Inheritance in Java

Java, Inheritance is an important pillar of OOP(Object-Oriented Programming). It is the mechanism in Java by which one class is allowed to inherit the features(fields and methods) of another class. In Java, Inheritance means creating new classes based on existing ones. A class that inherits from another class can reuse the methods and fields of that class. In addition, you can add new fields and methods to your current class as well.

Why Do We Need Java Inheritance?

Code Reusability: The code written in the Superclass is common to all subclasses. Child classes can directly use the parent class code.

Method Overriding: Method Overriding is achievable only through Inheritance. It is one of the ways by which Java achieves Run Time Polymorphism.

Abstraction: The concept of abstract where we do not have to provide all details is achieved through inheritance. Abstraction only shows the functionality to the user.

The syntax of Java Inheritance

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

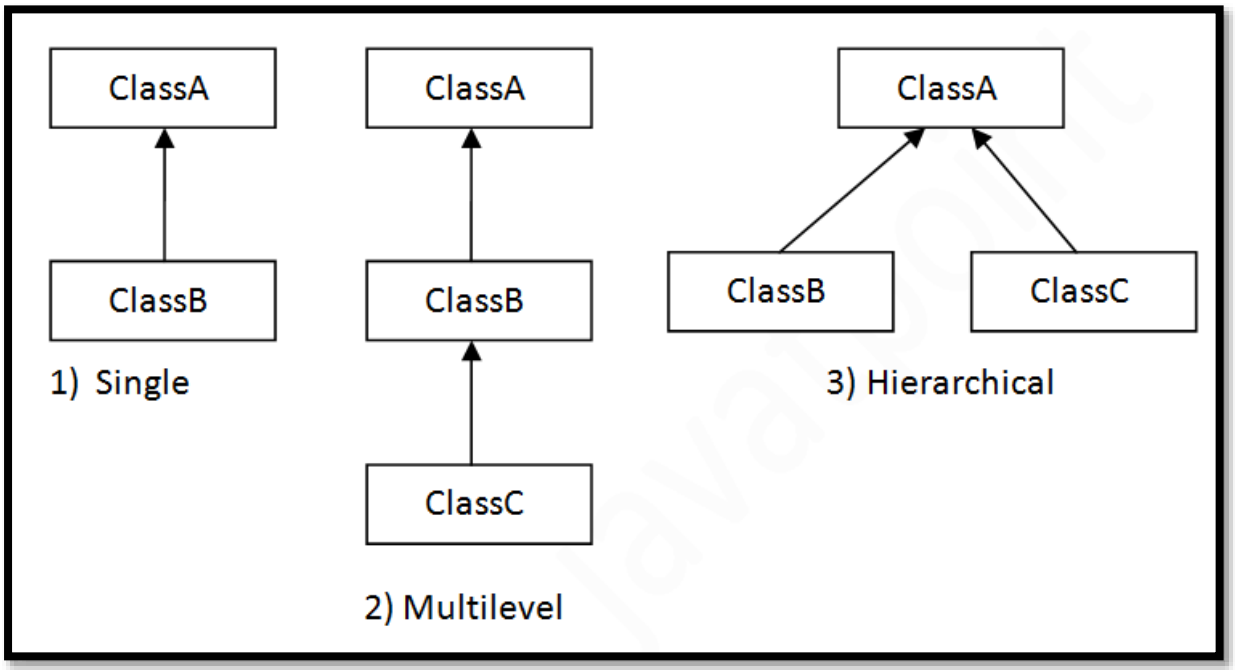
```
class Employee
{
 float salary=40000;
}
class Programmer extends Employee
{
 int bonus=10000;
```

