Introduction to Java

Module-1

Dr.M. Sivakumar
Professor & Technical Trainer / CSE
Saveetha School of Engineering, SIMATS, Chennai-602 105
sivakumarm.sse@saveetha.com
+91-9500600868

Outline

- ✓ Introduction to Java
- ✓ Features of Java
- ✓ Components of Java
  - JDK
  - JRE
  - JVM
- ✓ Data Types
- ✓ Variable

3 Billion

devices run Java—in your home, your car, and your office.

12 Million

Java developers worldwide.

JAVA

- Java is a general-purpose computer-programming language that is open source, platform independent, object-oriented and specifically designed to have as few implementation dependencies as possible.
- Java was originally developed by **James Gosling** at Sun Microsystems and released in 1995.
- Java was initially named as Oak language and renamed to JAVA in 1995.

Current Version	Java SE 15 (as of feb-2021)
Version we will use	Java SE 11 (LTS)
Setup size	149 MB (Linux), 152 MB (Windows x64)
Download Link	https://www.oracle.com/in/java/technologies/javase-jdk11-downloads.html
Official Website	https://java.com
Integrated **D**evelopment **E**nvironment (IDE)	1. Eclipse (going to use this IDE in later chapters) 2. NetBeans 3. IntelliJ IDEA Community Edition 4. BlueJ

Features of JAVA

Simple: Java inherits C/C++ syntax and many object-oriented features of C++.

Object Oriented: “Everything is an object” paradigm, which possess some state, behavior and all the operations are performed using these objects.

Robust: Java has a strong memory management system. It helps in eliminating error as it checks the code during compile and runtime.

Multithreaded: Java supports multiple threads of execution, including a set of synchronization primitives. This makes programming with threads much easier.

Features of JAVA (Cont.)

Architectural Neutral: Java is platform independent which means that any application written on one platform can be easily ported to another platform.

Interpreted: Java is compiled to bytecodes, which are interpreted by a Java run-time environment.

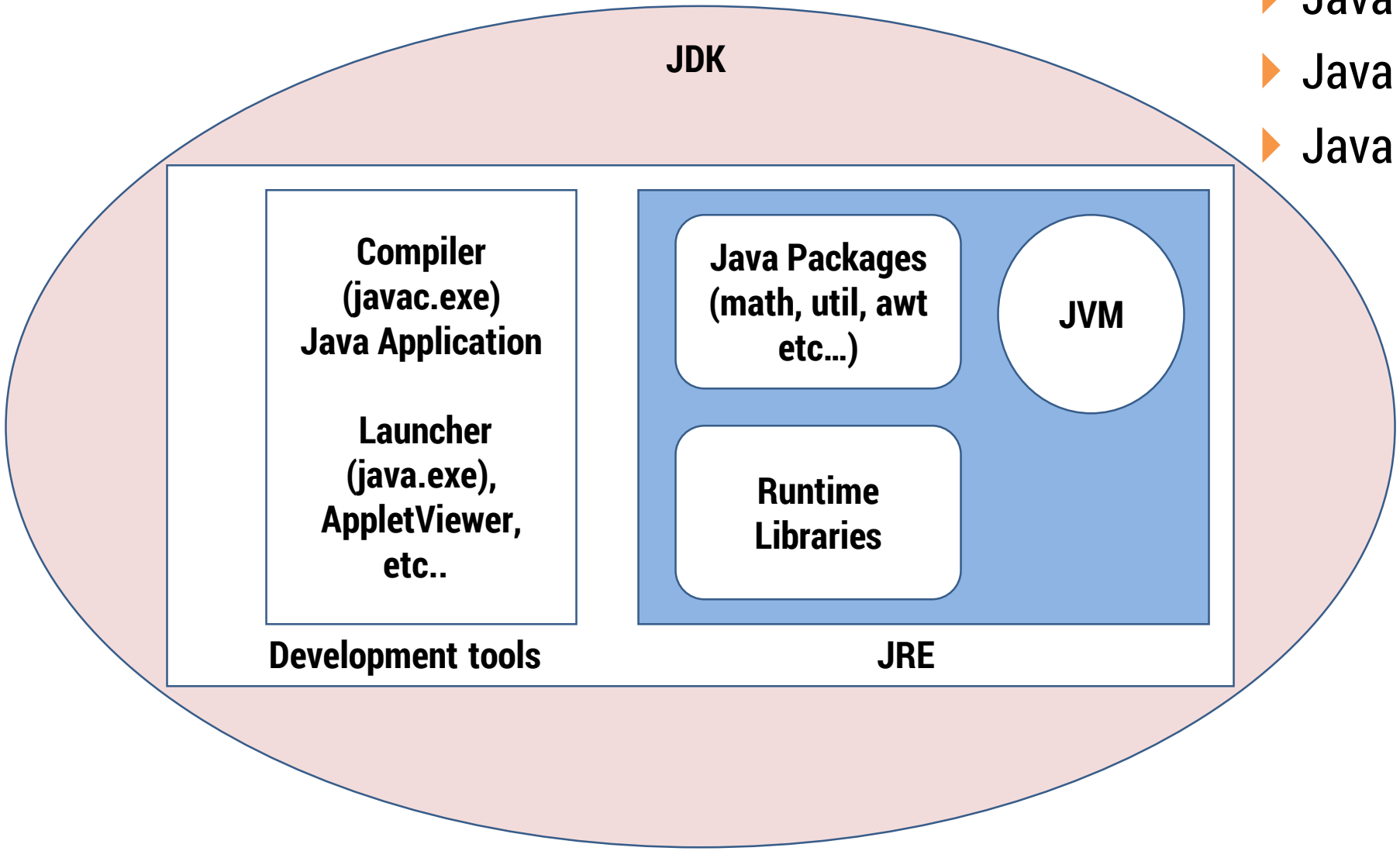
High Performance: Java achieves high performance through the use of bytecode which can be easily translated into native machine code. With the use of JIT (Just-In-Time) compilers, Java enables high performance.

