```
import { GooglePlus } from "@ionic-native/google-plus/ngx";
import { Platform } from "@ionic/angular";

import { LoadingController } from "@ionic/angular";
import { Storage } from "@ionic/storage";

@Injectable({
  providedIn: "root"
})
export class AuthService {
  user$: Observable<any>;

  constructor(
    private afAuth: AngularFireAuth,
    private db: DbService,
    private router: Router,
    private gplus: GooglePlus,
    private platform: Platform,
    private loadingController: LoadingController,
    private storage: Storage
  ) {
    this.user$ = this.afAuth.authState.pipe(
      switchMap(user => (user ? db.doc$(`users/${user.uid}`) : of(null)))
    );

    this.handleRedirect();
  }

  uid() {
    return this.user$
      .pipe(
        take(1),
        map(u => u && u.uid)
      )
      .toPromise();
  }

  async anonymousLogin() {
    const credential = await this.afAuth.auth.signInAnonymously();
    return await this.updateUserData(credential.user);
  }
```

Java

What is Java ?

✣ Java is simple high-performance, object-oriented, platform independent language, for develop software that can we use for gaming, desktop, database, and another application.

Java is a high-level, object-oriented programming language.

❃ Java was developed at Sun-microsystem in the year 1995 by James Gosling who is known as father of Java. Sun-microsystem which is now the subsidiary of Oracle Corporation.

✻ Java is one of the world's most important and widely used programming language.

Java's Lineage: Java is related to C++, which is direct descendant of C. Much of the character of Java is inherited from these two languages (C or C++). Java derives its syntax from 'C', and many of Java's object oriented programming(OOP) features were influenced by C++.

OOP: OOP is a programming language methodology that helps to solve complex programs through the use of inheritance, encapsulation and polymorphism.

The Creation of Java: Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991. It looks 18 moths to develop the first working version. This language was initially called "Oak," but was renamed "Java" in 1995, because "Oak" name was used by "Oak Technology".

- * The similarities between Java and C++, it is tempting to think of Java as simply the "Internet version of C++". While it is true that Java was influenced by C++, it is not an enhanced version of C++. Java is neither upwardly nor downwardly compatible with C++. One more another point: Java was not designed to replace C++. Java was designed to solve a certain set of problems while C++ was designed to solve a different set of problems.

Java Applets: An applet is a special kind of Java program that is designed to be transmitted over the internet and automatically executed inside a Java-compatible web browser. If the user clicks a link that contains an applet, the applet will download and run in the browser.

Use: They were typically used to display data provided by the server, handle user input, or provide simple functions, such as koan calculator, that execute locally, rather than on the server.

→ In general, there are two very broad categories of objects that are transmitted between the server and the client: passive and dynamic, active programs. For example, when we read our e-mail we are viewing passive data. Even when we download a program, the program's code is still only passive data until we execute it. By contrast the applet is a dynamic, self exciting program. Such a program is an active agent on the client computer, yet is initiated by the server.

The Bytecode: Bytecode is a highly optimized set of instructions designed to be executed by the Java Virtual Machine(JVM), which is part of the Java Runtime Environment (JRE). In essence, the original JVM was designed as an interpreter bytecode.

☆In Simple Language: Java source code is compiled into bytecode when we use Java Compiler. The byte code is gets saved on the drive with the extension of .class. After the converting source code into bytecode, bytecode is faded into JVM.

What's the reason behind developing Java:

(1): Java is a high-level language.

(2): Java is a secure language (Java programs is run on JVM).

(3): Java is a robust (powerful-language).

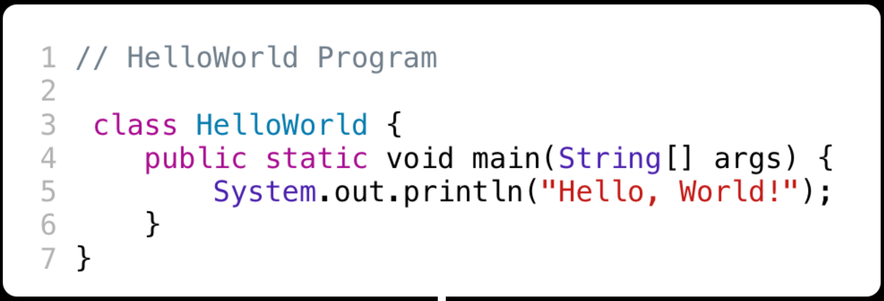
(4): Java is a platform independent language.

PLATFORM: Any hardware or software environment in which a program run is known as platform.

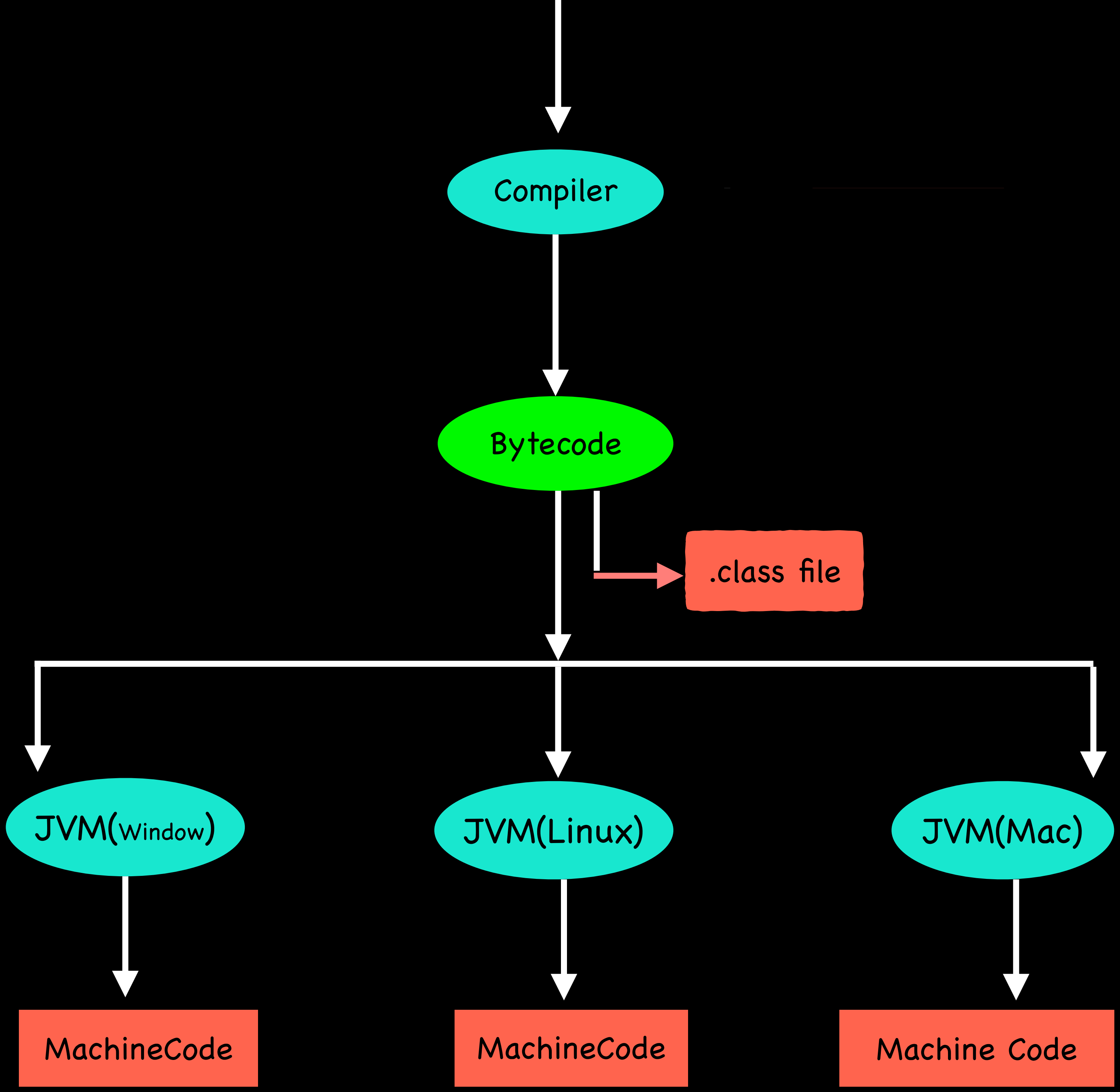
Motive of Java:

