Object Oriented Analysis and Design using Java

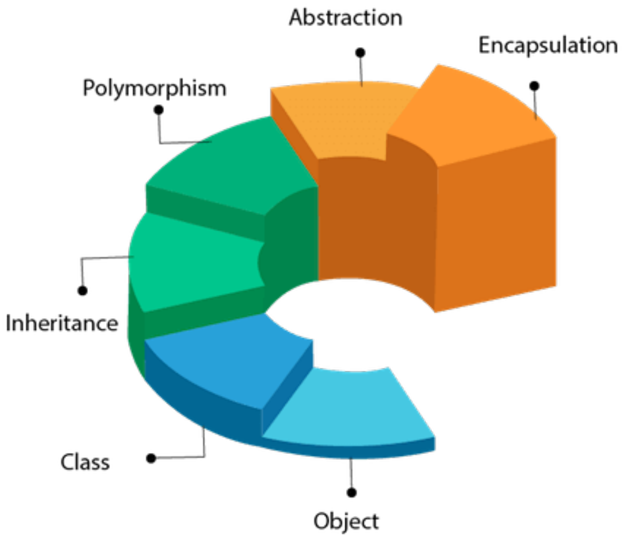
Prof: Mahitha G

Department of Computer Science and Engineering

Object Oriented Analysis and Design using Java

Introduction to OO Programming

OOPs (Object-Oriented Programming System)



Mahitha G

Department of Computer Science and Engineering

Object Oriented Analysis and Design using Java

Unit-01 : Object Oriented Programming

~~1: Introduction to course, Introduction to object-oriented concepts, Object Based Programming: JVM~~

~~2: Abstraction, Encapsulation, Composition~~

3: Class Attributes, Behaviour, Objects, and Methods

4: Interface and Implementation: Role of Constructors and Destructors, Garbage Collector

5: Parameter Passing, Value Type and Reference Type

6: Overloading of Methods Model

7: Java Recursion

8: Class Attributes and Behaviour: Difference between Class Methods and Instance Methods

9: Inheritance: Concepts of Single Rooted Hierarchy and Interface

10: Abstract Class in Programming Languages, Object Class in Java

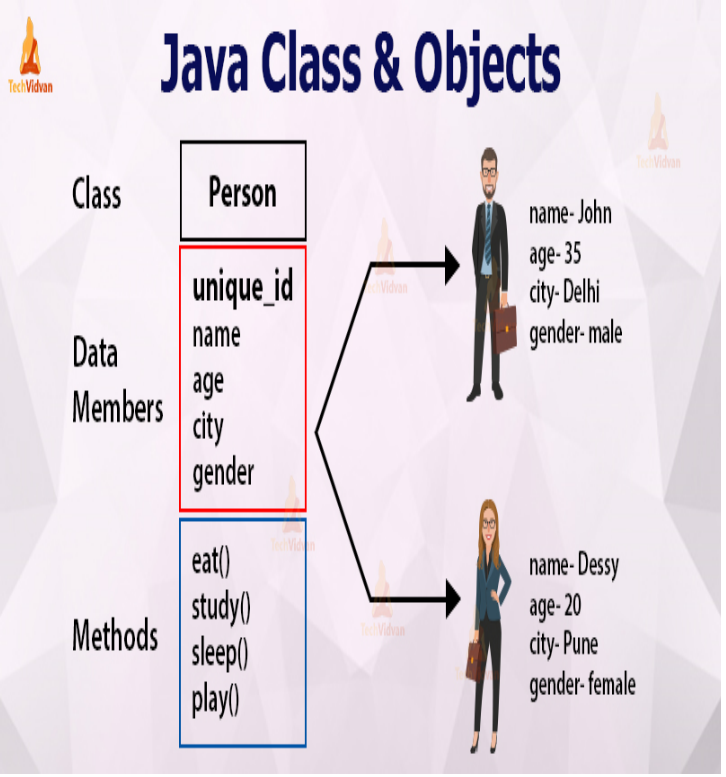
T1 : Chapter 6: Introducing Classes

1: “Java the Complete Reference”, Herbert Schildt ,McGraw-Hill ,11th Edition, 2018.