Features of JAVA (Cont.)

Distributed: Java provides a feature which helps to create distributed applications. Using Remote Method Invocation (RMI), a program can invoke a method of another program across a network and get the output. You can access files by calling the methods from any machine on the internet.

Dynamic: Java has ability to adapt to an evolving environment which supports dynamic memory allocation due to which memory wastage is reduced and performance of the application is increased.

Components of Java



- Java Virtual Machine (JVM)
- Java Runtime Environment (JRE)
- Java Development Kit (JDK)

Java Development Kit (JDK)

- JDK contains tools needed ,
  - To develop the Java programs and
  - JRE to run the programs.
- The tools include
  - compiler (javac.exe),
  - Java application launcher (java.exe),
  - Appletviewer, etc…
- Java application launcher (java.exe) opens a JRE, loads the class, and invokes its main method.

Java Runtime Environment (JRE)

- The JRE is required to run java applications.
- It combines the Java Virtual Machine (JVM), platform core classes and supporting libraries.
- JRE is part of the Java Development Kit (JDK), but can be downloaded separately.
- It does not contain any development tools such as compiler, debugger, etc.

Java Virtual Machine (JVM)

- JVM is a virtual machine that enables a computer to run Java programs as well as programs written in other languages and compiled to Java Bytecode.
- Bytecode is a highly optimized set of instructions designed to be executed by the Java Virtual Machine(JVM).
- Byte code is intermediate representation of java source code.
- Java compiler provides byte code by compiling Java Source Code.
- Extension for java class file or byte code is '.class', which is platform independent.
- JVM is virtual because , It provides a machine interface that does not depend on the operating system and machine hardware architecture.
- JVM interprets the byte code into the machine code.
- **JVM** itself is **platform dependent**, but **Java** is **Not**.

Java Virtual Machine (JVM)