WORA

WRITE ONCE RUN ANYWHERE



```
// HelloWorld Program

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



Uses of Java:

1. Desktop Applications(Media Player, Antivirus, etc..).
2. Web Applications (IRCTC, Apache Software Foundation, Wikipedia, IBM Developer Works, Stack-overflow, etc...).
3. Enterprise Applications (Banking Applications).
4. Mobile
5. Embedded System
6. Robotics
7. Games etc..

Types of Java Application:

1.Standalone Application: Standalone application is also known as desktop applications or window-based applications. A Standalone application is an application that runs locally on the devices and doesn't' require anything else to be functional. All thing is built in the app, so it doesn't need any an internet connection nor any other servers installed, They did not need any browser to run. Like: Media Player(VLC), Antivirus(QuickHeal, Kaspersky, Guardian). AWT(Abstract Window Toolkit) and Swing are used to create Standalone application.

2. Web Application: An application that runs on the

server-side and creates a dynamic page is called a Web Application. Servlet, JSP, Struts, Spring, Hibernate, JSF... etc are used to creating web applications in Java.

OR

A web application is application software that runs on a web-server, unlike computer-based software program that are run locally on the operating system(OS) of the device. Web application are accessed by the client/ user with the help of internet. These applications are programmed using client-served modelled structure- the client/user is provided services through an off-side server that is hosted by third-party.

Example of Commonly used Web Application:

1. Online Retail Stores
2. Internet Banking
3. Web-Mails

3. Enterprise Application: Enterprise application, also known as enterprise software, is computer software which is designed for larger corporate environment (Business, Government, Schools, Club, Charities) rather than individual users is called an Enterprise Application or Software.

✢ EJB is used for creating enterprise application.

4. Mobile Applications: An application which is developed for mobile device(Android, iOS) is called Mobile Application.

✧Java ME are used for developing mobile applications.

Java Platforms:

1. Java SE (Standard Edition,) (Include Basics to String)
2. Java EE (Enterprise Edition)
3. JavaME (Micro/Mobile Edition)
4. JavaFX

Feature of Java:

1. Simple: Java is easy to learn, and its syntax is quite Simple, clean and easy to understand.

Why Simple?

Ans: Java syntax and OOP is based on C++, It means if you already learned/understand the syntax and OOP of C++, learning to Java will be much easier. Java removed many complicated features like Pointers and Operator overloading, this makes Java too simple, and there is no need to remove unreferenced objects because there is an Auto Garbage Collection in Java, That's why it's Simple.

2.Object-Oriented: Java is an object-oriented programming language we learned this earlier. Everything in Java is an object. Object-Oriented means we organize our software as a combination of different types of objects that incorporate both data and behaviour.

