@kushacodescript

OOPS

Object-oriented programming

OOPs **concepts**

What is oops?
Object-oriented programming System(OOPs) is a programming paradigm based on the concept of “objects” that contain data and methods.

Why we learn oops?
The primary purpose of object-oriented programming is to increase the flexibility and maintainability of programs. Object oriented programming brings together data and its methods in a single object makes it easier to understand how a program works.

We will cover each and every feature of OOPs in detail so that you won’t face any difficultly understanding OOPs Concepts.

```
package basics;

public class MainClass {

    public static void main(String[] args) {
        System.out.println("Hello world");
    }

}
```

Class in OOPs Concepts

A class can be considered as a blueprint using which you can create as many objects as you like.

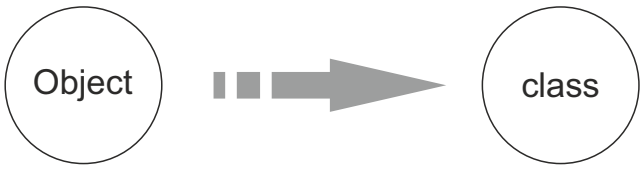
For example, here we have a class Website that has two data members (also known as fields, instance variables and object states). This is just a blueprint, it does not represent any website, however using this we can create Website objects (or instances) that represents the websites. We have created two objects, while creating objects we provided separate properties to the objects using constructor.

```
public class Website {
   //fields (or instance variable)
   String webName;
   int webAge;

   // constructor
   Website(String name, int age){
      this.webName = name;
      this.webAge = age;
   }
   public static void main(String args[]){
      //Creating objects
      Website obj1 = new Website("beginnersbook", 5);
      Website obj2 = new Website("google", 18);

     //Accessing object data through reference
     System.out.println(obj1.webName+" "+obj1.webAge);
     System.out.println(obj2.webName+" "+obj2.webAge);
   }
}
```

Object

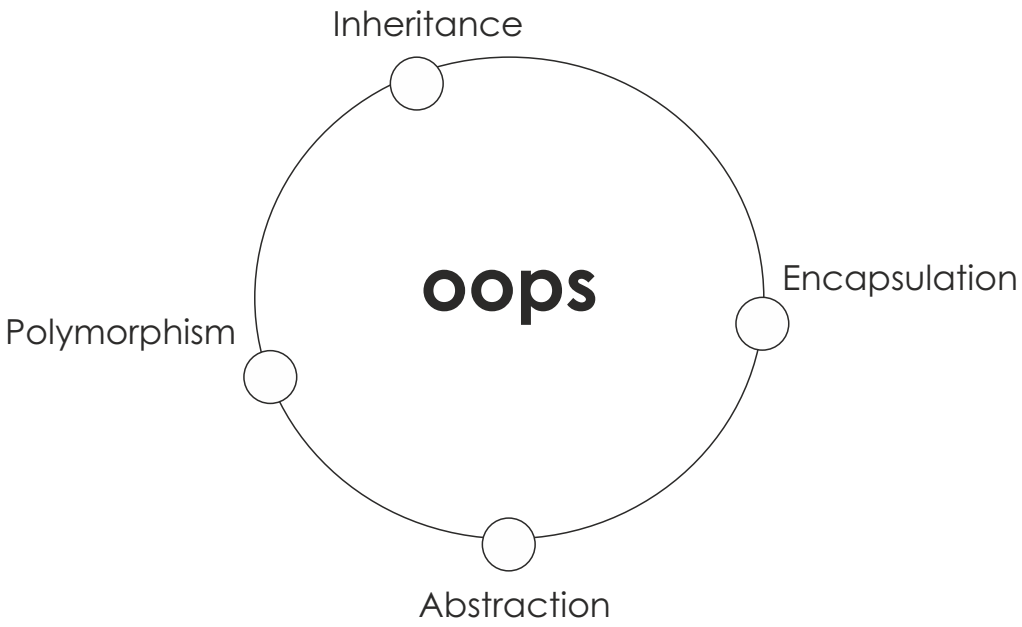
bundle of data and its states (is Attributes) and behaviors (is Method).

So if I had to write a class based on states and behaviours of Human. I can do it like this: States can be represented as instance variables and behaviours as methods of the class.




```
class Human
{
    int age;
    String name;

    void walking()
    {
        //Write code here
    }
    void eating()
    {
        //Write code here
    }
}
```

OOPs Features



These four features are the main OOPs Concepts that you must learn to understand the Object Oriented Programming in Java

OOPs Feature - 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 creating account on gmail, you enter your user_id, emailId, password and press Sign-up, what happens when you press Sign-up, how the input data sent to server, how it gets verified is all abstracted away from the you.

OOPs Feature - Encapsulation

Binding of data and methods that works on it in one unit is called encapsulation.

We understand it with example check below:

```
package com.javatpoint;
public class Student{
//private data member
private String name;
//getter method for name
public String getName(){
return name;
}
//setter method for name
public void setName(String name){
this.name=name
}
}
```

```
//A Java class to test the encapsulated class.
package com.javatpoint;
class Test{
public static void main(String[] args){
//creating instance of the encapsulated class
Student s=new Student();
//setting value in the name member
s.setName("vijay");
//getting value of the name member
System.out.println(s.getName());
}
}
```

OOPs Feature - Inheritance

1. Defining a new class based on an existing class by extending its common data members and methods.

2. Inheritance allows us DRY (do not repeat yourself) Method, it improves reusability in your application.

3. The parent class is called the base class or super class. The child class that extends the base class is called the derived class or sub class or child class.

```
class Subclass-name extends Superclass-name
{
  //methods and fields
}
```

```
class Employee{
 float salary=40000;
}
class Programmer extends Employee{
 int bonus=10000;
 public static void main(String args[]){
   Programmer p=new Programmer();
   System.out.println("Programmer salary is:"+p.salary);
   System.out.println("Bonus of Programmer is:"+p.bonus);
}
}
```

Types of Inheritance:

Single Inheritance | Hierarchical inheritance | Multiple Inheritance

OOPs Feature - Polymorphism

Polymorphism is a object oriented programming feature that allows us to perform a single action in different ways. For example, lets say we have a class Animal that has a method animalSound(), here we cannot give implementation to this method as we do not know which Animal class would extend Animal class. So, we make this method abstract like this:

```
class Bike{
  void run(){System.out.println("running");}
}
class Splendor extends Bike{
  void run(){System.out.println("running safely with 60km");}

  public static void main(String args[]){
    Bike b = new Splendor();//upcasting
    b.run();
  }
}
```

Types of Polymorphism:

Static Polymorphism | Dynamic Polymorphism

Thank You