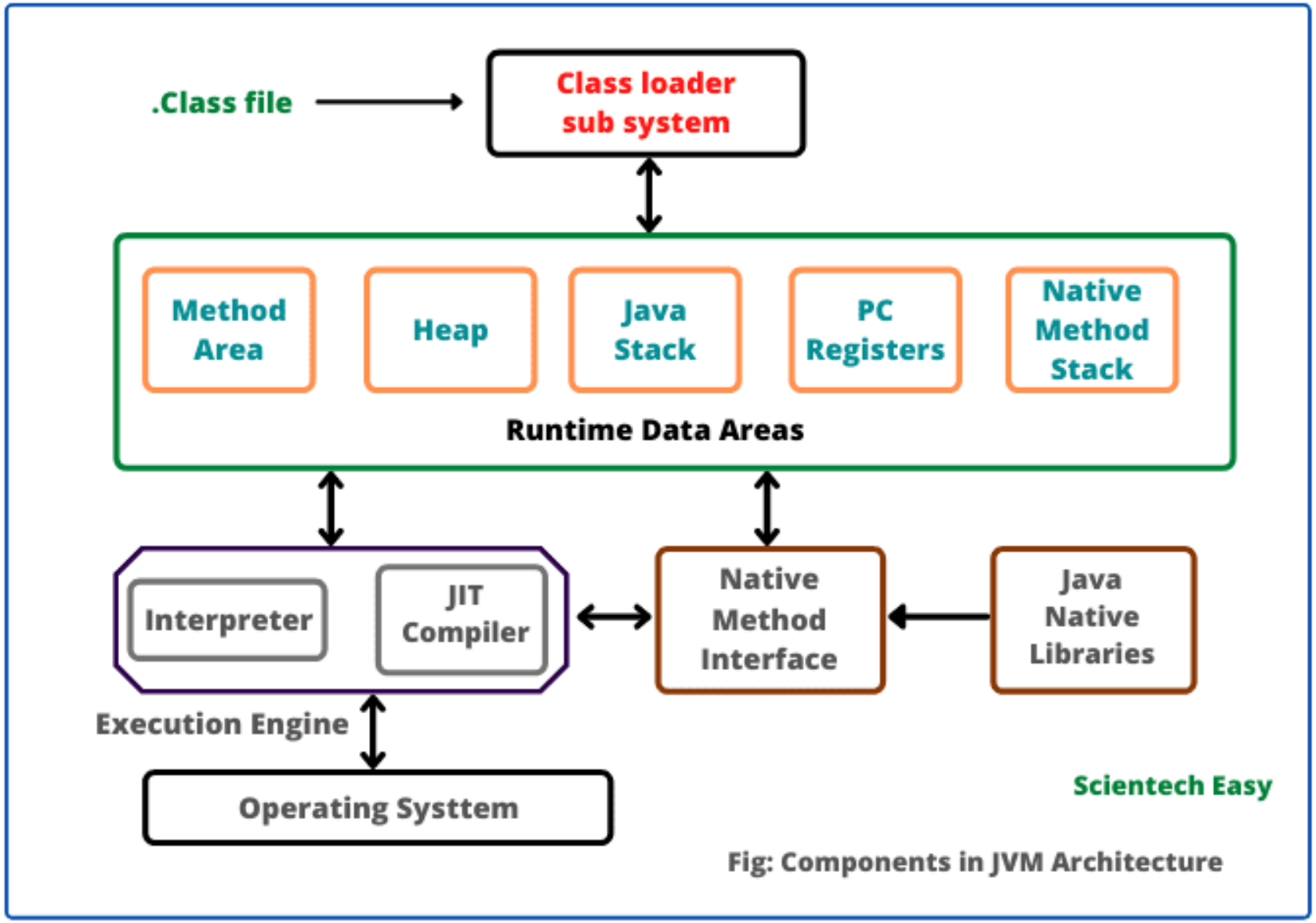


Fig: Components in JVM Architecture

Java Virtual Machine (JVM)

JVM contains the following main components that are as follows:

- Class loader sub system
- Runtime data areas
- Execution engine
- Native method interface
- Java native libraries
- Operating system

Java Virtual Machine (JVM)

How JVM works Internally?

Java Virtual Machine performs the following operations for execution of the program.

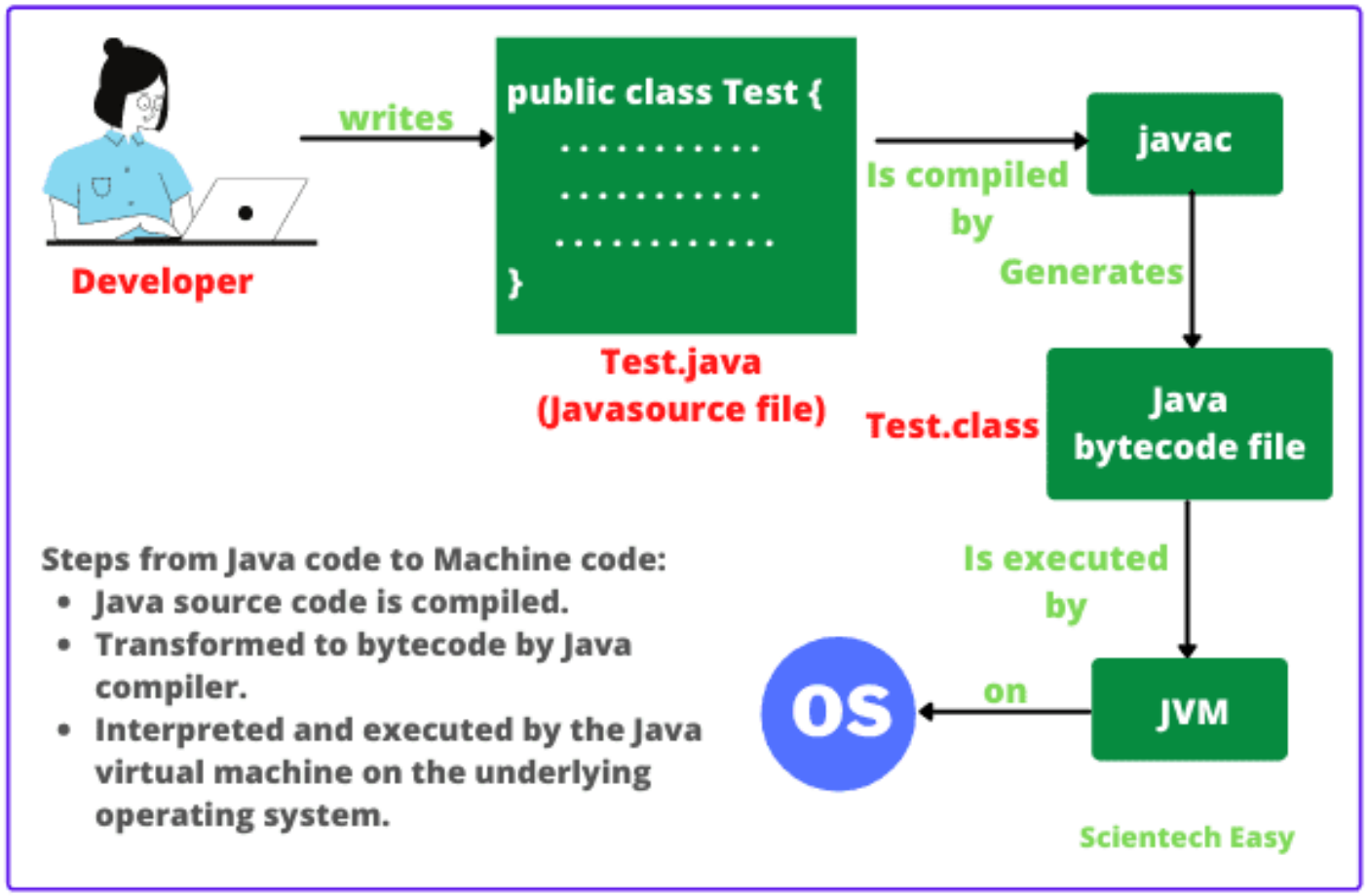
They are as follows:

a) Load the code into memory.

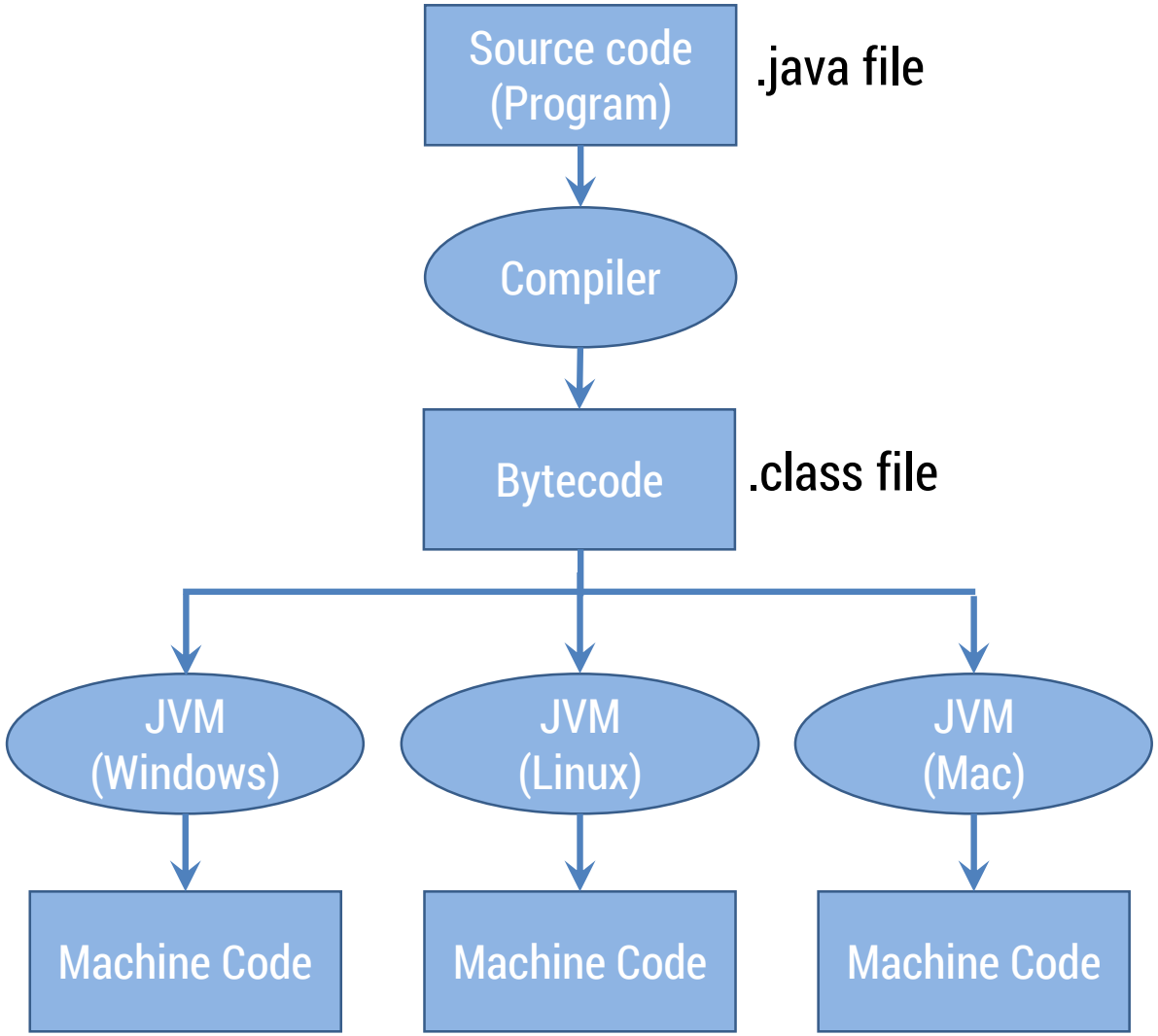
b) Verifies the code.

c) Executes the code

d) Provides runtime environment.



