

Introduction to Java as Object Oriented Programming Language

Content

- Principles of OOP:
 - Object,
 - Class,
 - Encapsulation,
 - Abstraction,
 - Inheritance,
 - Polymorphism
- Basic Constructs:
 - Constants,
 - variables and data types,
 - Wrapper classes,
 - Operators and Expressions
- Input & Output in Java:
 - command line arguments,
 - BufferedReader class and
 - Scanner class

What is an Object

- The Object is the real-time entity having some **state and behavior**.
- In Java, Object is an **instance of the class** having the instance variables like the **state** of the object and the **methods** as the behavior of the object.

State: It is represented by **attributes** of an object. It also reflects the properties of an object.

Behavior: It is represented by methods of an object. It also reflects the **response of an object with other objects**.

Identity: It gives a **unique name** to an object and enables one object to interact with other objects.

Example of an object: dog

Identity

Name of dog

State/Attributes

Breed

Age

Color

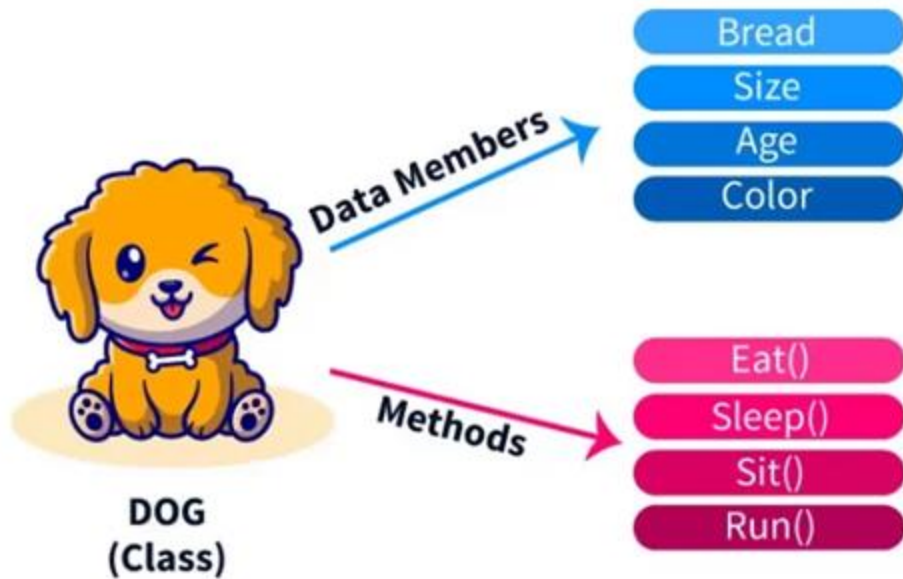
Behaviors

Bark

Sleep

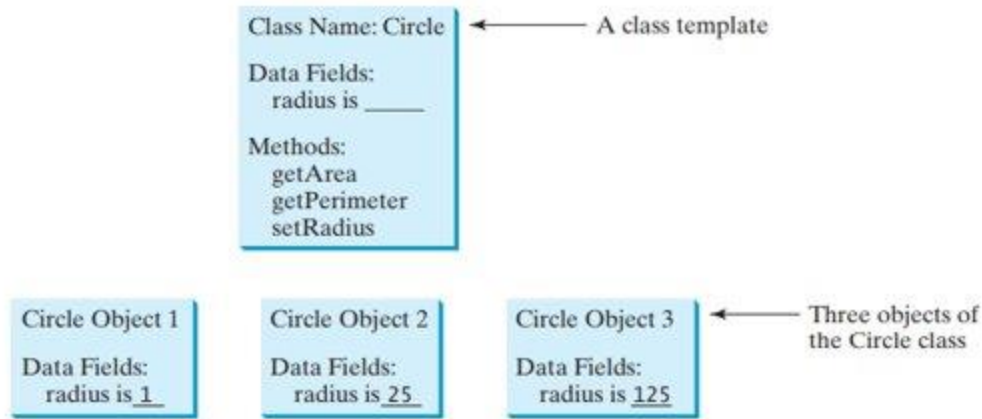
Eat

INTRODUCTION



What is class?

- A class is a group of objects which have **common properties**.
- It is a template or blueprint from which objects are created.
- In short, a class is the **specification or template of an object**.