Object Oriented Programming: Classes and Objects

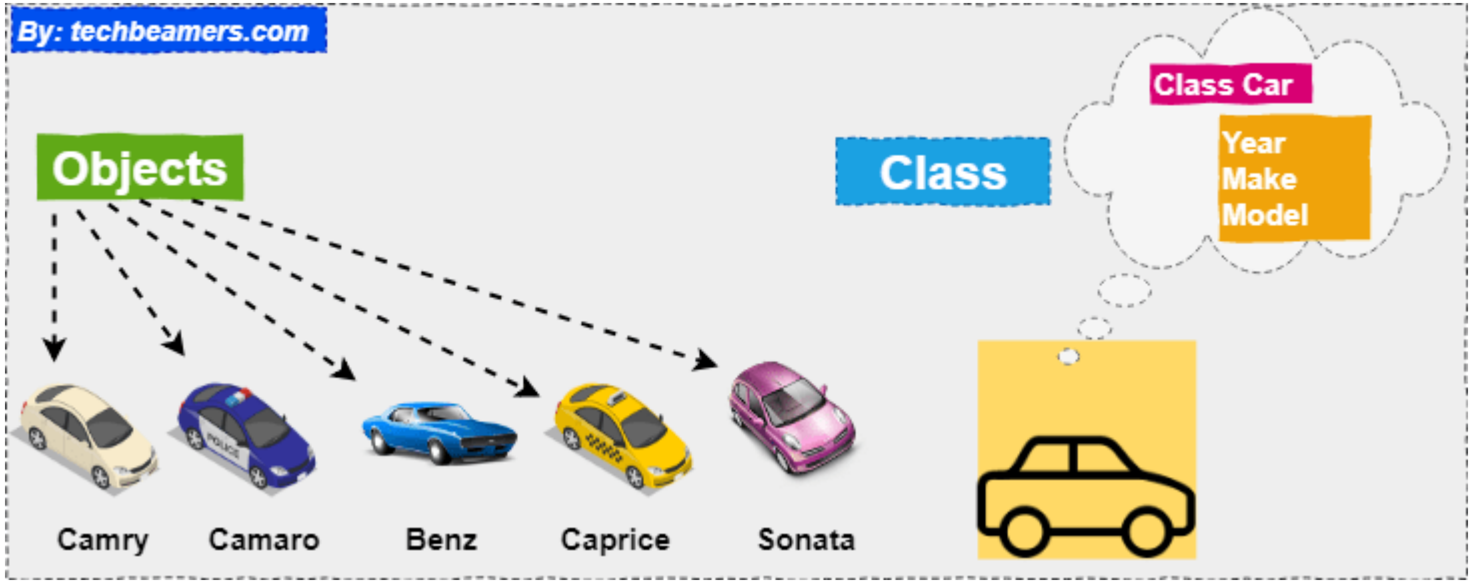
CLASSES and OBJECTS

- Defines a new data type.
- **Class is a template** for an object and an **object is an instance of a class.**
- A class may contain only data or only code or both.
- A class is the template or blueprint from which objects are made
- When you construct an object from a class, you are said to have created an instance of the class.
- Any concept to be implemented in java must be encapsulated within a class.
- **Think about classes as cookie cutters. Objects are the cookies themselves.**



Object Oriented Analysis and Design using Java

Object Oriented Programming: Class and Objects

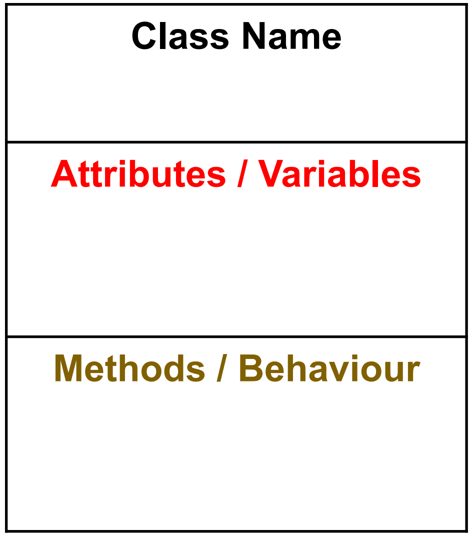


Object Oriented Programming: Class and Objects

General definition of a class in java:

```
class class_name {
data_type instance_variable1;
data_type instance_variable2;……..
data_type method1() {…//body of the method}
data_type method1() {…//body of the method}
…….
}
```