Basic Concept of OOPs are:

A. Object
B. Class
C. Inheritance
D. Polymorphism
E. Abstraction
F. Encapsulation

3. Robust: To understand how Java is robust, consider two of the main reason for program failure: memory management mistakes and mishandled exceptional conditions (run-time error). Memory management can be a difficult, tedious task in traditional programming environments. For example, in C/C++, We programmers will often manually allocate and free dynamic memory. This sometimes leads to problems, because we either forget to free memory that has been previously allocated or worse, try to free some memory that another part of their code is still using. Java virtually eliminates these problems by managing memory allocation and deallocation for us. (In fact, deallocation is completely automatic, because Java provides garbage collection for unused objects.

Exceptional conditions is traditional environments often arise in situations such as division by zero or "File not found". Java helps in this area by providing object-oriented execution handling.

4. Architecture-Neutral: When Java is created, one of main problems faced by programmers that is no guarantee existed that if we write a program today, it would run tomorrow- even on the same machine. Because OS upgrades, processor upgrades, and changes in core systems resources can all combine to make a program malfunction. The Java designers made several hard decision in the Java language and Java Virtual Machine(JVM) in an attempt to alter this situation. Their goal was WORA(Write Once Run Anywhere, anytime forever).

5. Platform-Independent: WORA.
Example: Write Your code on Mac and you can run your code in any platform(Windows OS, Linux OS).

6. Interpreted & High-Performance: Java is faster than other traditional interpreted programming language Java bytecode is "close" to native code. It is still a little bit slower than a compiled language(C++), because Java is interpreted language.

7 Security: As we likely aware, every time we download a “normal” program, we are taking a risk, because the code we are downloading might contain a virus, Trojan horse, or other harmful code. At the core of the problem is the fact that malicious code can cause it damage because it has gained unauthorised access to system resources. For example, a virus program might gather private information, such as credit card numbers, bank account balances, and passwords, by searching the content of our computer’s file systems. In order for Java to enable programs to be safely downloaded and executed on the client computer, it was necessary to prevent them from launching such an attack. Java achieved this protection by enabling us to continue an application to the Java execution environment and prevent it from accessing other parts of the computer. The ability to download a program with a degree of confidence that no harm will be done may have seen the single most innovative aspect of Java.

OR