How Java become Platform Independent?



Installing JDK

- Download JDK for Windows platform (.exe) from
  - https://www.oracle.com/technetwork/java/javase/downloads/index.html
- Install the executable of JDK
- Set the path variable of System variables by performing following steps
  - Go to "System Properties" (Right click This PC → Properties → Advanced System Settings)
  - Click on the "Environment variables" button under the "Advanced" tab
  - Then, select the "Path" variable in System variables and click on the "Edit" button
  - Click on the "New" button and add the path where Java is installed, followed by `\bin`. By default, Java is installed in `C:\Program Files\Java\jdk-11.0.1` (If nothing else was specified when you installed it). In that case, You will have to add a new path with: `C:\Program Files\Java\jdk-11.0.1\bin`
  - Then, click "OK", and save the settings
  - At last, open Command Prompt (cmd.exe) and type java -version to see if Java is running on your machine

Setting Path Variable

System

Control Panel › System and Security › System

Search Control Pa

Control Panel Home

Device Manager

Remote settings

System protection

Advanced system settings

View basic information about your computer

Windows edition

Windows 10 Pro

© 2018 Microsoft Corporation. All rights reserved.

System

1

System variables

Variable	Value
Path	C:\Program Files (x86)\Intel\iCLS Client\;C:\Program Files\Intel\iCL...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 158 Stepping 9, GenuineIntel
PROCESSOR_LEVEL	6
PROCESSOR_REVISION	9e09
PSModulePath	%ProgramFiles%\WindowsPowerShell\Modules;C:\WINDOWS\syst...

New... Edit... Delete

3

OK Cancel

System Properties

Computer Name | Hardware | Advanced | System Protection | Remote

You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

Settings...

User Profiles

Desktop settings related to your sign-in

Settings...

Startup and Recovery

System startup, system failure, and debugging information

Settings...

2

Environment Variables...

OK Cancel Apply

Edit environment variable

C:\Program Files (x86)\Intel\iCLS Client\

C:\Program Files\Intel\iCLS Client\

%SystemRoot%\system32

%SystemRoot%

%SystemRoot%\System32\Wbem

C:\Program Files\Java\jdk-11.0.1\bin

4

New

Edit

Browse...

Delete

Hello World Java Program

File must be saved as HelloWorld.java

Main method from where execution will start

String must start with capital letter

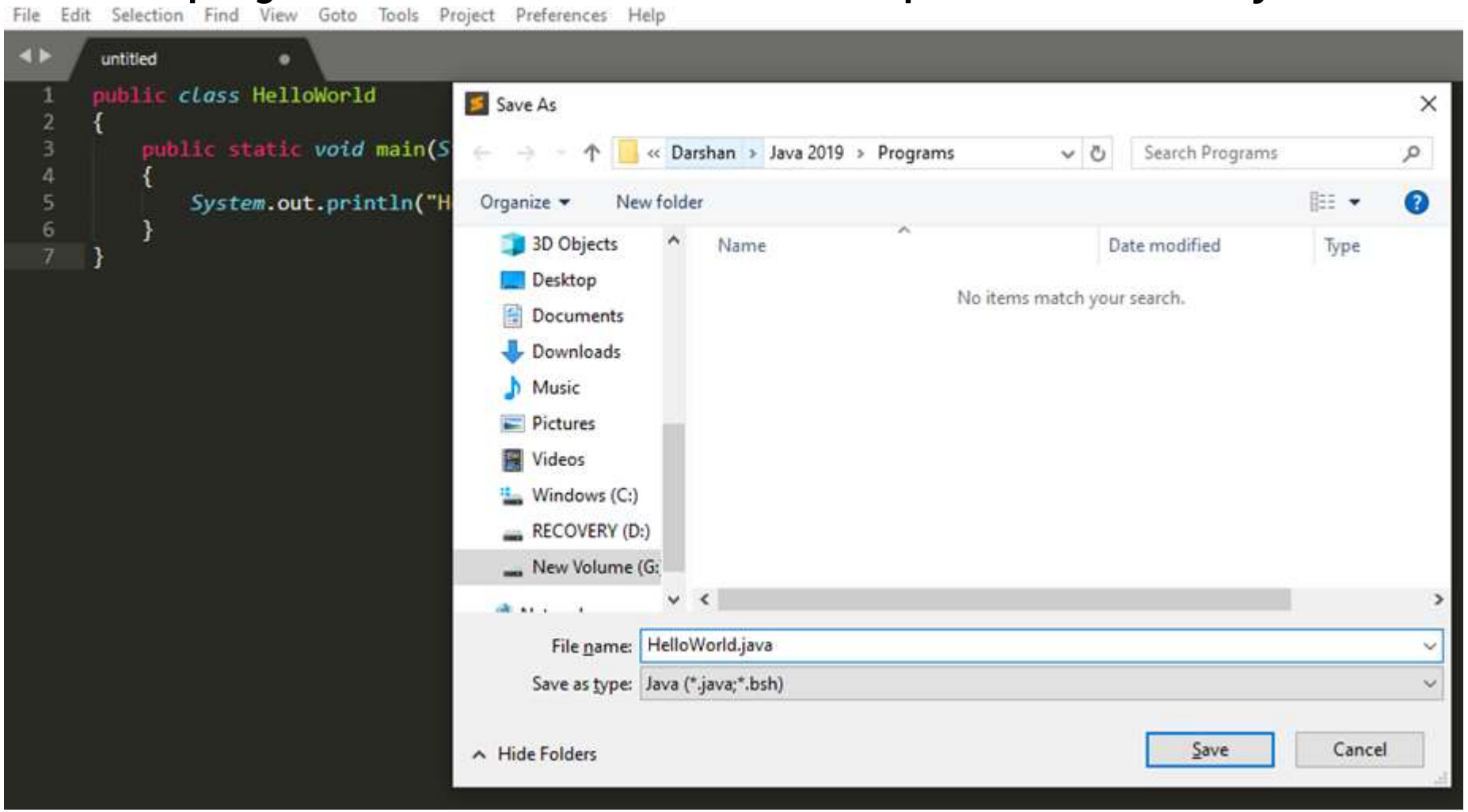
```
public class HelloWorld
{
  public static void main(String[] args)
  {
      System.out.println("Hello World");
  }
}
```