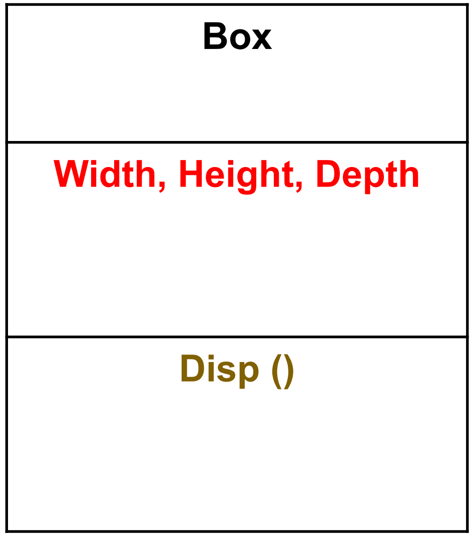
Classes contain instance variables and methods



Class Name

Attributes / Variables

Methods / Behaviour



Box

Width, Height, Depth

Disp ()

Object Oriented Programming: Class and Objects

To work with OOP, you should be able to identify three key characteristics of objects

1. **The object’s behavior-**What can you do with this object, or what methods can you apply to it?
2. **The object’s state**—how does the object react when you invoke those methods?
3. **The object’s identity**—how is the object distinguished from others that may have the same behavior and state?

- All objects that are instances of the same class share a family resemblance by supporting the same behavior.

- The behavior of an object is defined by **the methods that you can call.**

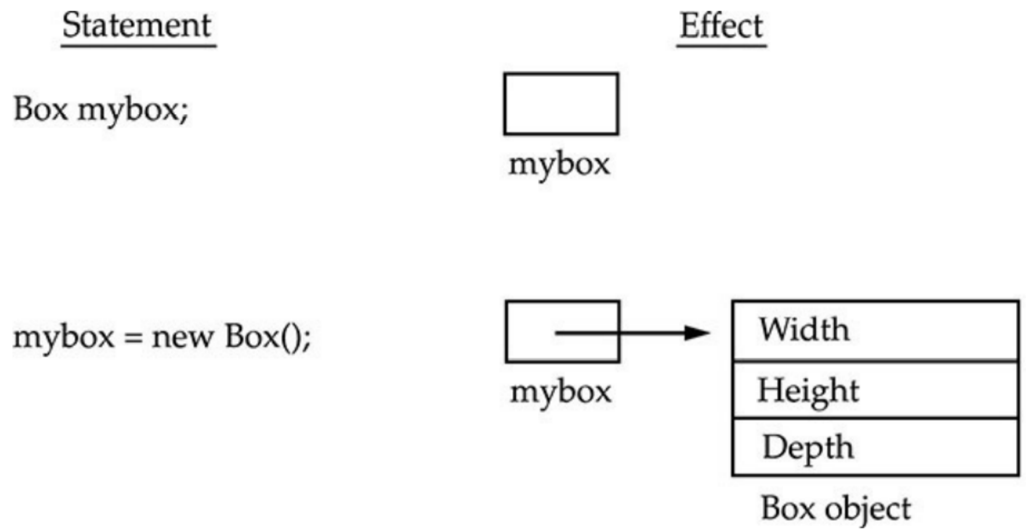
Object Oriented Programming: Class and Objects - Example

Creation of a Class:

```
class Box
{
        double width;
        double height;
        double depth;
        void disp()
        {
                System.out.println("width: "+width);
                System.out.println("height: "+height);
                System.out.println("depth: "+depth);
        }
}
```

Instantiation of an object:

Box mybox = new Box()



