
#18
POLYMORPHISM
ZOOMING

Polymorphism

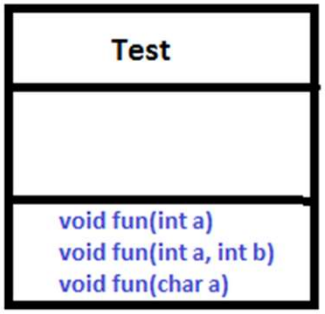
- As **Real life example of polymorphism:** A person at the same time can have different characteristic. Like a man at the same time is a father, a husband, an employee. So the same person posses different behaviour in different situations. This is called polymorphism.
- Polymorphism is considered as one of the important features of Object Oriented Programming.
- Polymorphism allows us to perform a single action in different ways. In other words, polymorphism allows you to define one interface and have multiple implementations.
- The word “poly” means many and “morphs” means forms, So it means many forms.

Polymorphism

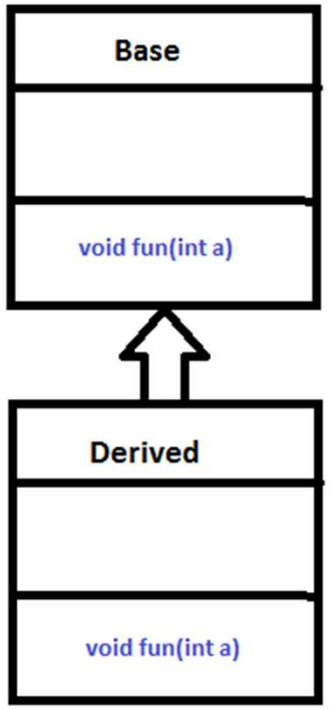
- In java polymorphism mainly divided into two types.
  - Compile time polymorphism
  - Runtime polymorphism

Compile time polymorphism

- It is also known as static polymorphism.
- It is achieved by function overloading or operator overloading.



Overloading



Overriding

Compile time polymorphism

Function overloading

- When there are multiple functions with same name but different parameters then these functions are said to be overloaded.
- Functions can be overloaded by change in number of arguments or/and change in type of arguments.

```
class MultiplyFun {
    static int Multiply(int a, int b , int c)
    {
        return a * b * c;
    }
    static double Multiply(double a, double b)
    {
        return a * b;
    }
}
class Main {
    public static void main(String[] args)
    {
        System.out.println(MultiplyFun.Multiply(2, 4 , 6));
        System.out.println(MultiplyFun.Multiply(5.5, 6.3));
    }
}
```

Compile time polymorphism

Operator overloading

- When there are multiple functions with same name but different parameters then these functions are said to be overloaded.
- Operator can be overloaded by change in type of arguments.

```
class MultiplyFun {
    static int Multiply(int a, int b)
    {
        return a + b;
    }
    static double Multiply(String a, String b)
    {
        return a + b;
    }
}
class Main {
    public static void main(String[] args)
    {
        System.out.println(MultiplyFun.Multiply(2, 4));
        System.out.println(MultiplyFun.Multiply('a', 'b'));
    }
}
```

Runtime polymorphism

- It is known as Dynamic Method Dispath.
- It is a process in which a function call to the overridden method is resolved at Runtime.
- This type of polymorphism is achieved by **Method Overriding**.

```
class Parent {
    void Print() {
        System.out.println("parent class");
    }
}
class subclass1 extends Parent {
    void Print() {
        System.out.println("subclass1");
    }
}
class subclass2 extends Parent {
    void Print() {
        System.out.println("subclass2");
    }
}
class TestPolymorphism3 {
    public static void main(String[] args) {
        Parent a;
        a = new subclass1();
        a.Print();
        a = new subclass2();
        a.Print();
    }
}
```