

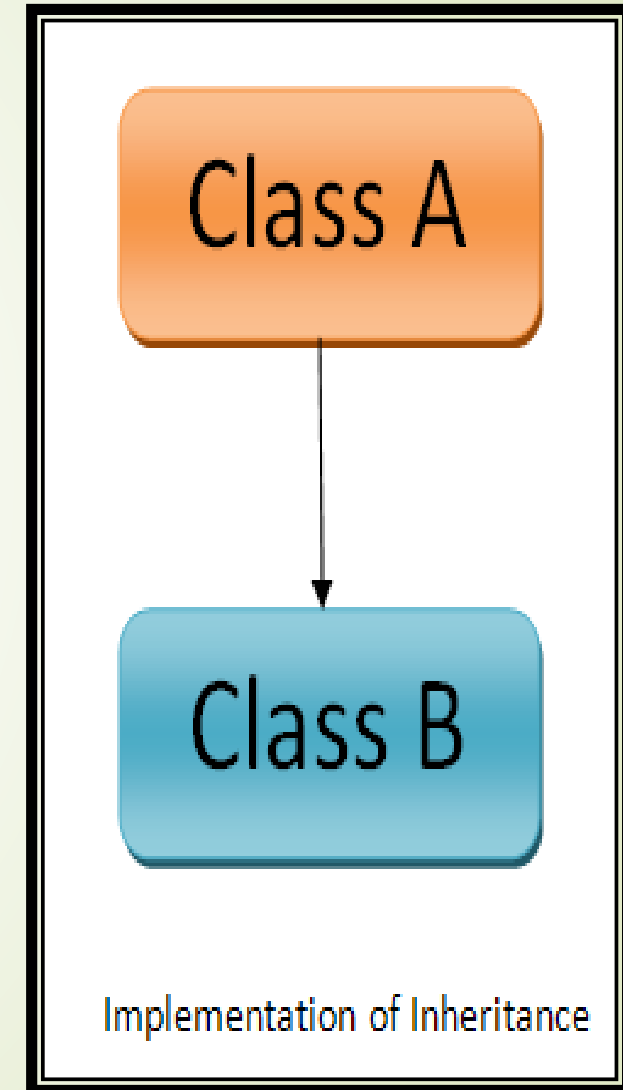


# **INHERITANCE IN JAVA**

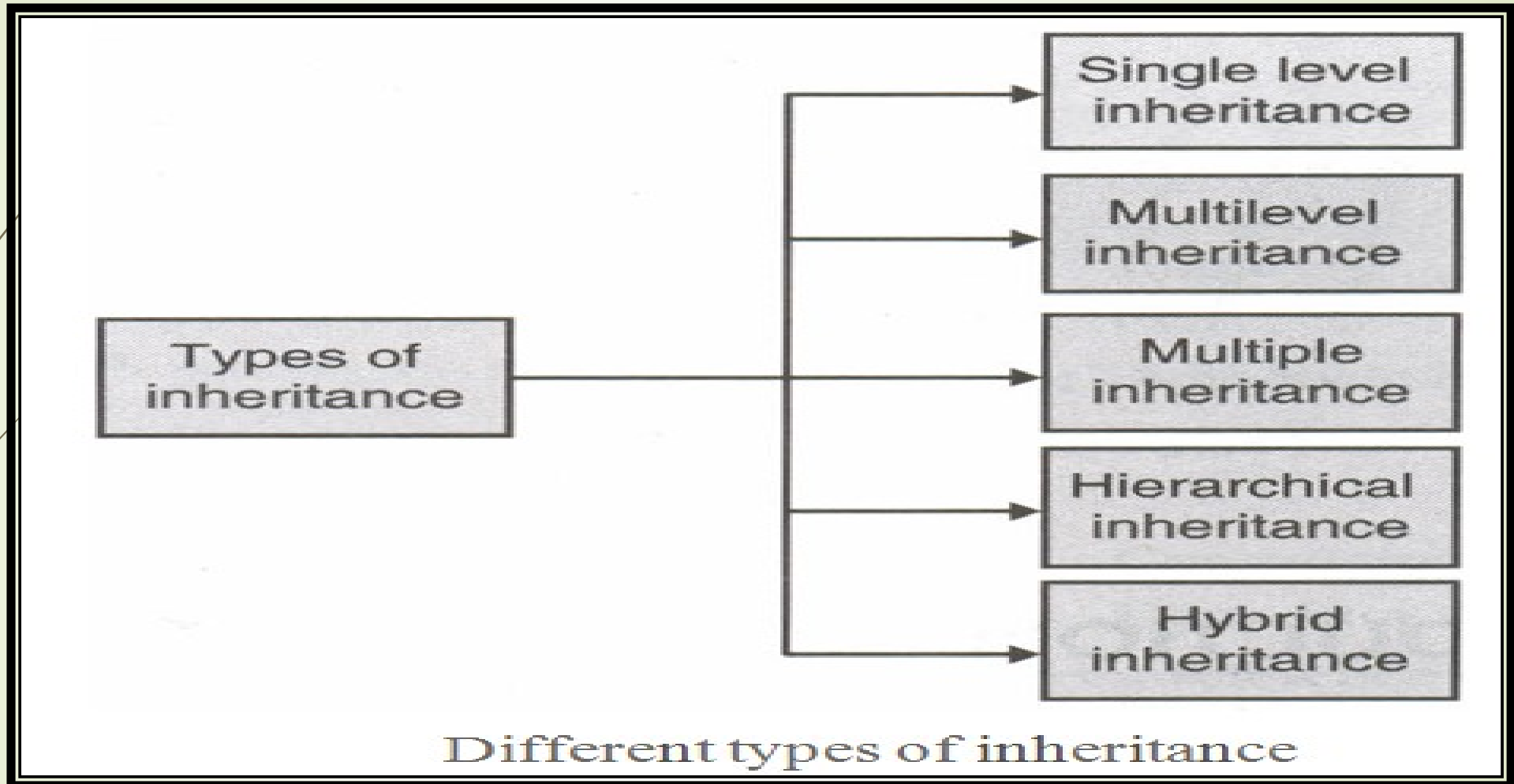
By

Dr. Amal S.Ajrash

- ❖ The term inheritance implies that one class can inherit a part or all of its structure and behavior from another class. In other words the inheritance in java a mechanism of deriving **new class** from **old class**.
- ❖ Inheritance provides the idea of reusability, i.e., a code once written can be used again and again in a number of new **classes**.
- ❖ The old class is known as- **Base Class** / Super class / Parent Class
- ❖ The new class is known as- **Sub Class** / Derived class/ Child class
- ❖ A subclass can add to the structure and behavior that it **inherits**.
- ❖ It can also replace or modify inherited behavior (though not **inherited structure**).