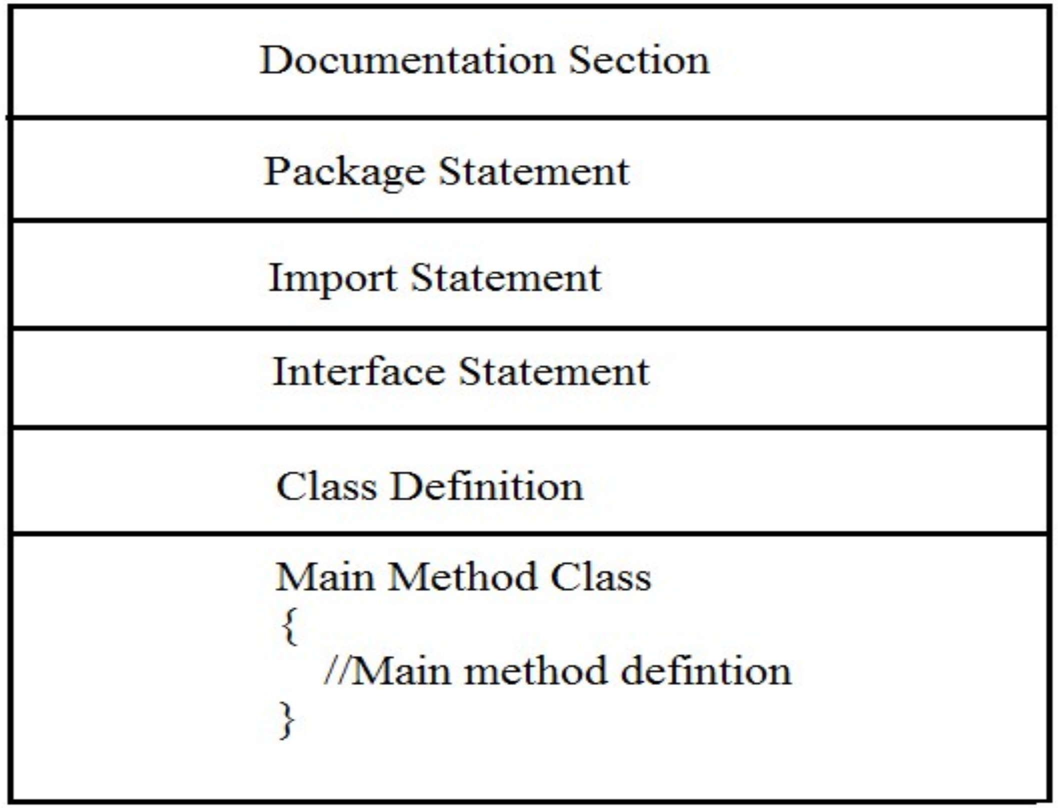
Object Oriented Programming: Access Modifers

The access modifiers in Java specifies the accessibility or scope of a field, method, constructor, or class. We can change the access level of fields, constructors, methods, and class by applying the access modifier on it.

- There are four types of Java access modifiers:
- **Private**: The access level of a private modifier is only within the class. It cannot be accessed from outside the class.
- **Default**: The access level of a default modifier is only within the package. It cannot be accessed from outside the package. If you do not specify any access level, it will be the default.
- **Protected**: The access level of a protected modifier is within the package and outside the package through child class. If you do not make the child class, it cannot be accessed from outside the package.
- **Public**: The access level of a public modifier is everywhere. It can be accessed from within the class, outside the class, within the package and outside the package.

Note: Examples for Default, Protected will be discussed later

Object Oriented Programming: General Java Program Structure



Documentation Section

Package Statement

Import Statement

Interface Statement

Class Definition

```
Main Method Class
{
    //Main method defintion
}
```

Object Oriented Analysis and Design using Java

Object Oriented Programming: Simple Java Program

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

Object Oriented Programming: Java Program Execution

Compiling and launching a Java program from the **command line** once JDK is installed.

1: Open a command prompt window and go to the directory where you saved the java program (MyFirstJavaProgram.java).

