation has the extension .class
- Java Runtime executes the .class file

To execute:
- Command: java <name of the executable>
- For example: java Ex01FirstJavaProgram

JDK, JRE, JVM
=============
JDK
---
- JDK stands for Java Development Kit
- It is a software development environment used to develop Java
applications.
- It includes:
```

1. Java Runtime Environment (JRE),
2. Interpreter/loader (java),
3. Compiler (javac)
4. Archiver (jar)
5. Documentation generator (Javadoc) and
6. Other tools needed in Java development.

JRE
---
- JRE stands for Java Runtime Environment.
- Sometimes it is also referred as Java RTE
- The Java Runtime Environment provides the minimum requirements for executing a Java application.
- It consists of the JVM, core classes, and supporting files.

JVM
---
- JVM stands for Java Virtual Machine
- When we execute a java program, a Runtime instance of JVM executes the java byte-code

Summary
-------
JDK = JRE + Development Tools
JRE = JVM + Runtime Library

The Process
==========
- First we need to install a machine specific JVM on the system where we wish to run the code.
- We create a java program, compile it and get an executable (.class) file that contains the byte code understood by the JVM.
- Now this executable file will be converted into machine code and run/executed by the Java Runtime System on Java Virtual Machine.

Refer Image : 01_compile_execute.png

So, there are 2 processes
- Compilation: Source code is converted into bytecode (.java => .class)
- Execution: compiled code runs (.class => run)

Note:
- Always .class files run.
- .java files never run

Note:
Java is platform independent
JVM is platform dependent
-----X-----X-----X-----
Variables
=========
- Its all about data and how we process data.
- A software application's task is to work on data.

```
So, in a proram
- Data is stored in => Variables.
- Data (in variables) is processed by => Methods

Variable Declaration
--------------------
- We have to declare a variable well in advance.
- The declaration contains
      - Mentioning the data type of the variable
      - Providing a proper name to the variable

Ex: double discountedPrice = 499.99

- We can't use a variable if it not declared and initialized

Data Types
==========
- We MUST specify the data type of every variable that we want to use
- We cannot redeclare a variable with another type
- This is known as static typing.
- Java is a statically and strongly typed language

Static languages ex: Java, C, C++
Dynamically/loosely typed languages: JavaScript, Python

ex:

let number1 = 10;

number1 = 10.1;

number1 = "hi";

There are 2 types of data types:
1. Primitive data type
2. Reference data type

Primitive data types
--------------------
There are 8 primitive data types in java:

1. Integer data type
- There are 5 Integer data type: byte, short, int, long, char
- These are used to hold integer values of different range
- This is how variables are declared:

byte b = 1;
short s = 2;
int i = 4;
long l = 10;

2. Character data type
- It is used to hold a single character
char c = '@';
```

```
3. Decimal data Type
- There are decimal types: float, double
- This is how variables are declared:

float f = 1.2f;
float f = 1.2F;
double d = 1.2;

4. Boolean data type
- It is used to hold boolean values
boolean isPassed = true;
boolean isAbsent = false;

Refer images: 02_data_types.png, 03_integers.png, 04_double.png,
05_char_bool.png

More on data types
------------------
- Any number written without a decimal is automatically considered int.

- Any number written with a decimal is automatically considered double.
- During an operation, the type of result will change according to
biggest data type.
  ex : 2 * 3.14 * 2.1f => double (as 3.14 is automatically considered
double)
- A smaller data type variable cannot store a bigger type data value
- All the data types arranged from smaller to bigger type:
      byte < short < char < int < long < float < double
      boolean is not numerical, hence it is not in this hierarchy

Literals
--------
The value that we assign to variables are known as Literals.
Ex:
1. Integer Literal : 2
2. Decimal Literal : 1.2
3. Boolean Literals : true and false
4. Character Literal : '@'
5. String Literal : "Hi, how are you?"

Type Casting
------------
- We cannot assign data of bigger type into variable of smaller data
type.
For ex: double d = 1.2;
int i = d; // not allowed

- Double is bigger than float and all decimal numbers are automatically
considered double
for ex: float num = 1.2; // it will give error as 1.2 is of type double

- Hence we have to manually/explicitly tell Java that 1.2 is not double
by appending f/F to it
```

```
for ex: float num = 1.2f;

Reference data type
==================
1. Class
2. Interface
3. Enum
4. Array

We will learn about reference data types in topics to come.
-----X-----X-----X-----
Keywords
========
These are pre-defined, reserved words that have special meanings to the
compiler.