# TYPES OF INHERITANCE



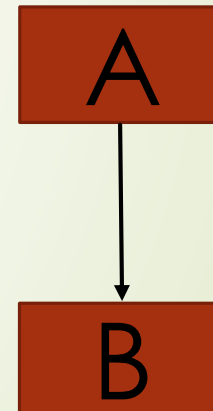
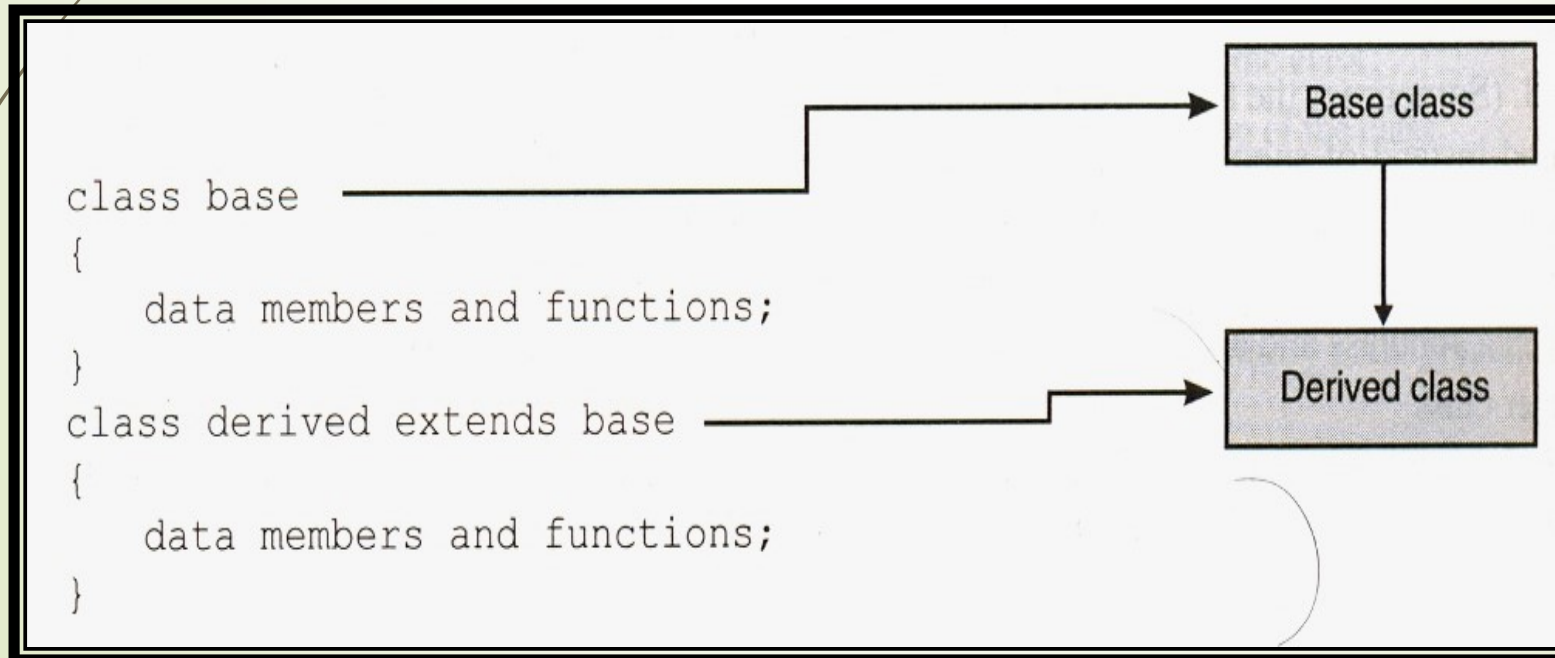
Inheritance is generally of five types : single level, multilevel, multiple, hierarchical and hybrid.

➤ **Single inheritance -**

➤ When a class extends another **one class** only then we call it a single inheritance.



➤ The below flow diagram shows that class B extends only one class which is A.

➤ Here A is a **parent class** of B and B would be a **child class** of A.