System must start with capital letter

- We have to save this in HelloWorld.java file as it has public class named HelloWorld.
- String and System are inbuilt Java Classes.
- Classes in java are always written in Camel case.

How to execute Java Program?

1. Save the program with the same name as the public class with **.java** extension.



How to execute Java Program?

2. Open command prompt (cmd) / terminal & navigate to desired directory / folder.

```
C:\WINDOWS\system32\cmd.exe

G:\Darshan\Java 2019\Programs>_
```

3. Compile the ".java" file with **javac** command.

```
C:\WINDOWS\system32\cmd.exe

G:\Darshan\Java 2019\Programs>javac HelloWorld.java
```

4. Execute the ".class" file with **java** command without extension.

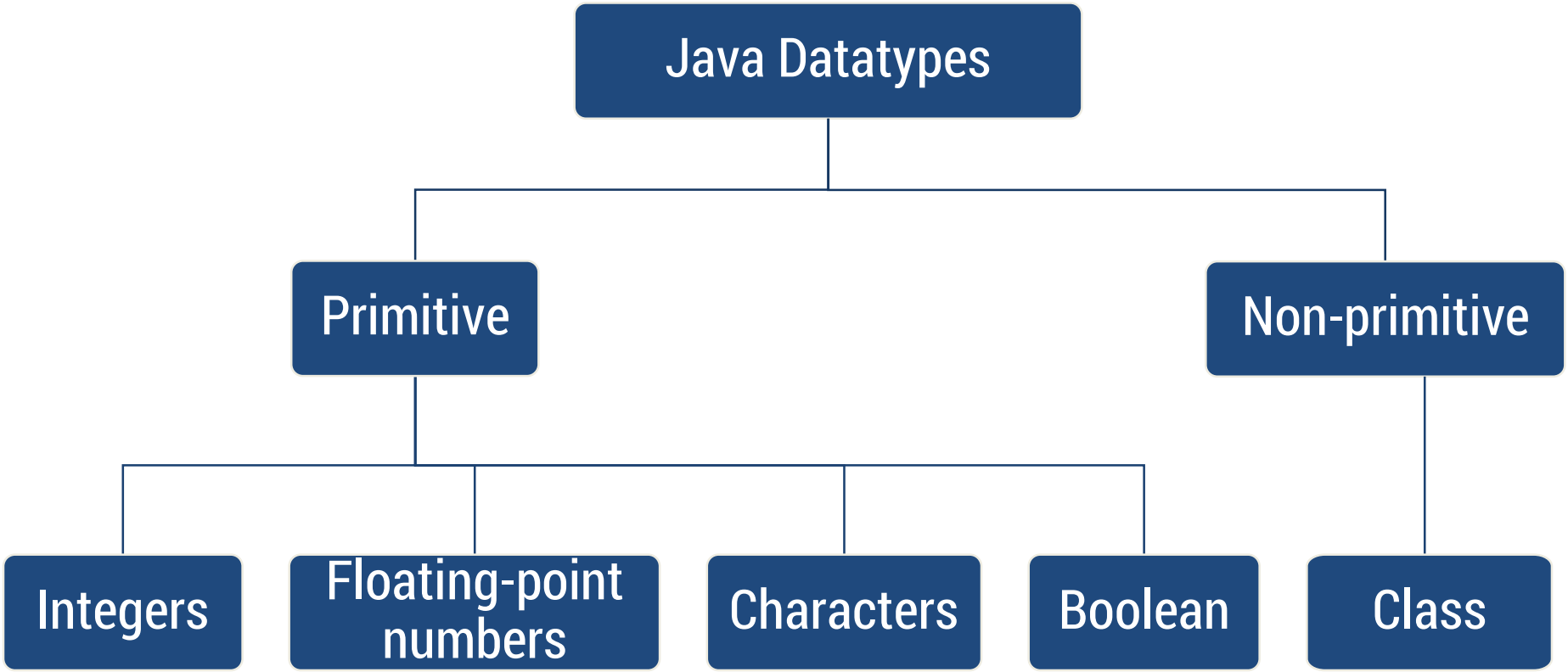
```
C:\WINDOWS\system32\cmd.exe

G:\Darshan\Java 2019\Programs>java HelloWorld
Hello World
```