Class

Blueprint



Object



Van

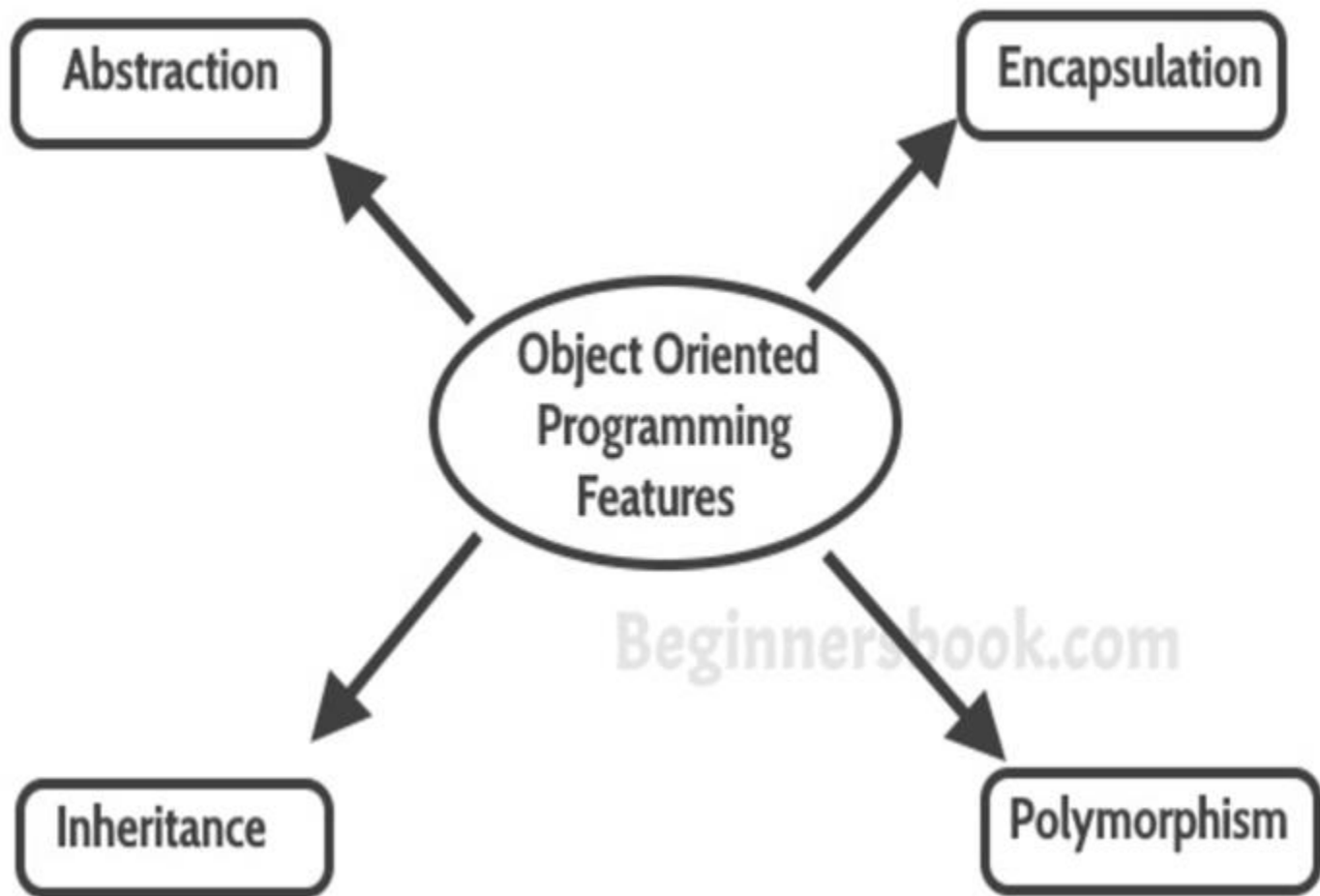


Audi



Sports car

Class	Object
1) Class is a collection of similar objects	1) Object is an instance of a class
2) Class is conceptual (is a template)	2) Object is real
3) No memory is allocated for a class.	3) Each object has its own memory
4) Class can exist without any objects	4) Objects can't exist without a class
5) Class does not have any values associated with the fields	5) Every object has its own values associated with the fields



Abstraction

- One of the most fundamental concept of OOPs is Abstraction.
- Abstraction is a process where you **show only “relevant” data** and **“hide” unnecessary details of an object from the user.**
- For example, when you login to your Amazon account online, you enter your user_id and password and press login, what happens when you press login, how the input data sent to amazon server, how it gets verified is all abstracted away from the you.
- *Abstraction* means hiding lower-level details and **exposing only the essential and relevant details to the users.**

Abstraction helps to reduce complexity.