# Single Inheritance example program in Java

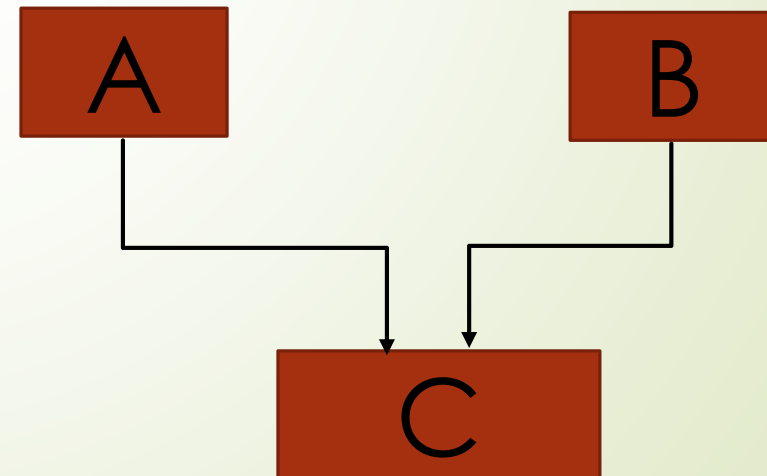
```
Class A
{
    public void methodA()
    {
        System.out.println("Base class method");
    }
}
Class B extends A
{
    public void methodB()
    {
        System.out.println("Child class method");
    }
    public static void main(String args[])
    {
        B obj = new B();
        obj.methodA(); //calling super class method
        obj.methodB(); //calling local method
    }
}
```



```
class Vehicle
{
    Vehicle()
    {
        System.out.println("Vehicle is created");
    }
}
class Bike extends Vehicle{
    Bike()
    {
        System.out.println("Bike is created");
    }
}
public static void main(String args[]){
    Bike b=new Bike();
    b.Vehicle();
    b.Bike();
}
}
```

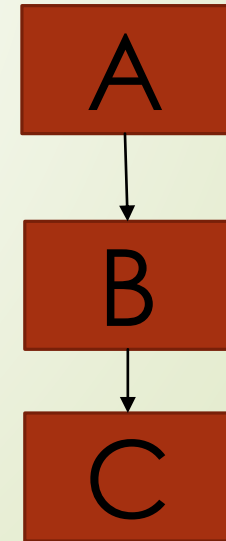
# “Multiple Inheritance”

- “Multiple Inheritance” refers to the concept of one class extending (Or inherits) **more than one base class**.
- The problem with “multiple inheritance” is that the derived class will have to manage the dependency on two base classes.



# Multilevel Inheritance

- **Multilevel inheritance** refers to a mechanism in OO technology where one can inherit from a derived class, thereby making this derived class the base class for the new class.
- As you can see in below flow diagram C is subclass or child class of B and B is a child class of A.

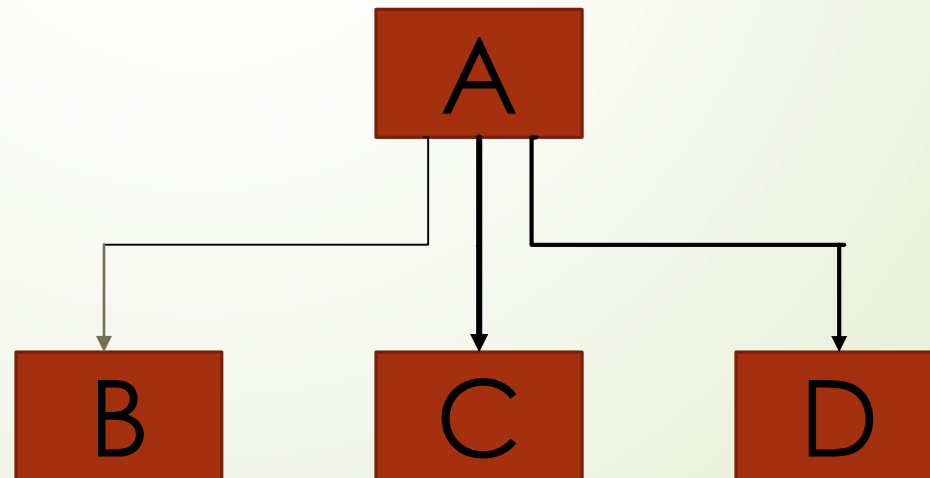




```
Class X
{
    public void methodX()
    {
        System.out.println("Class X method");
    }
}
Class Y extends X
{
    public void methodY()
    {
        System.out.println("class Y method");
    }
}
Class Z extends Y
{
    public void methodZ()
    {
        System.out.println("class Z method");
    }
    public static void main(String args[])
    {
        Z obj = new Z();
        obj.methodX(); //calling grand parent class method
        obj.methodY(); //calling parent class method
        obj.methodZ(); //calling local method
    }
}
```