Identifiers

- They are used for class names, method names and variable names.
- An identifier may be any descriptive sequence of
  - uppercase(A…Z) and lowercase(a..z) letters
  - Numbers(0..9)
  - Underscore(_) and dollar-sign($) characters
- Examples for valid Identifiers,
  - AvgTemp
  - count
  - a4
  - $test
  - this_is_ok
- Examples for invalid Identifiers,
  - 2count (Identifiers can not start with digit)
  - High-temp (Identifiers can not contain dash)
  - Ok/NotOK (Identifiers can not contains slash)

Data Types


Java Datatypes
Primitive
Non-primitive
Integers
Floating-point numbers
Characters
Boolean
Class

Primitive Data Types

Data Type	Size	Range	Example
byte	1 Byte	-128 to 127	byte a = 10;
short	2 Bytes	-32,768 to 32,767	short a = 200;
int	4 Bytes	-2,147,483,648 to 2,147,483,647	int a = 50000;
long	8 Bytes	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807	long a = 20;
float	4 Bytes	1.4e-045 to 3.4e+038	float a = 10.2f;
double	8 Bytes	4.9e-324 to 1.8e+308	double a = 10.2;
char	2 Bytes	0 to 65536 (Stores ASCII of character)	char a = 'a';
boolean	Not defined	true or false	boolean a = true;

Escape Sequences

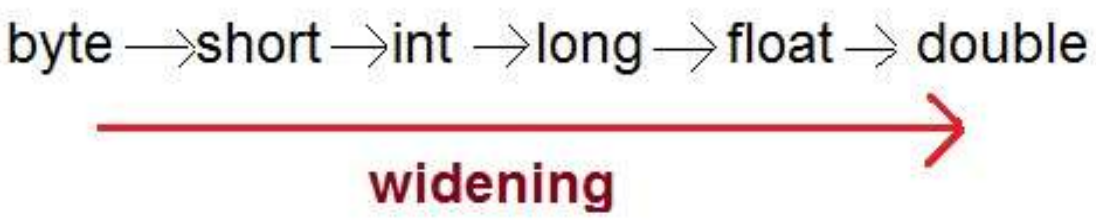
- Escape sequences in general are used to signal an alternative interpretation of a series of characters.
- For example, if you want to put quotes within quotes you must use the escape sequence, \", on the interior quotes.

```
System.out.println("Good Morning \"World\"");
```

Escape Sequence	Description
\’	Single quote
\”	Double quote
\\	Backslash
\r	Carriage return
\n	New Line
\t	Tab

Type Casting

- Assigning a value of one type to a variable of another type is known as Type Casting.
- In Java, type casting is classified into two types,
  - Widening/Automatic Type Casting (Implicit)



  - Narrowing Type Casting(Explicitly done)

Automatic Type Casting

- When one type of data is assigned to other type of variable , an *automatic type conversion* will take place if the following two conditions are satisfied:
  - The two types are compatible
  - The destination type is larger than the source type
- Such type of casting is called "*widening conversion*".
- Example:

  int can always hold values of byte and short

```
public static void main(String[] args) {
    byte b = 5;
    // √ this is correct
    int a = b;
}
```

Variables

When programming it is often necessary to store a value for use later on in the program.

A variable is a label given to a location in memory containing a value that can be accessed or changed.

Think of a variable as a box with a label that you can store information in.

