Object Oriented Programming

INHERITANCE

Inheritance

- A mechanism in which one object acquires all the properties and behaviors of a parent object.
- When you inherit from an existing class, you can reuse methods and fields of the parent class. Moreover, you can add new methods and fields in your current class also.
- Inheritance represents the IS-A relationship which is also known as a parent-child relationship.

Inheritance

Terms used in Inheritance:

Class: A class is a group of objects which have common properties. It is a template or blueprint from which objects are created.

Sub Class/Child Class: Subclass is a class which inherits the other class. It is also called a derived class, extended class, or child class.

Super Class/Parent Class: Superclass is the class from where a subclass inherits the features. It is also called a base class or a parent class.

Reusability: As the name specifies, reusability is a mechanism which facilitates you to reuse the fields and methods of the existing class when you create a new class. You can use the same fields and methods already defined in the previous class.

Inheritance

Syntax:

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

- The **extends keyword** indicates that you are making a new class that derives from an existing class.

Inheritance Example

```
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);
}
}
```

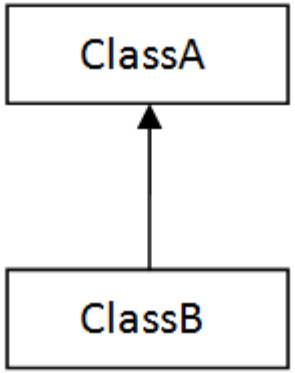
Output:

```
Programmer salary is:40000.0
Bonus of programmer is:10000
```

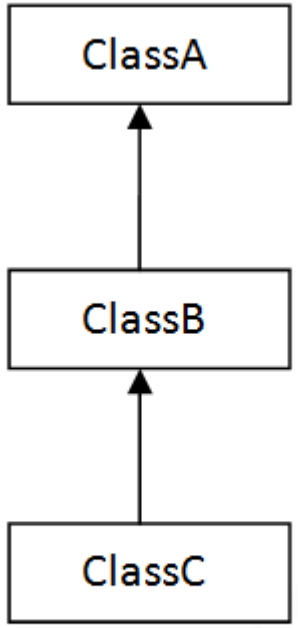
Types of Inheritance

1. Single Inheritance
2. Multilevel Inheritance
3. Multiple Inheritance
4. Hierarchical Inheritance
5. Hybrid Inheritance

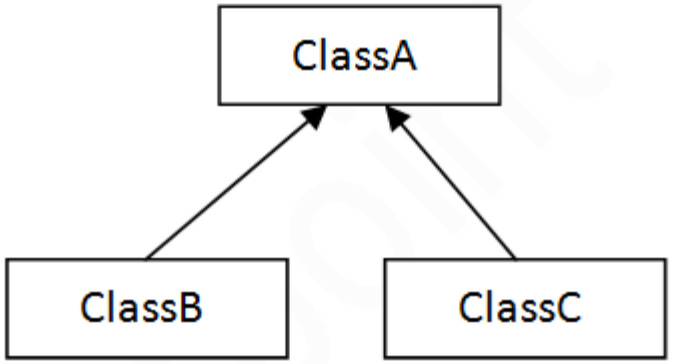
Types of Inheritance



1) Single

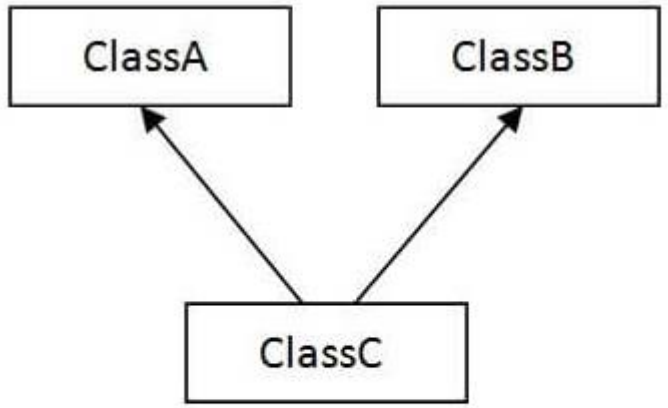


2) Multilevel

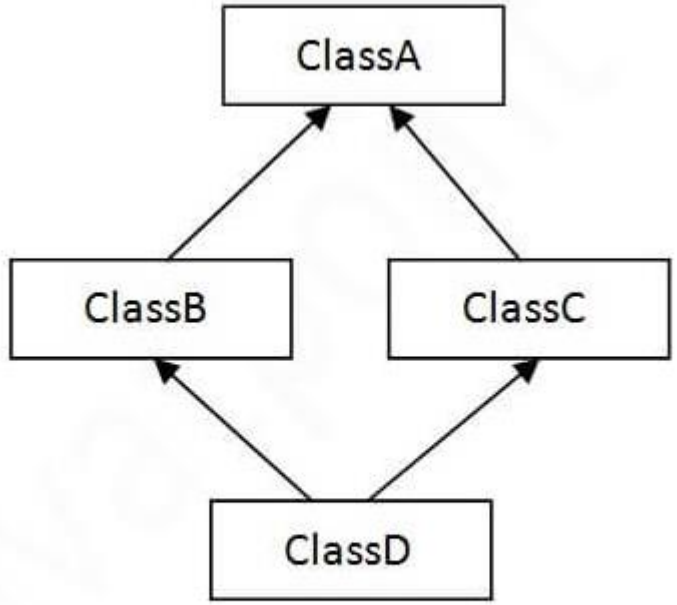


3) Hierarchical

Types of Inheritance



4) Multiple



5) Hybrid

Types of Inheritance in JAVA

1. Single Inheritance
2. Multilevel Inheritance
3. Hierarchical Inheritance

In java programming, **multiple** and **hybrid** inheritance is supported through interface only.

Single Inheritance

- When a **class inherits another class**, it is known as a single inheritance.

In the example, **Dog class inherits the Animal class**, so there is the single inheritance.

Example:

```
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();
}}
```

Output:

```
barking...
eating...
```

Multilevel Inheritance

- When there is a chain of inheritance, it is known as multilevel inheritance.
- Suppose, for Class A, Class B and Class C ,

  C class inherits B class which again inherits the A class, so there is a multilevel inheritance.

Multilevel Inheritance Example

BabyDog class inherits the Dog class which again inherits the Animal class, so there is a multilevel inheritance.

Example:

```
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();
}}
```

Output:

```
weeping...
barking...
eating...
```

Hierarchical Inheritance

- When two or more classes inherits a single class, it is known as hierarchical inheritance.

Example:

```
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();
//c.bark();//C.T.Error
}}
```

Output:

```
meowing...
eating...
```

Why multiple inheritance is not supported in java?

- To reduce the complexity and simplify the language
- To ignore ambiguity while calling method

Ambiguity of calling method

Consider a scenario where A, B, and C are three classes. The C class inherits A and B classes. If A and B classes have the same method and you call it from child class object, there will be ambiguity to call the method of A or B class.

Example:

```
class A{
void msg(){System.out.println("Hello");}
}
class B{
void msg(){System.out.println("Welcome");}
}
class C extends A,B{//suppose if it were

 public static void main(String args[]){
   C obj=new C();
   obj.msg();//Now which msg() method would be invoked?
}
}
```

Output:

Compile Time Error

Thank You