# Hierarchical Inheritance

- In such kind of inheritance one class is inherited by many **sub classes**.
- In below example class B,C and D **inherits** the same class A.
- A is **parent class (or base class)** of B,C & D.



```
Class A
{
    public void methodA()
    {
        System.out.println("method of Class A");
    }
}
Class B extends A
{
    public void methodB()
    {
        System.out.println("method of Class B");
    }
}
Class C extends A
{
    public void methodC()
    {
        System.out.println("method of Class C");
    }
}
```

```
Class D extends A
```

```
{  
    public void methodD()  
    {  
        System.out.println("method of Class D");  
    }  
}
```

```
Class MyClass
```

```
{  
    public static void main(String args[])  
    {  
        B obj1 = new B();  
        C obj2 = new C();  
        D obj3 = new D();  
        obj1.methodA();  
        obj2.methodA();  
        obj3.methodA();  
    }  
}
```

Programmer.java x

```
1 //Simple Demo of Inheritance
2 class Employee
3 {
4     float salary=40000;
5     protected int da=2000; //private show error
6 }
7 class Programmer extends Employee
8 {
9     int bonus=10000;
10    public static void main(String args[])
11    {
12        Programmer p=new Programmer();
13        System.out.println("Programmer salary is:"+p.salary);
14        System.out.println("Bonus of Programmer is:"+p.bonus);
15        System.out.println("DA of Programmer is:"+p.da);
16    }
17 }
```

teacher\_demo.java

```
1 //Demo for Inheritance //Student class inherit
2 //teacher class. Teacher.class file is generated
3 class Teacher
4 {   int id;
5     String name,address;
6     float sal;
7     void setid(int id)
8     {   this.id=id; }
9     int getid()
10    {   return id; }
11    void setname(String name)
12    {   this.name=name; }
13    String getname()
14    {   return name; }
15    void setaddress(String address)
16    {   this.address=address; }
17    String getaddress()
18    {   return address; }
19    void setsal(float sal)
20    {   this.sal=sal; }
21    float getsal()
22    {   return sal; }
23 }
```

```
7  class teacher_demo
8  {
9      public static void main(String args[])
10     {
11         Teacher t1=new Teacher();
12         t1.setid(2001);
13         t1.setname("Ramanuj");
14         t1.setaddress("Haldia West Bengal");
15         t1.setsal(40000f);
16         System.out.println("id = "+t1.getid());
17         System.out.println("Name = "+t1.getname());
18         System.out.println("Address = "+t1.getaddress());
19         System.out.println("Salary = "+t1.getsal());
20     }
21 }
```

**Output:**

**Id= 2001**

**Name = Ramanuj**

**Address = Haldia West Bengal**

**Salary = 40000.0**

```
student_demo.java
1 //Student class will inherit Teacher class
2 //Go to teacher_demo.java
3 //Instead of salary we will add new field marks
4 class Student extends Teacher
5 {
6     //Since id,name,address are already available
7     //in Teacher class we omit those instance
8     //variables and corresponding methods
9     int marks; // Student variable
10    void setmarks(int marks)
11    {    this.marks=marks;    }
12
13    int getmarks()
14    {    return marks;    }
15 }
16 class student_demo
17 {
18     public static void main(String args[])
19     {
20         Student s1=new Student();
21         s1.setid(1001); //belongs to Teacher class
22         s1.setname("Ajay");//belongs to Teacher class
23         s1.setaddress("Kolkata West Bengal");
24         s1.setmarks(91);
25         System.out.println("id = "+s1.getid());
26         System.out.println("Name = "+s1.getname());
27         System.out.println("Address = "+s1.getaddress());
28         System.out.println("Marks = "+s1.getmarks());
29     }
30 }
```

Output:

Id= 1001

Name = Ajay

Address = Kolkata West Bengal

Marks = 91



Thank you!