Advantages of Java Abstraction



Inheritance

In OOP, computer programs are designed in such a way where everything is an object that interact with one


another. Inheritance is one such concept where the **properties of one class can be inherited by the other.** It

helps to reuse the code and establish a relationship between different classes.

1. Parent class (Super or Base class)

2. Child class (Subclass or Derived class)

A class which inherits the properties is known as Child Class whereas a class whose properties are inherited is known as Parent class.



Son i am
Base Class

Dad i am
Derive Class

Polymorphism

- Polymorphism means taking many forms, where 'poly' means many and 'morph' means forms.
- It is the ability of a variable, function or object to take on multiple forms.
- In other words, polymorphism allows you define one interface or method and have multiple implementations.

In Shopping malls behave like

CUSTOMER

In Bus behave like

PASSENGER

In School behave like

STUDENT

At Home behave like

SON



Bowler Class : (Parent Class)
-bowlingMethod()

FastPacer :(Child Class)
-Bowling Method()



MediumPacer :(Child Class)
-Bowling Method()



Spinner :(Child Class)
-Bowling Method()



The point of above discussion is simply that a **same name** tends to multiple forms. All the three classes above inherited the bowlingMethod() but their **implementation is totally different from one another.**

Encapsulation

- Encapsulation is a mechanism where you **bind your data and code together as a single unit.**
- It also means to hide your data in order to make it safe from any **modification.**
- Eg: medical capsule, where the drug is always safe inside the capsule. Similarly, through encapsulation the methods and variables of a class are well hidden and safe.

class

{

data members

+

methods (behavior)

}

ENCAPSULATION

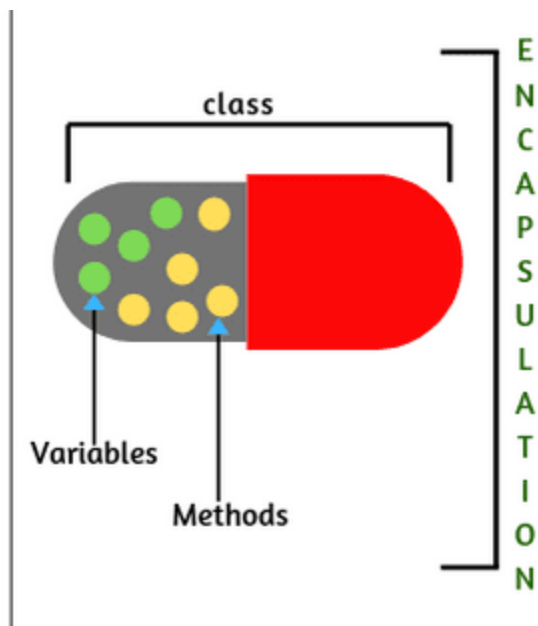


Fig: Encapsulation

Abstraction	Encapsulation
1. Abstraction solves the problem in the design level.	1. Encapsulation solves the problem in the implementation level.
2. Abstraction is used for hiding the unwanted data and giving relevant data.	2. Encapsulation means hiding the code and data into a single unit to protect the data from outside world.
3. Abstraction lets you focus on what the object does instead of how it does it	3. Encapsulation means hiding the internal details or mechanics of how an object does something.
<p>4. Abstraction- Outer layout, used in terms of design.</p> <p>For Example:-</p> <p>Outer Look of a Mobile Phone, like it has a display screen and keypad buttons to dial a number.</p>	<p>4. Encapsulation- Inner layout, used in terms of implementation.</p> <p>For Example:- Inner Implementation detail of a Mobile Phone, how keypad button and Display Screen are connect with each other using circuits.</p>

Steps of Executing Java Program

1. For executing any java program, you need to Install the JDK.
If you don't have installed it, download and install it.
<https://www.oracle.com/java/technologies/javase-downloads.html>
2. Set path of the jdk/bin directory.
3. Create the java program using notepad
4. Compile and run the java program using javac and java command.

Set path

- The path is required to be set for using tools such as javac, java, etc.
- If you are saving the Java source file inside the JDK/bin directory, the path is not required to be set because all the tools will be available in the current directory.
- However, if you have your Java file outside the JDK/bin folder, it is necessary to set the path of JDK.
- We can set temporary as well as permanent path.

