Object Oriented Analysis and Design with Java

UE19CS353

Dr. L. Kamatchi Priya
Department of Computer Science and Engineering

UE19CS353: Object Oriented Analysis and Design with Java

Object Oriented Concepts

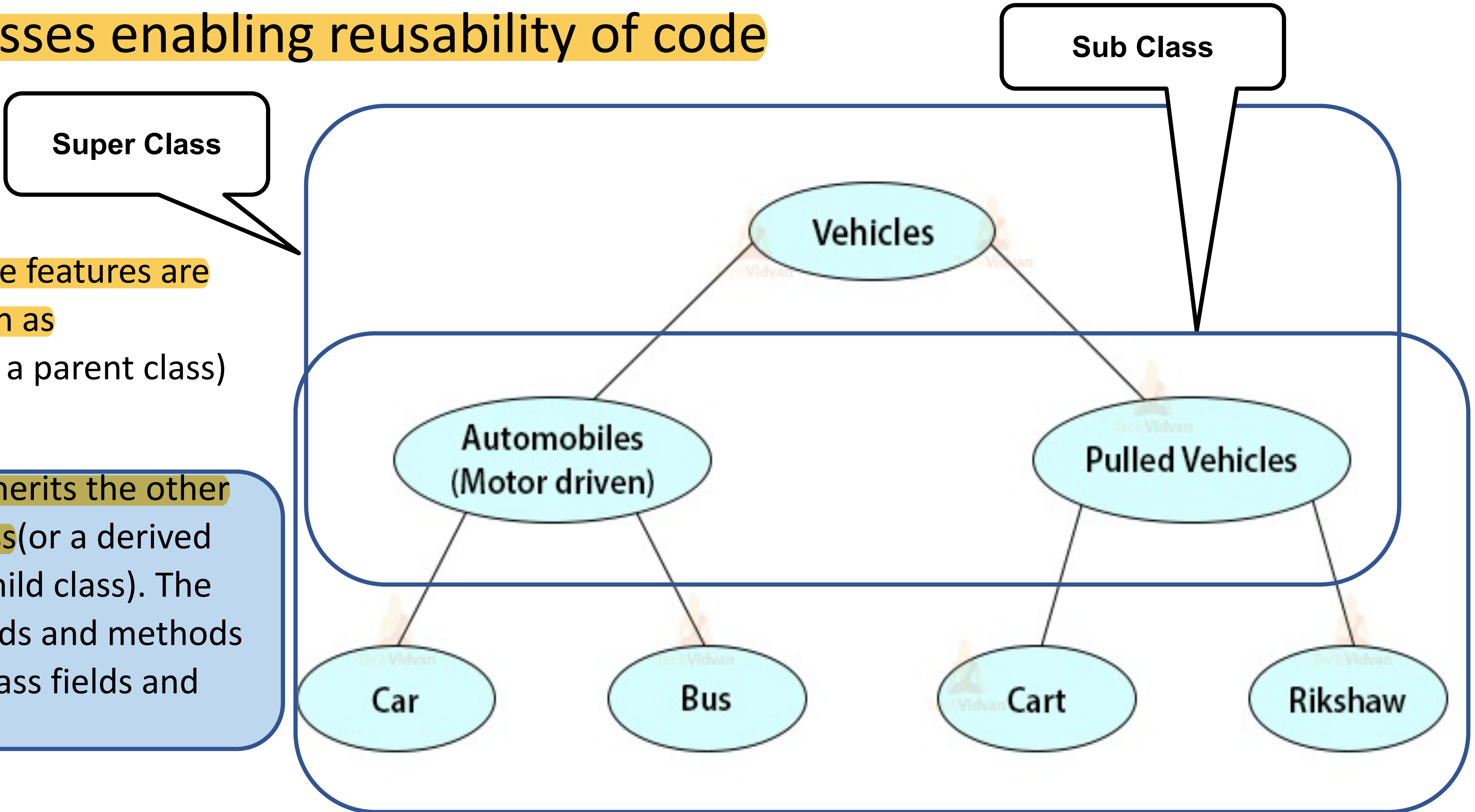
Dr. L. Kamatchi Priya
Department of Computer Science and Engineering

Inheritance

- Acquires the properties (data and methods) from one class to other classes enabling reusability of code

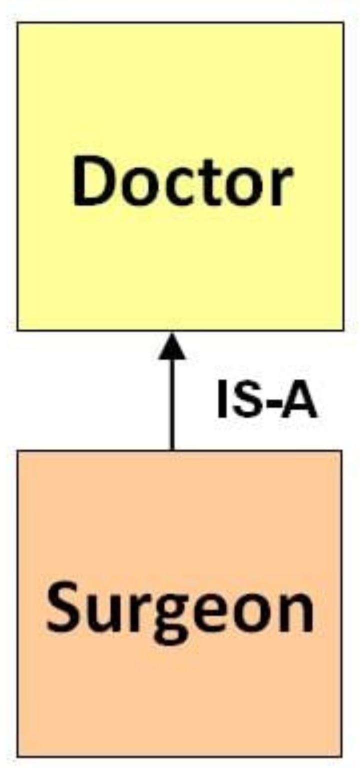
Super Class: The class whose features are inherited is known as superclass (or a base class or a parent class)

Sub Class: The class that inherits the other class is known as a subclass(or a derived class, extended class, or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.