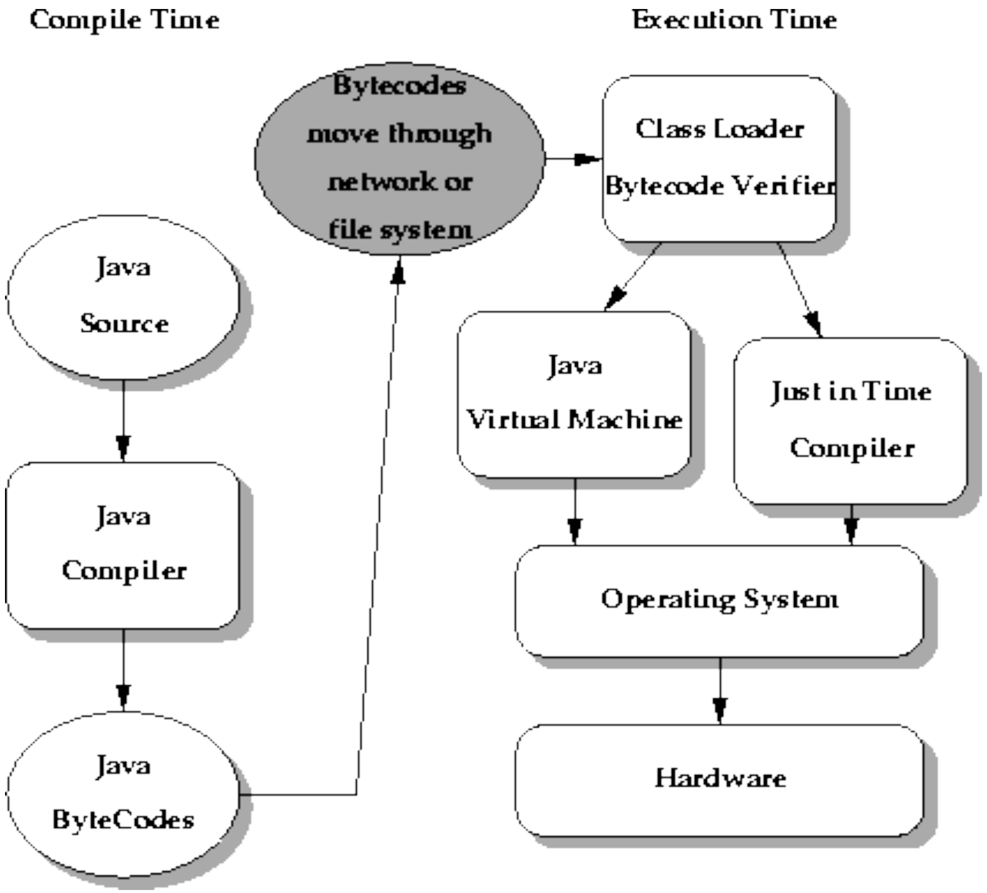
2: Type 'javac MyFirstJavaProgram.java' and press enter to compile your code.
If there are no errors in your code, the command prompt will take you to the next line

3: Type ' java MyFirstJavaProgram ' to run your program.
You will be able to see the result printed on the window.

(Or)

Use IDE – Eclipse, NetBeans, IntelliJ, BlueJ…..

Object Oriented Programming: Translation Process



Object Oriented Programming: Access Modifiers Private- Example

```
class A{
private int data=10;
private void msg(){System.out.println(“Welcome to OOAD with Java class");}
}

public class Sample{
 public static void main(String args[]){
   A obj=new A();
   System.out.println(obj.data);                    //Compile Time Error
   obj.msg();                                       //Compile Time Error
   }
}
//accessing the private members from outside the class, so there is a compile-time error.
```

Object Oriented Programming: Access Modifiers - Example

If you make any class constructor private, you cannot create the instance of that class from outside the class.

```
class A{
private A()  { }                                   //private constructor
void msg(){System.out.println("Welcome to OOAD with java class");}
}
public class Sample
{
 public static void main(String args[]){
   A obj=new A();                                  //Compile Time Error
 }
}
```

Object Oriented Programming: Access Modifiers Public - Example

The **public access modifier** is accessible everywhere. It has the widest scope among all other modifiers.

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

PES UNIVERSITY ONLINE

THANK YOU

Mahitha G

Department of Computer Science and Engineering

mahithag@pes.edu