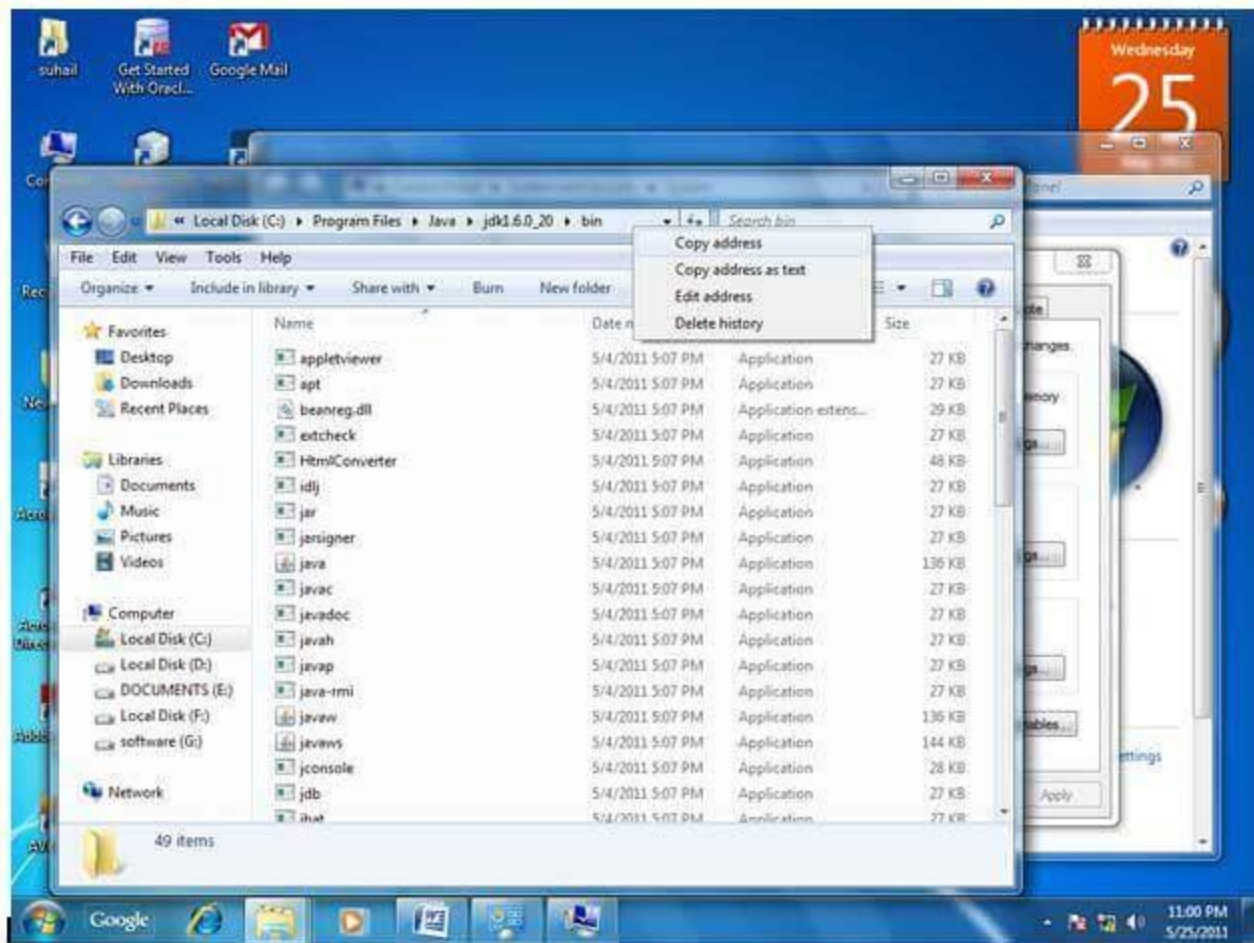
How to set the Temporary Path

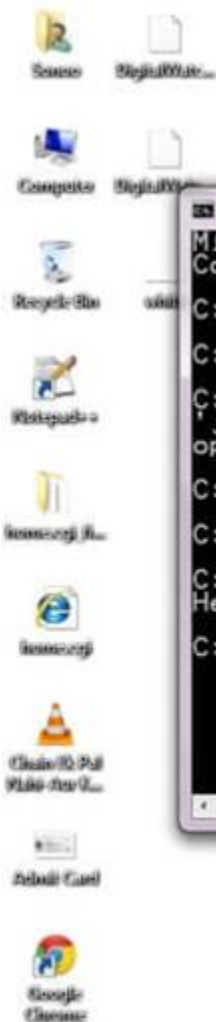
To set the temporary path of JDK, you need to follow the following steps:

1. Open the command prompt
2. copy the path of the JDK/bin directory
3. write in command prompt: `set path=copied_path`

For Example:

```
set path=C:\Program Files\Java\jdk-14\bin
```

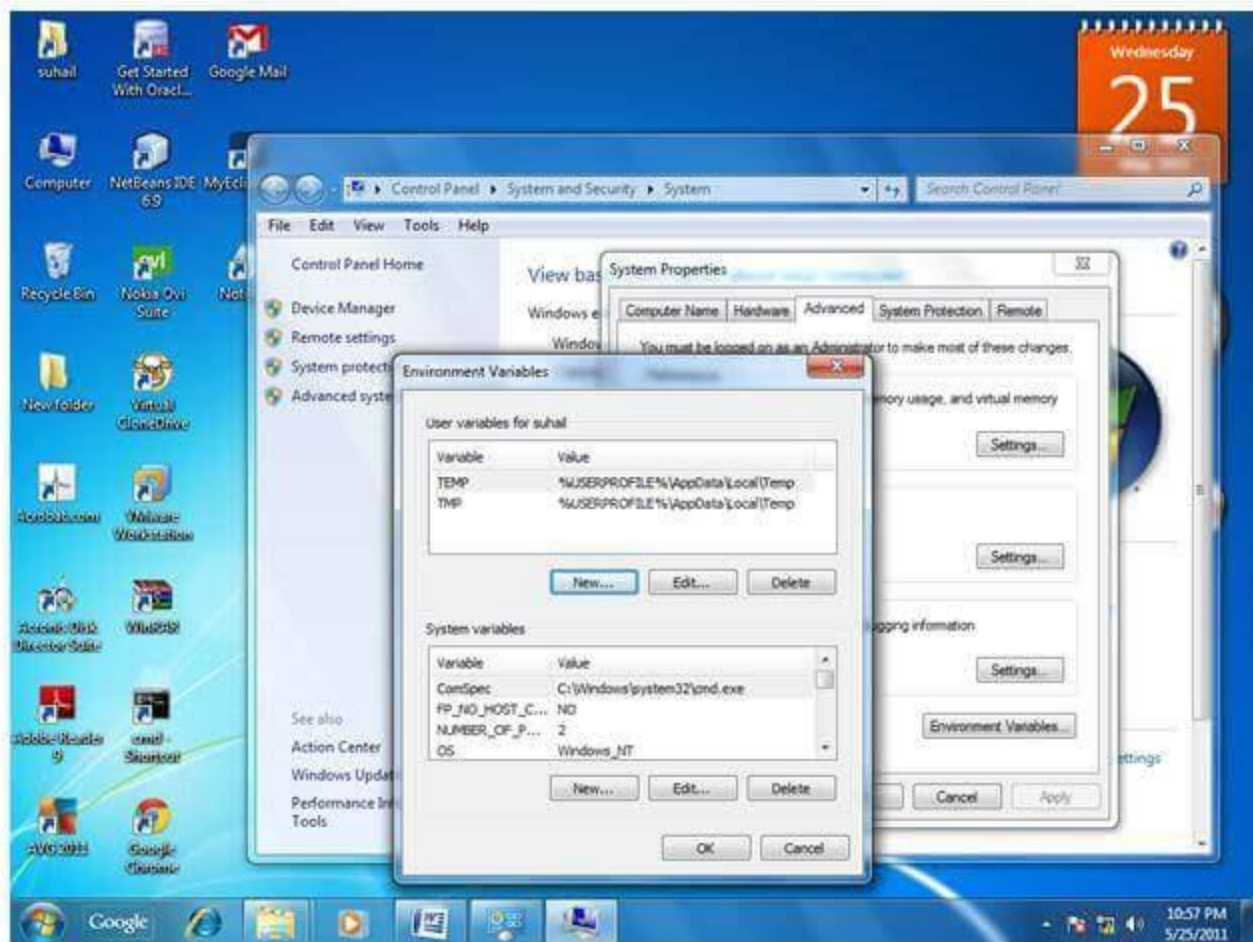


```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\Sonoo>cd\
C:\>cd new
C:\new>javac Simple.java
'javac' is not recognized as an internal or external command,
operable program or batch file.
C:\new>set path=C:\Program Files\Java\jdk1.6.0_03\bin
C:\new>javac Simple.java
C:\new>java Simple
Hello Java
C:\new>
```