Inheritance

- The child class is a specific type of the parent class.



Inheritance in Java

```
class Teacher {
    String designation = "Teacher";
    String collegeName = "PESU";
    void does(){
        System.out.println("Teaching");
    }
}
public class JavaTeacher extends Teacher{
    String mainSubject = "Java";
    public static void main(String args[]){
        JavaTeacher obj = new JavaTeacher();
        System.out.println(obj.collegeName);
        System.out.println(obj.designation);
        System.out.println(obj.mainSubject);
        obj.does();
    }
}
```

Cannot inherit from more than one Super Class

Pros and Cons…

Advantages:

- Reusability and saves time
- Enhances readability
- Overriding

With inheritance, we will be able to override the methods of the base class so that meaningful implementation of the base class method can be designed in the derived class.

Pros and Cons…

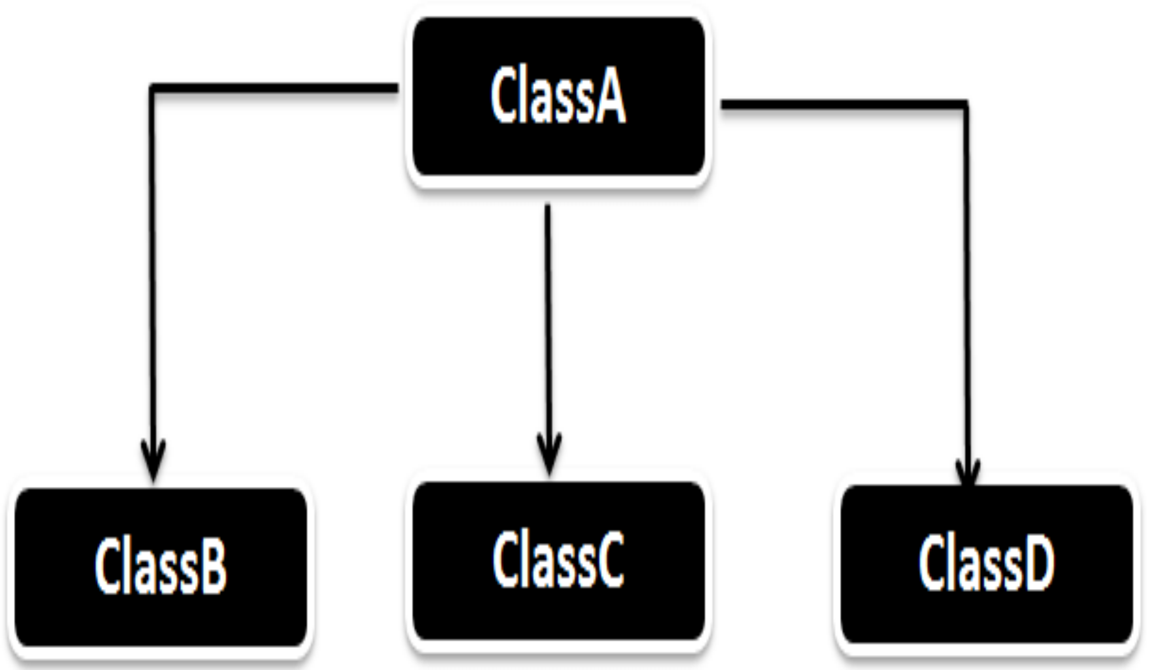
Disadvantages:

- Works slower
- Memory wastage
- Coupling

Inheritance increases the coupling between base class and derived class. A change in base class will affect all the child classes.

Types of Inheritance

- Single Inheritance
- Multilevel Inheritance
- Hierarchical Inheritance



Types of Inheritance

Single Inheritance:

```
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

Multilevel Inheritance