Java is best known for its security. With Java, We can develop virus-free systems. Java is secured because No exploit pointers and Java programs run inside a Java Virtual Machine (JVM)

8 Portability: Java is portable because it facilitates to carry the Java bytecode to any platform. It doesn't require any implementation. In other words, A mechanism that allows the same application to be downloaded and executed by a wide variety of CPUs, operating systems, and browser is required. The same application code must work on all computers.

9 Dynamic: Java is a dynamic language. It supports the dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++.

→ Java supports dynamic compilation and automatic memory management (garbage collection).

10.Multithread: A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it doesn't occupy memory for each thread. Threads are important for multi-media, Web applications, etc.

JDK(Java Development Kit): The JDK is a software development tool which is used to develop Java application/program

JDK Platforms released by Oracle Corporation:

- Standard Edition Java Platform
- Enterprise Edition Java Platform
- Micro Edition Java Platform

JRE(Java Runtime Environment): JRE provides the runtime environment for the JVM. In other word we can say that the JRE is the implementation of JVM .

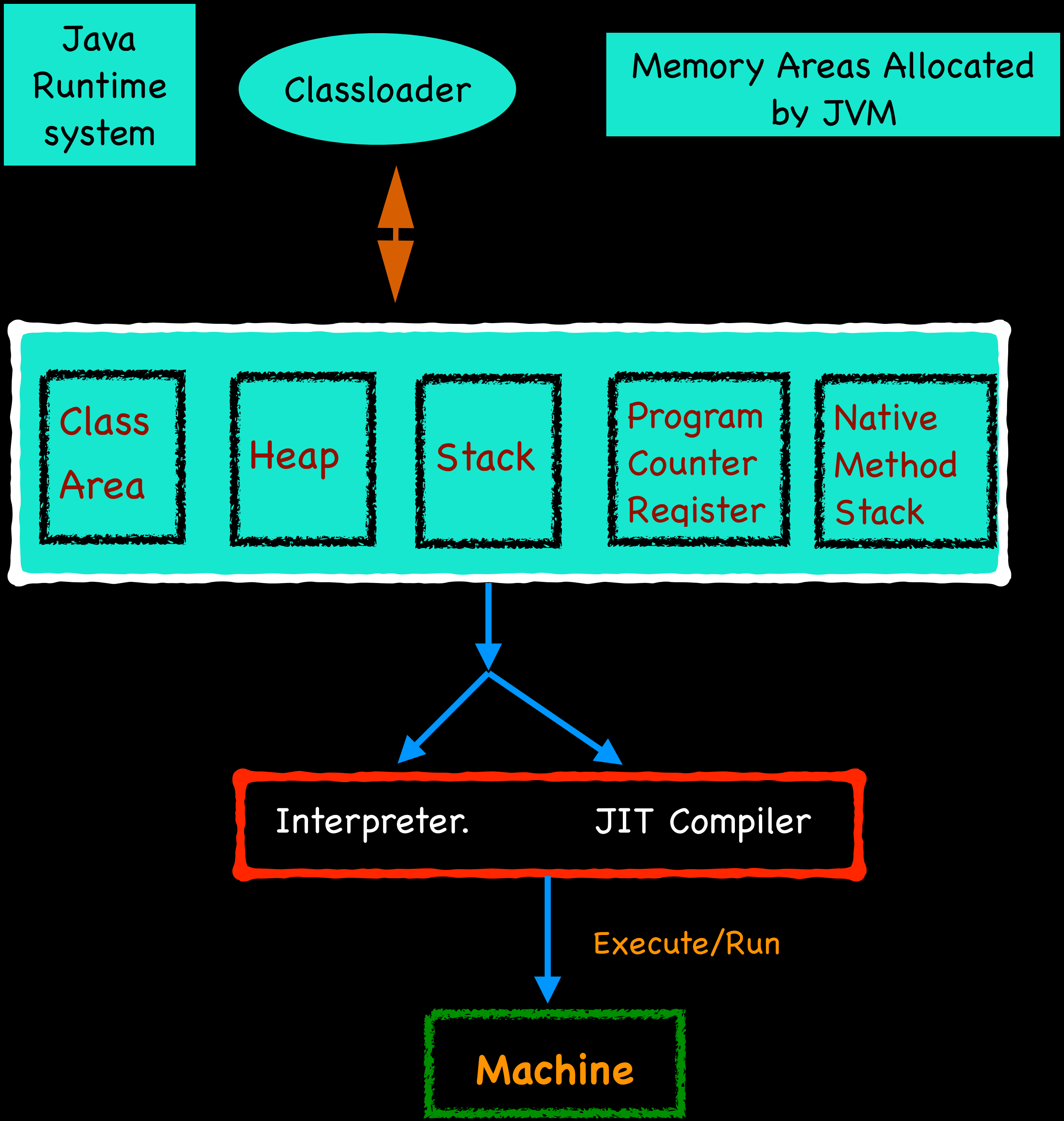
JRE contains a set of libraries + other files that Java uses at runtime.

JVM(Java Virtual Machine): JVM is an abstract machine, it is called virtual because, it doesn't exist physically. It is a specification that provides a runtime environment in which Java bytecode can be executed. It can also run those programs which are written in other languages and compiled to Java bytecode. JVM is responsible to run java application in JRE.

JDK = Development tool + JRE
JRE = Class Libraries +JVM

JVM performs the following operations:

- Loads source code.
- Checks and verifies the source code.
- Provide runtime environment to execute the code.



Java Runtime system

Classloader

Memory Areas Allocated by JVM

Class Area

Heap

Stack

Program Counter Register

Native Method Stack

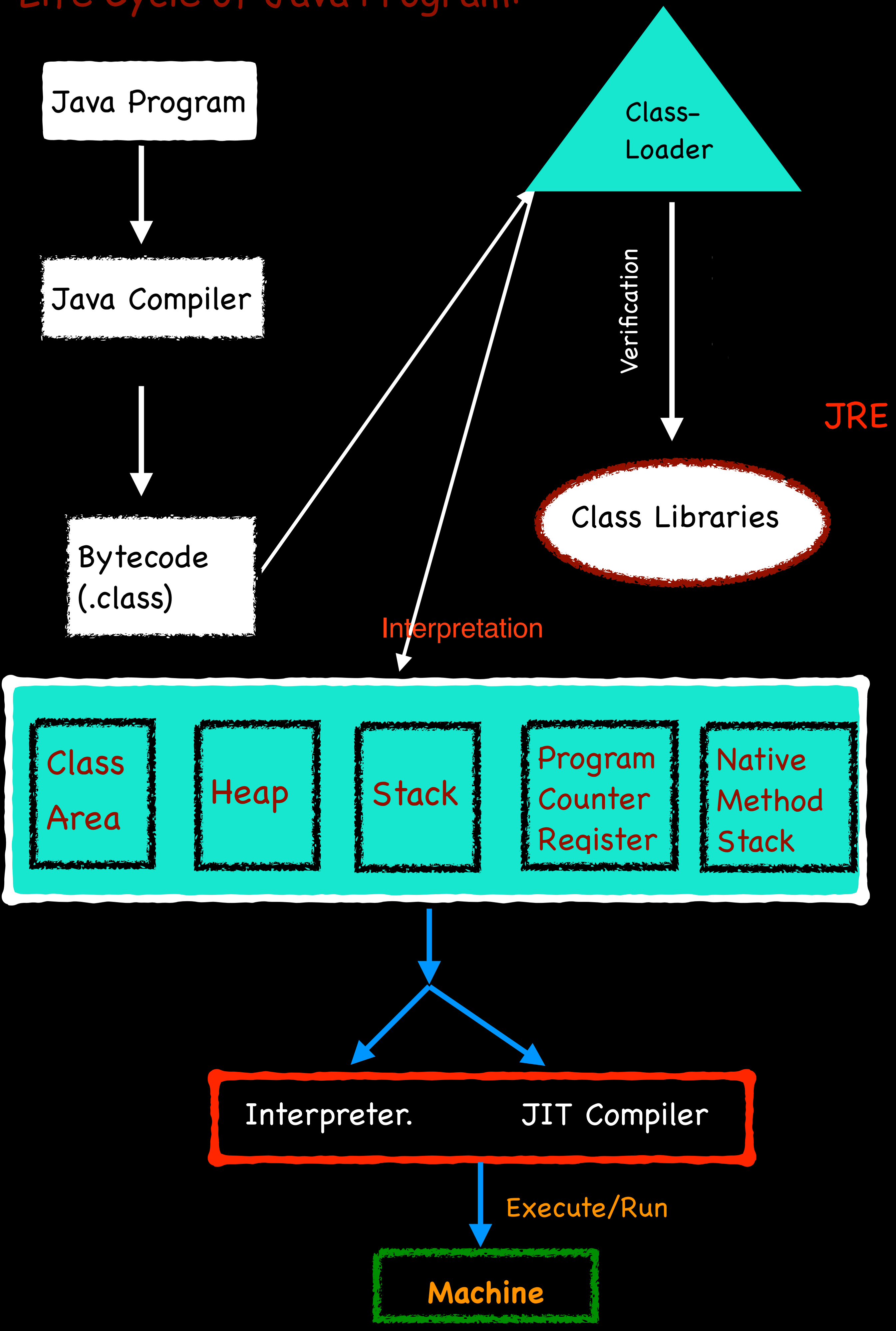
Interpreter.

JIT Compiler

Execute/Run

Machine

Life Cycle of Java Program:



Java Program

Java Compiler

Bytecode (.class)

Class-Loader

Verification

Class Libraries

JRE

Interpretation

Class Area

Heap

Stack

Program Counter Register

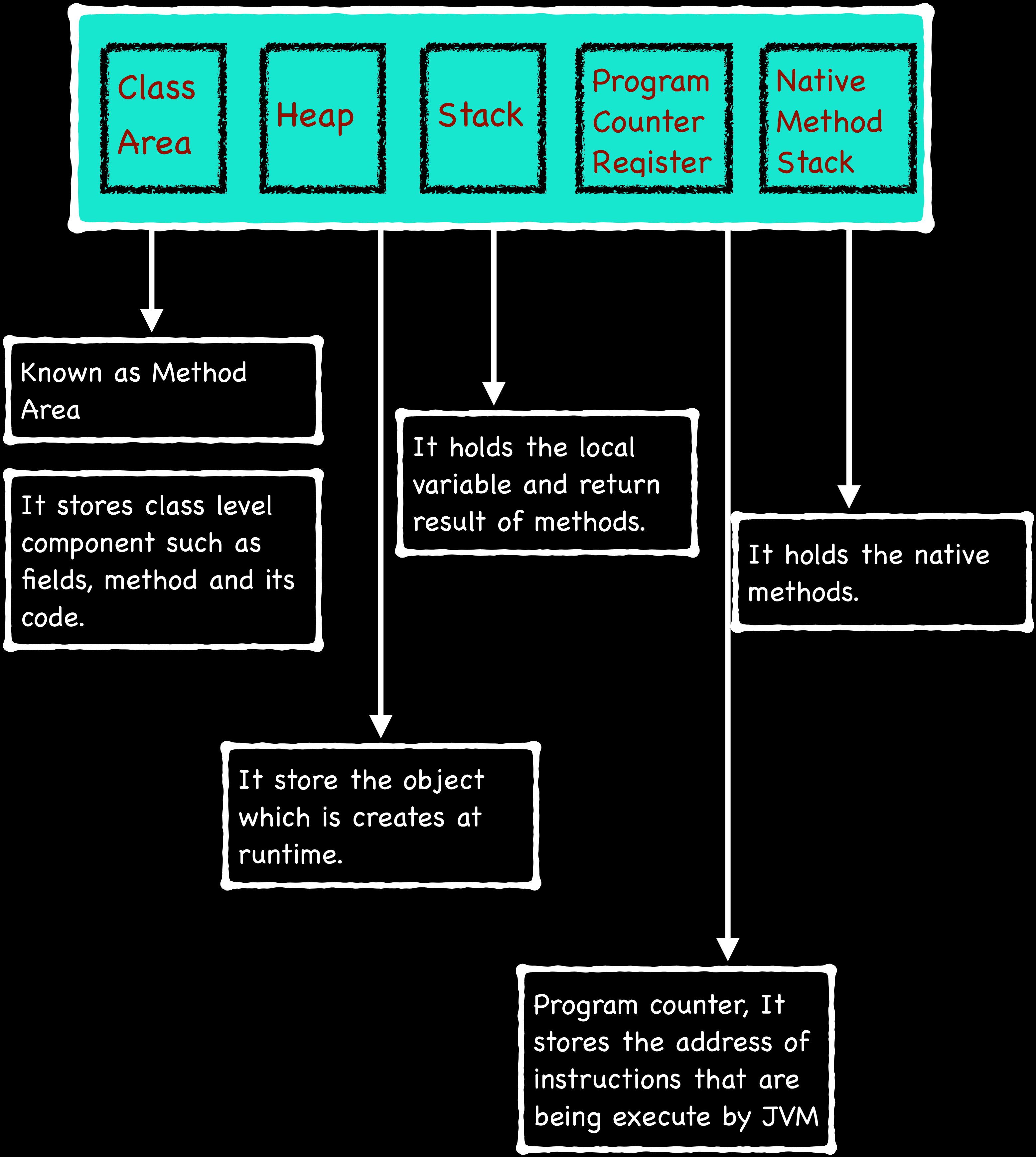
Native Method Stack

Interpreter.

JIT Compiler

Execute/Run

Machine



Class Area

Heap

Stack

Program Counter Register

Native Method Stack

Known as Method Area

It stores class level component such as fields, method and its code.

It store the object which is creates at runtime.

It holds the local variable and return result of methods.

Program counter, It stores the address of instructions that are being execute by JVM

It holds the native methods.

Basic Syntax: A Java program is a collection of objects, and these objects communicate through method calls to each other to work together.