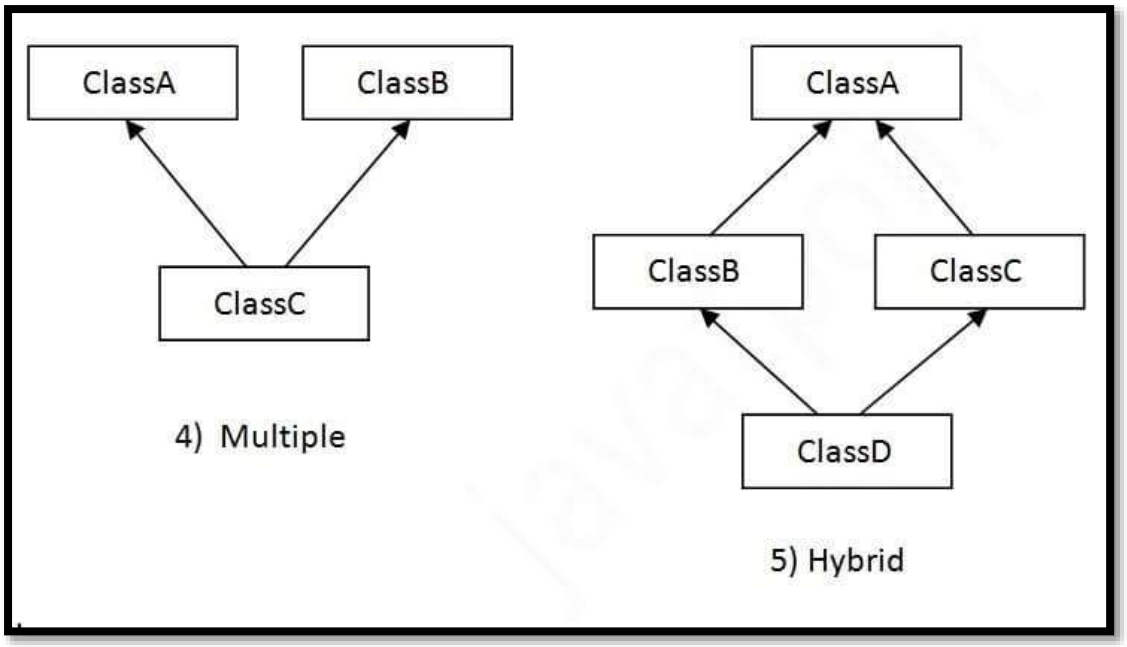
```
 public static void main(String args[])
{
  Programmer p=new Programmer();
  System.out.println("Programmer salary is:"+p.salary);
  System.out.println("Bonus of Programmer is:"+p.bonus);
}
}
```

Types of inheritance in java

On the basis of class, there can be three types of inheritance in java: single, multilevel and hierarchical.

In java programming, multiple and hybrid inheritance is supported through interface only. We will learn about interfaces later





Single Inheritance Example

When a class inherits another class, it is known as a single inheritance. In the example given below, Dog class inherits the Animal class, so there is the single inheritance.

Save File: TestInheritance.java

```
class Animal
{
void eat()
{
System.out.println("eating...");
}
}
class Dog extends Animal
{
void bark()
{
System.out.println("barking...");
```

```
}
}
class TestInheritance
{
public static void main(String args[])
{
Dog d=new Dog();
d.bark();
d.eat();
}
}
```

Multilevel Inheritance Example

When there is a chain of inheritance, it is known as multilevel inheritance. As you can see in the example given below, BabyDog class inherits the Dog class which again inherits the Animal class, so there is a multilevel inheritance.

Save File: TestInheritance2.java

```
class Animal
{
void eat()
{
System.out.println("eating...");
}
}
class Dog extends Animal
{
void bark()
{
```

```
System.out.println("barking...");
}
}
class BabyDog extends Dog
{
void weep()
{
System.out.println("weeping...");
}
}
class TestInheritance2
{
public static void main(String args[])
{
BabyDog d=new BabyDog();
d.weep();
d.bark();
d.eat();
}
}
```

Hierarchical Inheritance Example

When two or more classes inherits a single class, it is known as hierarchical inheritance. In the example given below, Dog and Cat classes inherits the Animal class, so there is hierarchical inheritance.

Save File: TestInheritance3.java

```
class Animal
{
```

```
void eat()
{
System.out.println("eating...");
}
}
class Dog extends Animal
{
void bark()
{
System.out.println("barking...");
}
}
class Cat extends Animal
{
void meow()
{
System.out.println("meowing...");
}
}
class TestInheritance3
{
public static void main(String args[])
{
Cat c=new Cat();
c.meow();
c.eat();
```

```
}

}
```