```
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();
   }
}
```

Types of Inheritance

Hierarchical Inheritance

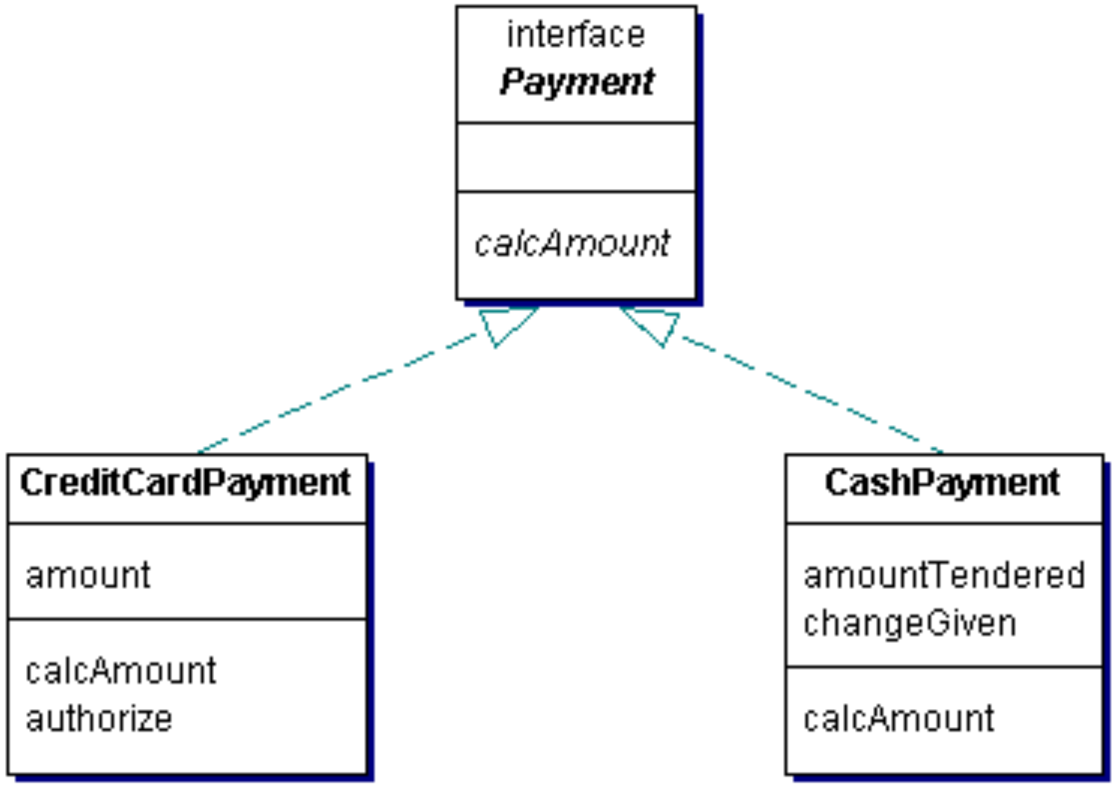
```
class Animal{
      void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
      void bark(){System.out.println("barking...");}
}
class Cat extends Animal{
      void meow(){System.out.println("meowing...");}
}
class TestInheritance3{
      public static void main(String args[]){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

Interface

- An abstract type used to specify the behaviour of a class.
- Contains static constants and abstract methods.
- Class can implement multiple interfaces.

```
interface Pet{
  public void showLove();
}
class Dog implements Pet{
   public void showLove(){
     System.out.println("Wag tail");
  }
}
class Cat implements Pet{
   public void showLove(){
     System.out.println(“Cuddle");
  }
}
```

Interface


interface
Payment
calcAmount
CreditCardPayment
amount
calcAmount
authorize
CashPayment
amountTendered
changeGiven
calcAmount

Why Interface

1. Used to achieve abstraction

2. Supports the functionality of Multiple inheritance

3. It can used to achieve loose coupling

Object Oriented Analysis and Design with Java

Inheritance Vs Interface

Category	Inheritance	Interface
1. Description	Inheritance is the mechanism in java by which one class is allowed to inherit the features of another class.	Interface is the blueprint of the class. It specifies what a class must do and not how. Like a class, an interface can have methods and variables, but the methods declared in an interface are by default abstract (only method signature, no body).
2. Use	It is used to get the features of another class.	It is used to provide total abstraction.
3.Syntax	class <subclass_name> extends <superclass_name> { }	interface <interface_name> { } class <class_name> implements <interface_name> {}
4. Number of Inheritance	It is used to provide the following types of inheritance - multi-level, single, hybrid and hierarchical inheritance)	It is used to provide multiple inheritance
5. Keywords	It uses ***extends*** keyword.	It uses ***implements*** keyword.

Object Oriented Analysis and Design with Java
Inheritance Vs Interface

Category	Inheritance	Interface
6. Inheritance	We can inherit lesser classes than Interface if we use Inheritance.	We can inherit enormously more classes than Inheritance, if we use Interface.
7. Method Definition	Methods can be defined inside the class in case of Inheritance.	Methods cannot be defined inside the class in case of Interface (except by using static and default keywords).
8. Overloading	It overloads the system if we try to extend a lot of classes.	System is not overloaded, no matter how many classes we implement.
9. Functionality Provided	It does not provide the functionality of loose coupling	It provides the functionality of loose coupling.
10. Multiple Inheritance	We cannot do multiple inheritance (causes compile time error).	We can do multiple inheritance using interfaces.

THANK YOU

Dr. L. Kamatchi Priya

Department of Computer Science and Engineering

priyal@pes.edu