Basic Terminologies in Java:

1. Object: The object is an instance of a class, have Behavior and state.
Example: A car is an object whose **states** are: brand, colour, number-plate.
Behavior: Running on the Road

2. Class: A class is a blueprint of objects and status.
Example: Blueprint of the house is class.

3. Method: The behaviour of an object is the method.
Examples: The fuel indicator indicates the amount of fuel left in the car.

4. Instance Variables: Every object has its own unique set of instance variables. The state of an object is generally created by the values that are assigned to these instance variables.

The Basic Syntax:

Comments: Comments are non-executable statement, we write a remark into a program's source code with the help of comments. Non-executable means the content of a comment are ignored by the compiler. Comments describes or explains the operation of the program to anyone who is reading the source code. In real applications, comments generally explain how some part of the program works or what a specific feature does.

There are three types of comments in Java:

1. **Single Line Comment (//):** Single line comments starts with // and ends at the end of the line. Single-line comments are used for brief, line-by-line description.
2. **Multiline Comments(/*….*/):** Multiline comments starts with /* and ends with */, anything between these two comments symbols is ignored by the compiler. Multiline comments are used for longer remarks.
3. **Documentation Comments(/**….*/:** Documentation comments starts with /** and ends with */. This comment is used to produce an HTML file that

Identifiers: Identifiers are used to name things, such as classes, variables, and methods. They can be a class name, variable name, method name, package name, constant and more.

OR

Identifiers are the names that are used to identify the programming elements include by us/programmer.

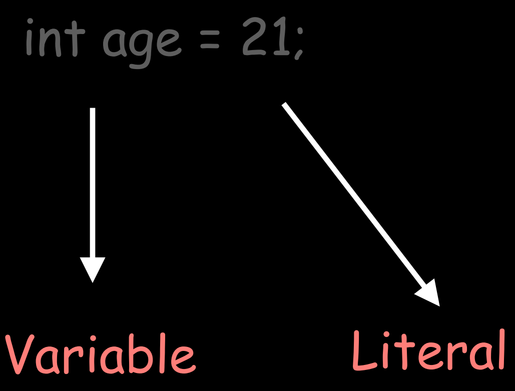
Rules of Using Identifiers:

1. An identifier cannot start with a digit.
2. A valid identifiers starts with [A-Z] or [a-z] or numbers [0-9], and underscore(_)
3. There should not be a any space in an identifier.
4. A keyword cannot be used as a variable/ identifier names.

Valid Identifiers: TeNet, starship, uss4, $fifty, dhruv_lenka

Invalid Identifiers: 2uss, Yes/No, dhruv-lenka

Literals: Any constant value which can be assigned to the variable is called as literal/ constant.



int age = 21;

Variable

Literal

Types of Literals:

1. **Integer Literals:** int age = 21, int octAge = 021,
int hexAge = 0x2a.

2. **Character Literals:** ‘d’, ‘@' etc..

3. **Boolean Literals:** isEven = true, isOdd = false.

4. **String:** “DhruvLenka”,
“dhruvlenka@protonmail.com".

5. **Floating-Point Literals:** float length = 5.5′,
double interest = 9999.4456.

6. **BackSlash Literals:** \n, \t, \b, \v, \r, \’, \”.

7. **Null(Unavailable) Literals:** string age = null,
string stuName = null.

Separators: In Java, there are a few characters that are used as separators. The most commonly used separators in Java is the semicolon.