There are 65+ keywords in java.

Ex: class, public, static, void, int, float etc

Identifiers
===========
The names that we give to classes, interfaces, objects, variables,
methods, enums are called identifiers

Rules:
1. We can use a-z, A-Z, 0-9, _, $ as an identifier
2. The identifier cannot start with a number
3. We can create an identifier of any length
4. Space is not allowed in a name
5. Keywords cannot be used as identifiers

Ex: dellLaptop_001Screen, _productId,
$price,
addressOfTheCompanyWhoBoughtTheLaptopForItsOfficeWork

can't use: 001_dell, total price

Naming Conventions
==================
There are some conventions that the programmer must follow to name
identifiers such as class, package, variable, constant, method etc.

Note : Convention, not Rule

Why use naming conventions?
---------------------------
- By using standard Java naming conventions, we make our code easier to
read for ourselves and for other programmers.

- Readability of Java program is very important.

- It indicates that less time is spent to figure out what the code does.
```

```
- But, it is not forced to follow. So, it is known as a convention and
not rule.

Conventions
-----------
1) The name of a class
      - should start with uppercase letter
    - If the name is created with multiple words, each word must start
with a capital letter (Upper Camel Case)
      - Should be a noun
      - Ex: String, System, Thread, FirstProgram, LayoutManager etc.

2) The name of an interface
      - Same naming convention as that of class.
      - should be an adverb
      - Ex:  Runnable, Remote, ActionListener etc.

3) The name of a method
      - should start with lowercase letter
      - if the name is created with multiple words, the first letter of
1st word should be in smaller case and then each subsequent word must
start with a capital letter (Lower Camel Case)
      - should be a verb
      - Ex: doSomething(), main(), print(), println() etc.

4) The name of a variable
      - Same naming convention as that of method (Lower Camel Case)
      - should be a noun
      - Ex: firstName, orderNumber etc.

5) The name of package
      - should be in lowercase
      - Ex: java.lang, java.sql, java.util etc.

6) The name of constants (final variables):
      - Should be in uppercase letters
      - For a multi word constant, each subsequent word must be separated
by an underscore
      - Ex:  RED, YELLOW, MAX_PRIORITY etc.

Note: Java is a Case sensitive language, main() and Main() are 2
different methods
-----X-----X-----X-----
Data Input
==========
What is the use of any software?
- Process data

Different operations that take place on data:
1. Data Input
2. Data Transfer
3. Data Processing
4. Data Storage
5. Data Output
```

```
We are just developing "simple" applications, so

As of now consider that there are following 3 ways to input data:
1. Hard Coded Input
2. Scanner Input
3. Command Line Input

Hard Coded Input
----------------
Hard Coded Data is for static data

Data is set in the variables in source code.
ex: int number = 10;

Command Line Input
------------------
Command Line Arguments is for dynamic input

At one time only 1 value can be assigned to a variable
String name = "Kaustubh";

So, we can use arrays to hold multiple values at once
String[] names = {"Jay", "Yash", "Raj"};

names[0] => "Jay"
names[1] => "Yash"
names[2] => "Raj"

Similarly,
String[] args = {"12", "32.2", "1.009"};

Inorder to convert a numerical string ("10000") into a number, we use
certain methods

Use parse methods to parse String data passed from command line

Integer.parseInt() : it is used to convert an integer string to int data
type
Double.parseDouble() : it is used to convert a double string to double
data type

java ProcessData.java 12 23.2

int n1 = Integer.parseInt(args[0]);
double n2 = Double.parseDouble(args[1]);

3. Scanner Input is for dynamic input
      - Use Scanner class present in java.util package
      - Scanner class has methods that read data from keyboard
      - Scanner is a class that scans the input from various sources
      - Inorder to scan keyboard input, we do: Scanner s = new
Scanner(System.in);
      - Inorder to receive an integer, we use s.nextInt();
```

```
      - Inorder to receive a double, we use s.nextDouble();
-----X-----X-----X-----X-----
```