Variables

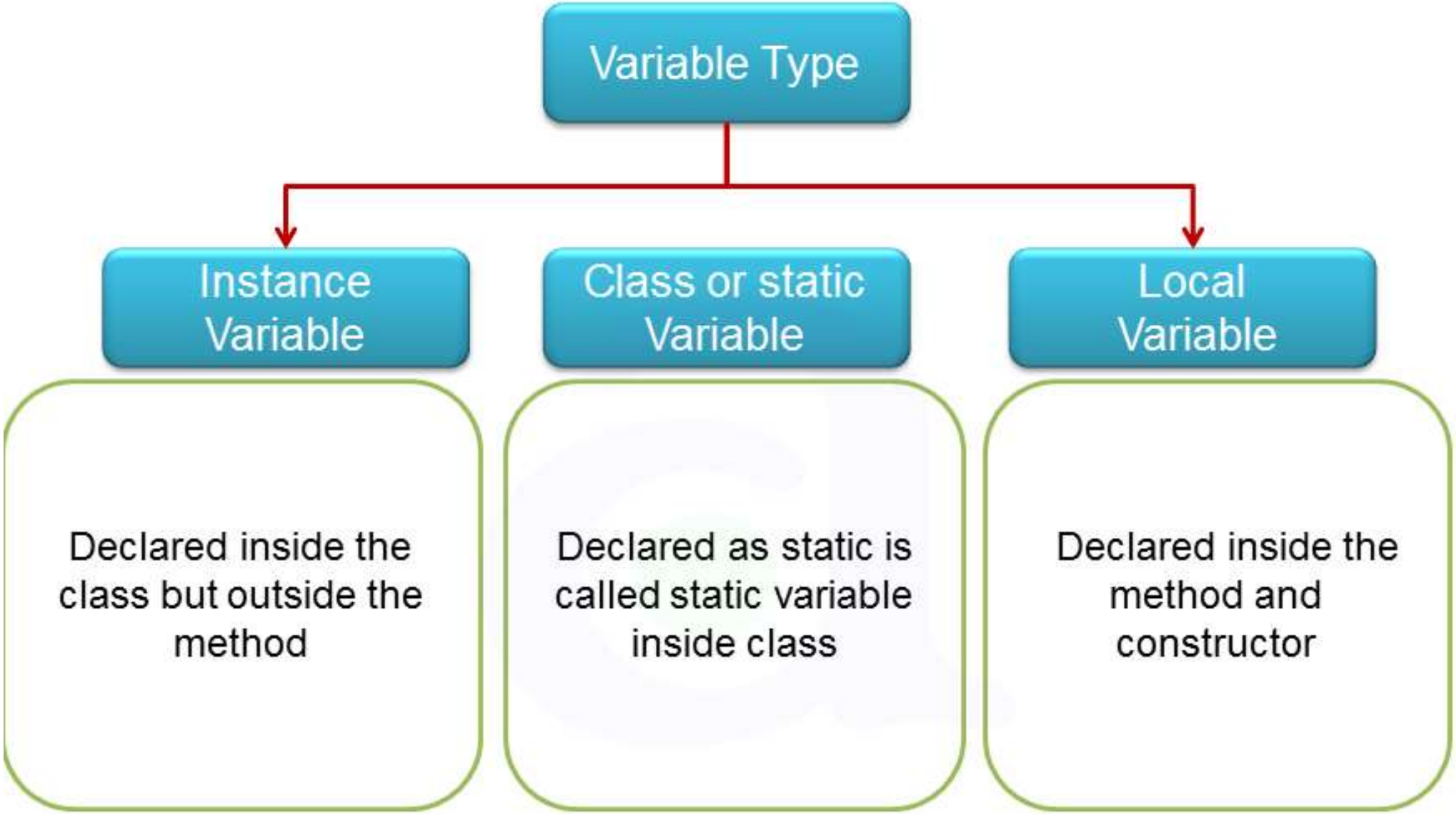
- **Variable** is an identifier which holds data or another one variable.
- It is an identifier whose value can be changed at the execution time of program.
- It is used to identify input data in a program.
- Declaration and Initialization

```
int  age ;    // declaration

int rollno=50;   // Initialization

age = 20;     // Assignment
```

Variables


Variable Type
Instance Variable
Class or static Variable
Local Variable
Declared inside the class but outside the method
Declared as static is called static variable inside class
Declared inside the method and constructor

Variables

Local Variable

- A variable that is declared and used inside the body of methods, constructors, or blocks
- It must be assigned a value at the time of creating.
- No access modifiers/static can be used with local variables

```
public static void main(String[] args) {
    int a = 5;  // local variable
    int b; // local variable must be initialized
}
```

Variables

Instance Variable

- A variable that is declared inside the class but outside the body of methods, constructors, or blocks
- These variables are created and destroyed based on object.
- Access modifiers can be used with local variables

```
class AA {
int aa; // instance variable
public static void main(String[] args) {
    int a = 5;  // local variable
}
```

Variables

Static/Class Variable

- Static variables are always declared inside the class but outside of any methods, constructors, or blocks.
- It will have common memory, where all the objects shares the same.

```
class AA {
int aa; // instance variable
static int bb; // static variable
public static void main(String[] args) {
    int a = 5;  // local variable
}
```