semicolon is often used to terminate statements.

( )	**Parantheses**	Used to contain lists of parameters in method definition and invocation also used for defining precedence in expressions containing expressions in control statements and surrounding cast types.
{ }	Braces	Used to contain the values of automatically initialized arrays, block of code, for. classes, methods, local scopes.
[ ]	Brackets	Used to declare array types. Also used when dereferencing array values.
;	Semicolon	Terminates statements
,	Comma	
.	Period	Used to separate package names from subpackages and classes. Also used to seperate a variable or method from a reference variable.
::	Colons	Used to create a method or constructor reference
...	Ellipsis	Indicates a variable-parity parameter
@	Ampersand	Begins an annotation

Keywords: There are 61 keywords are currently defined in Java. These keywords, combined with the syntax of the operators and separators, form the foundation.

abstract	**Assert**	boolean	break
catch	char	class	const
do	double	else	enum
final	finally	float	for
implements	import	instanceof	int
module	native	new	open
private	protected	provides	public
short	static	strictfp	super
this	throw	throws	to
try	Uses	void	volatile
byte	opens	case	package
continue	requires	default	return
exports	switch	extends	synchronised
goto	transient	if	transitive
interface	while	long	with

Variables: A variables is an identifier which is used to store some value.

OR

A variables is a container which holds the value while the Java program is executed.

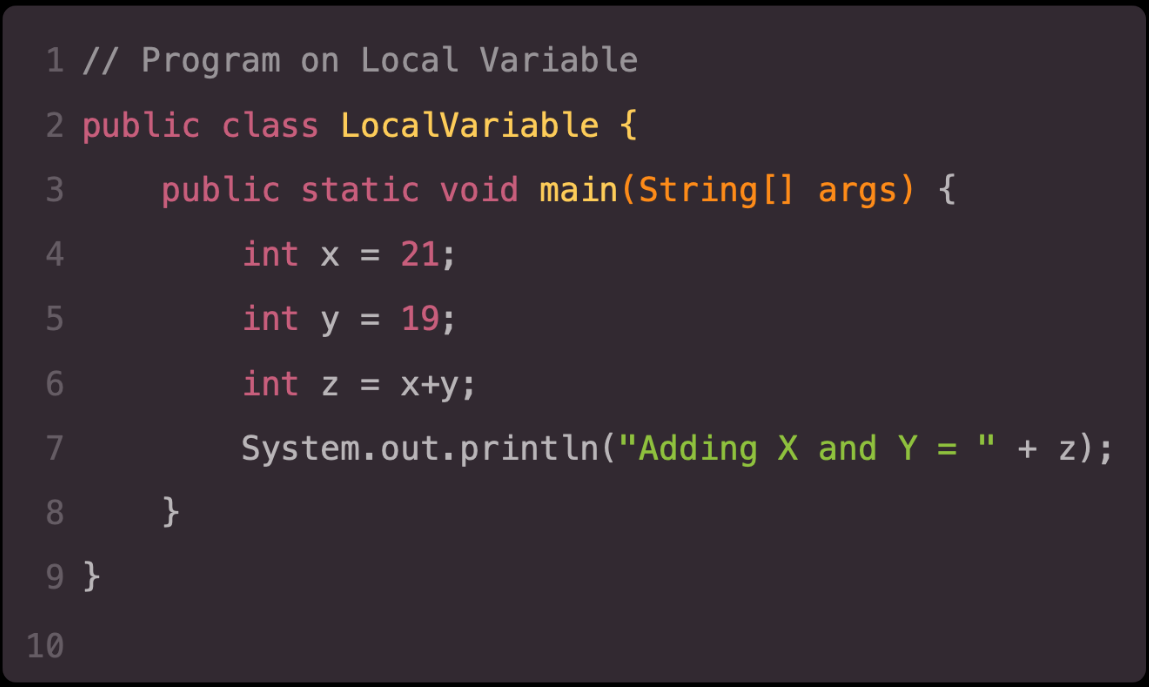
✧ A variable is assigned with a data type.

✧ Variable is name of memory location.

Types of Variables in Java:

1. **Local variable:** A variable which is declared in inside the body of the method is called Local variable.
2. **Instance variable:** A variable which is declared in inside the class but outside the body of the method, is called an instance variable. It is not declared as static.
3. **Static variable:** A variable which is declared as static is called a static variable. It cannot be local variable. We can create a single copy of the static variable and we can share it among all the instances of the class. Memory allocation for static variables happens only once when. The class is loaded in the memory.

Local Variable PROGRAM:



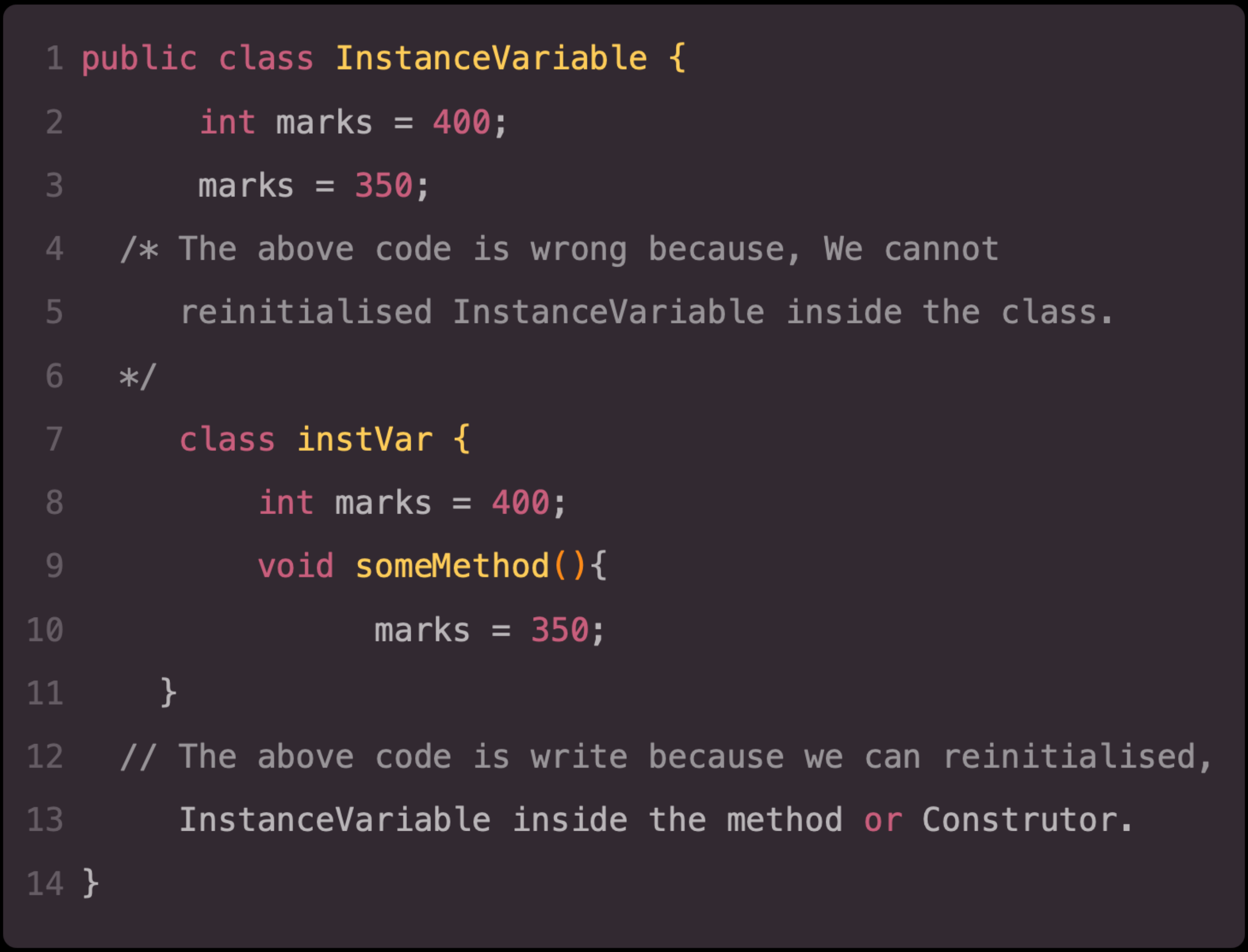
```java
// Program on Local Variable
public class LocalVariable {
    public static void main(String[] args) {
        int x = 21;
        int y = 19;
        int z = x+y;
        System.out.println("Adding X and Y = " + z);
    }
}

```

Instance Variable PROPERTIES:

1. Instance Variable always gets default value.
2. Instance Variable cannot be reinitialised directly within class but they can reinitialised inside method or

Static Variable PROPERTIES:

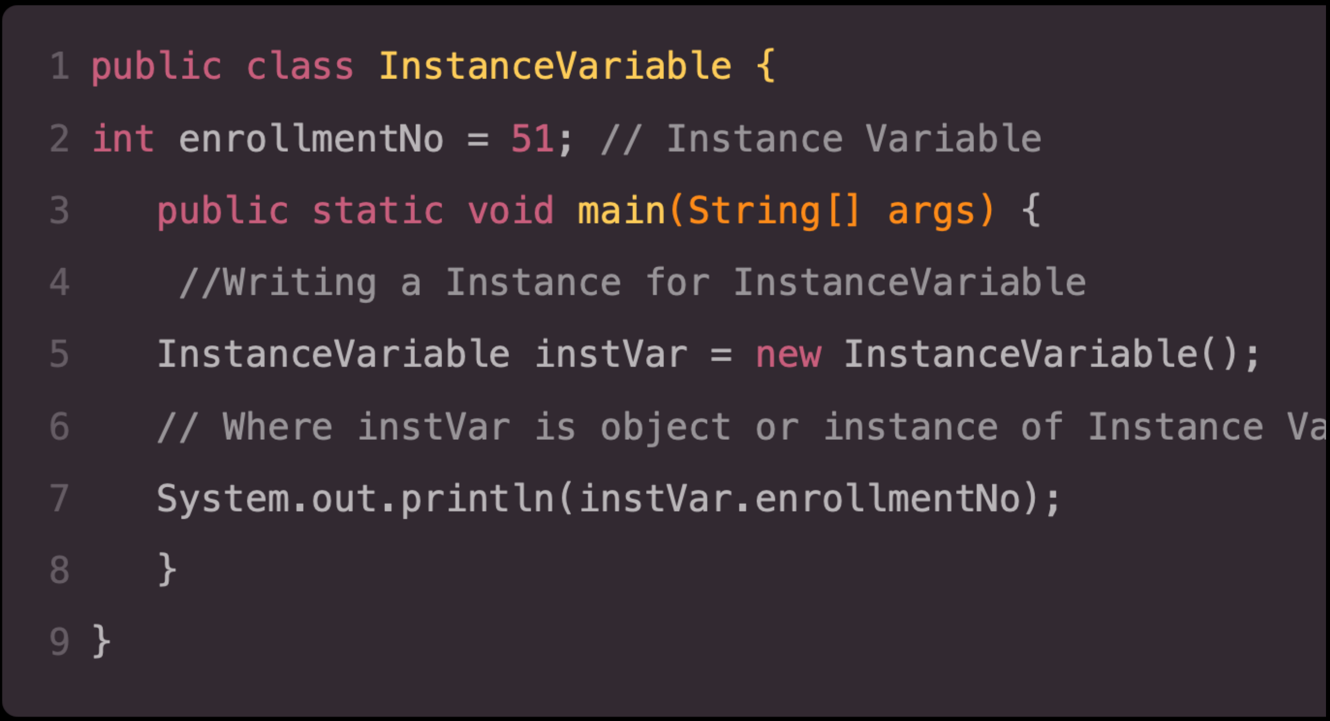
- Static Variable is Independent, Independent means anyone can us Static variable inside the class and static variable don't need any help of any object.