How to set Permanent Path

- For setting the permanent path of JDK, you need to follow these steps:
- Go to *My Computer/ This PC - properties -> Advanced System Settings tab -> Environment variables -> new tab of user variable -> write path in variable name -> write path of bin folder in variable value -> ok -> ok - > ok*



Structure of Java Program

```
class class-name
{
    public static void main(String args[])
    {
        statement1;
        statement2;
        ...
        ...
    }
}
```

HelloWorld program

“First.java”

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

public static void main(String args[])

The diagram illustrates the components of the Java main method signature. The signature is **public static void main(String args[])**. Arrows point from labels to specific parts of the signature: 'Access Specifier' points to 'public', 'Keyword' points to 'static', 'Return type' points to 'void', 'Method name' points to 'main', and 'Array of string type' points to 'String args[]'.

Access Specifier

Keyword

Method name

Array of string type

Return type

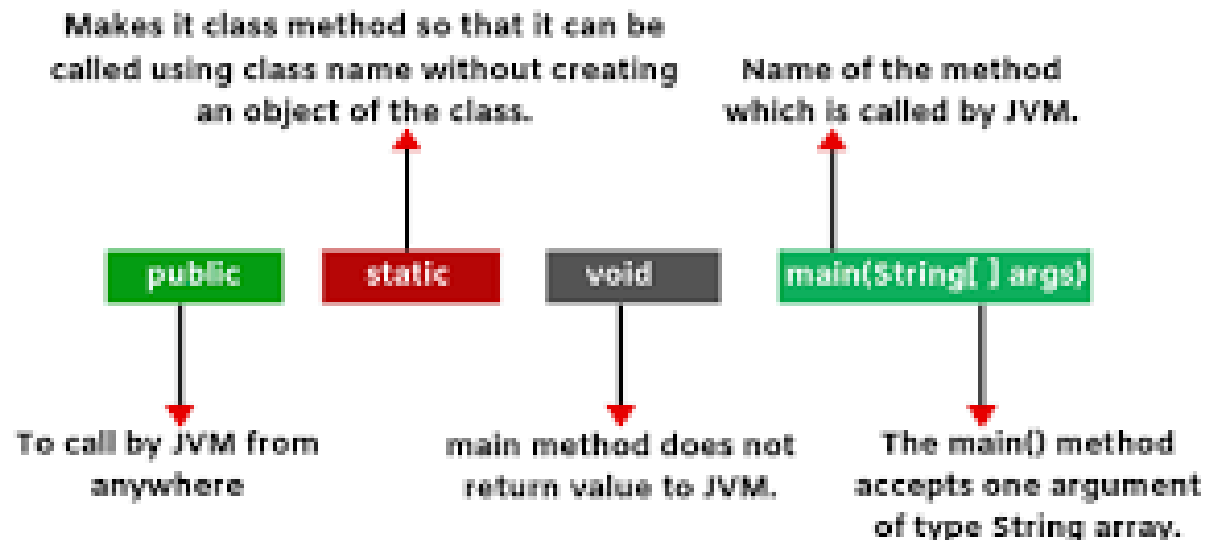


Fig: Java main method

Parameters used in First Java Program

- **class** keyword is used to declare a class in java.
- **public** keyword is an access modifier which represents visibility. It means it is visible to all.
- **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is **no need to create an object** to invoke the static method. The main method is executed by the JVM, so it doesn't require to create an object to invoke the main method. So it saves memory.

- **void** is the return type of the method. It means it doesn't return any value.
- **main** represents the starting point of the program.
- **String[] args** is used for **command line argument**. We will learn it later.
- **System.out.println()** is used to print statement. Here, System is a class, out is the object of PrintStream class, println() is the method of PrintStream class. We will learn about the internal working of System.out.println statement later.

Naming Conventions

- **Class names** should be nouns, in mixed case with the first letter of each internal word capitalized. Try to keep your class names simple and descriptive. Use whole words-avoid acronyms and abbreviations (unless the abbreviation is much more widely used than the long form, such as URL or HTML).
- E.g. First, Sample, SimpleInterest etc.
- **Program names** preferably should be same as class name.
- E.g.First.java, Sample.java, SimpleInterest.java etc.

Thank You