```
public class InstanceVariable {
      int marks = 400;
      marks = 350;
  /* The above code is wrong because, We cannot
     reinitialised InstanceVariable inside the class.
  */
     class instVar {
         int marks = 400;
         void someMethod(){
               marks = 350;
    }
  // The above code is write because we can reinitialised,
     InstanceVariable inside the method or Construtor.
}
```

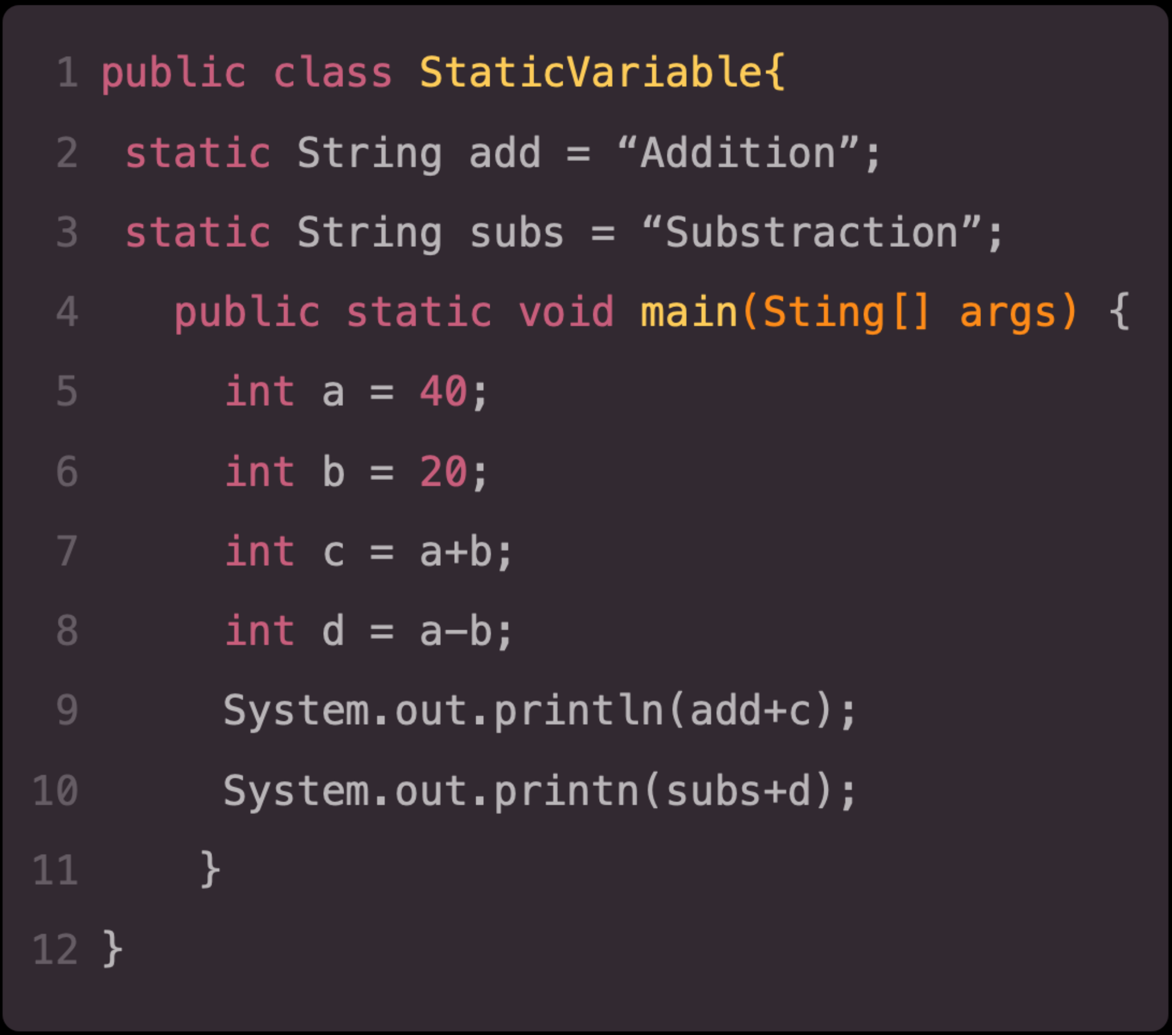
Instance Variable PROPERTIES EXAMPLE

Instance Variable Program:



```
public class InstanceVariable {
int enrollmentNo = 51; // Instance Variable
   public static void main(String[] args) {
    //Writing a Instance for InstanceVariable
   InstanceVariable instVar = new InstanceVariable();
   // Where instVar is object or instance of Instance Va
   System.out.println(instVar.enrollmentNo);
   }
}
```

Static Variable Program:



```
public class StaticVariable{
 static String add = “Addition”;
 static String subs = “Substraction”;
   public static void main(Sting[] args) {
     int a = 40;
     int b = 20;
     int c = a+b;
     int d = a-b;
     System.out.println(add+c);
     System.out.printn(subs+d);
    }
}
```

Data Type: Data types are declaration of variables.

OR

Data types are used to tell the variables which type of data it can store.

There are two types of data type present in Java:

1. Primitive Data Type: The primitive data type include int, char, float, double, long, short, boolean and byte.
2. Non-Primitive Data Type: The non-primitive data type include Classes, Interfaces, and Arrays.

Defining Primitive Data Type:

1. Integer: This group includes bytes, short, int, and long.

   Ex: int 20, int -20, short =, long a = 1000000L.

2. Floating-Point Numbers: This group includes float and double, which represent numbers with fractional precision.

Ex: float = 21.2, double = 210000.2

3. Characters: This data type is used to store characters.

Ex: char letter = ‘D’, char = ‘L’

4. Boolean: The boolean data type is used to store only two possible values (True and False).

✻ Java does not support unsigned integers, Many other programming languages like C,C++ supports both signed and unsigned integer. However, Java's designers felt that unsigned integers were unnecessary. Specifically, they felt that the concept of unsigned was used mostly to specify the behaviour of the high-order bit, which defines the sign of an integer value. Java manages the meaning of the high-order bit differently, by adding a special "unsigned right shift" operator. Thus, the need for an unsigned integer type was eliminated.

Name	Width	Range
long	64	-9,223,372,036,854,775, 808 To 9.223,372,036,854,775,807
int	32	-2,147,483,648 To 2,147,483,647
short	16	-32768 To 32767
byte	**8**	**-128 To 127**

✧ Byte variables are declared by use of the byte keyword like long, int and short.

☞ 'long' is useful for those occasions where an 'int' type is not enough to hold the desired value. The Range of a 'long' is quite large. This makes it useful when big, whole numbers are needed.

Floating-Point Data Types: Floating-point numbers, also known as real numbers, are used when evaluating expressions that require fraction precision. Helps to calculate such as square root, or transcendentals such as sine and cosine, result in a value whose precision requires a floating-type. There are two king of floating-point types float and double which represent single and double-precision numbers.

Double	64	4.9e-324 to 1.8e+308
float	32	1.4e-045 to 3.4e+038

Double: double precision, as denoted by the double keyword. Double precision is actually faster than single precision on some modern processor that have been optimized for high-speed mathematical calculations. All transcendental math functions, such as sin(), cos(), and sqrt(), return double values. When you need to maintain accuracy over many iterative calculations, or are manipulating larges-valued numbers.

Characters: In Java, the data type used to store characters is char. A key point to understand is that Java uses Unicode to represent characters. At the time of Java’s creation, Unicode required 16 bits. Thus, in Java char is a 16 bits type. The range of a char is 0 to 65,536. There are no negative chars.

Escape Sequence: /n: New Line

/f: Form Feed

/t: Tab

/b: Backspace

/r: Carriage return

\\: Backslash

\”: Double Quote

\’: Single Quote

\uxxx: HexaD Unicode Char

\ddd: Octal character(ddd)

Unicode: Unicode provides a unique number for every character;

No matter what the platform,
No matter what the program,
No matter what the language.

Why Java uses Unicode ?

Ans: To enable a computer systems for storing text and numbers which is understandable by humans, there should be a code that transforms characters into numbers. Unicode is a standard of defining the relevant code by using character encoding. Character encoding is the process for assigning a number for every character.

Characters Before Unicode:

Fundamentally, computers just deal with numbers. They store letters and other characters by assigning a number for each one. Before Unicode was invented, there were hundreds of different systems, called character encodings, for assigning these numbers.

Early character encodings also conflicted with one another. That is, two encodings could use the same number for two different characters, or use different numbers for the same character. Any given computer (especially servers) would need to support many different encodings. However, when data is passed through different computers or between different encodings, that data runs the risk of corruption.

Unicode Characters:

The Unicode Standard provides a unique number for every character, no matter what platform, device, application or language.

It has been adopted by all modern software providers and now allows data to be transported through many different platforms, devices and applications without corruption. Support of Unicode forms the foundation for the representation of languages and symbols in all major operating systems, search engines, browsers, laptops, and smart phones—plus the Internet and World Wide Web (URLs, HTML, XML, CSS, JSON, etc.). Supporting Unicode is the best way to implement ISO/IEC 10646.