Saraswati Vandana

या कुन्देन्दु तुषार हार धवला
या शुभ्र वस्त्रान्विता ।
या वीणा वर दंड मंडितकरा
या श्वेत पद्मासना ॥

या ब्रह्मा अच्युत शंकर प्रभ्रतिभिः
देवै सदा पूजिता ।
सा मां पातु सरस्वती भगवती
निःश्येश जाड्यापह ॥

Faculty of Engineering & Technology
Sankalchand Patel College of Engineering, Visnagar

Java Programming
(1ET1030406)

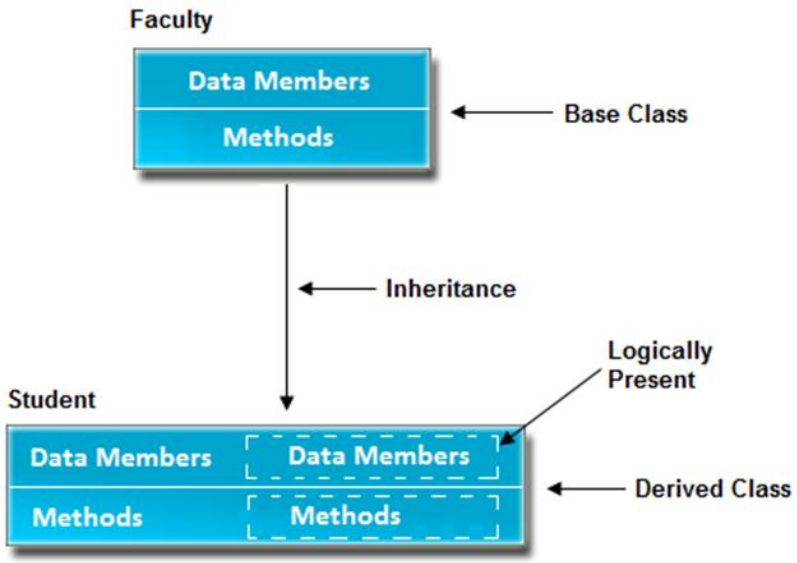
Unit-4 : Inheritance, Interface & Polymorphism

Prepared By
Mr. Mehul S. Patel
Department of Computer Engineering & Information Technology

Content

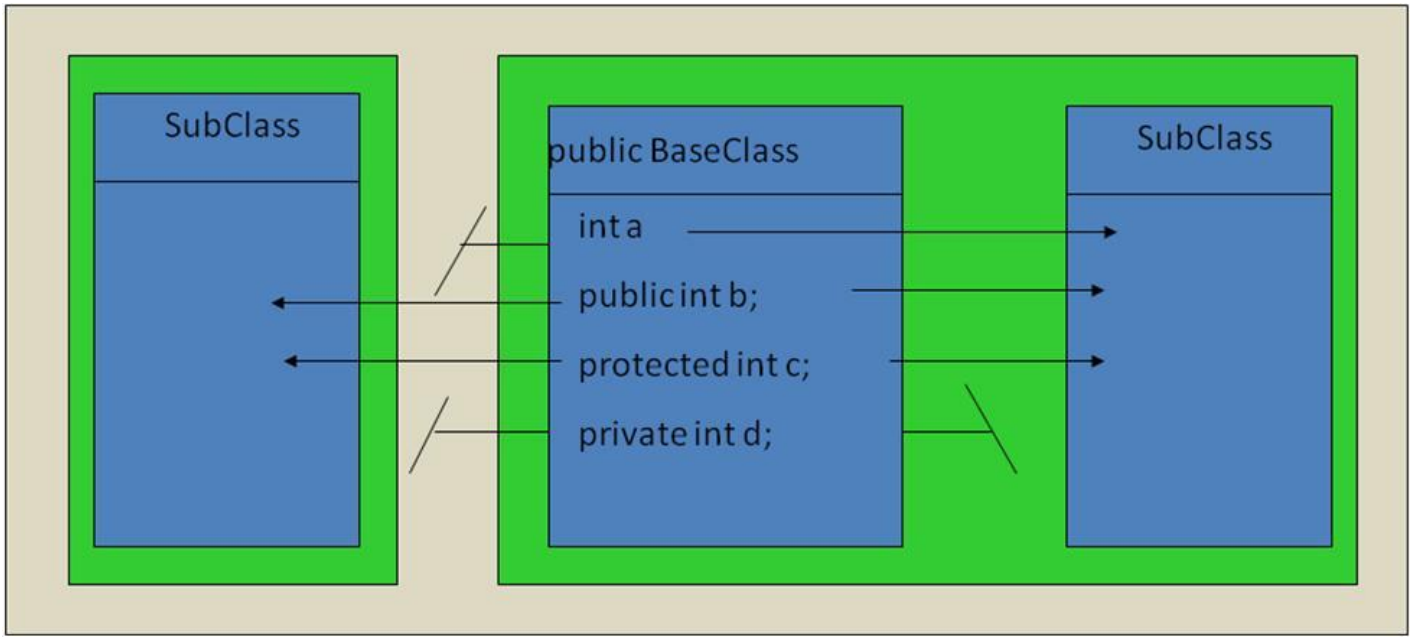
- Use of Inheritance
- Inheriting Data members and Methods
- Constructor in inheritance
  - Super keyword
- Types of Inheritance
- Method overriding
- Stop Inheritance
  - Final keywords
- Abstract Class
- Creation and Implementation of an interface
- Multiple Inheritance using Interface
- Interface Inheritance
- Dynamic method dispatch
- Instance of operator
- Understanding of Java Object Class
- Comparison between Abstract Class and interface

Use of Inheritance




```
class Faculty
{
    String name;
    String branch;
}
class Student extends Faculty
{
    String enrollmentNo;
    String sem;
}
```

Inheriting Data members and Methods



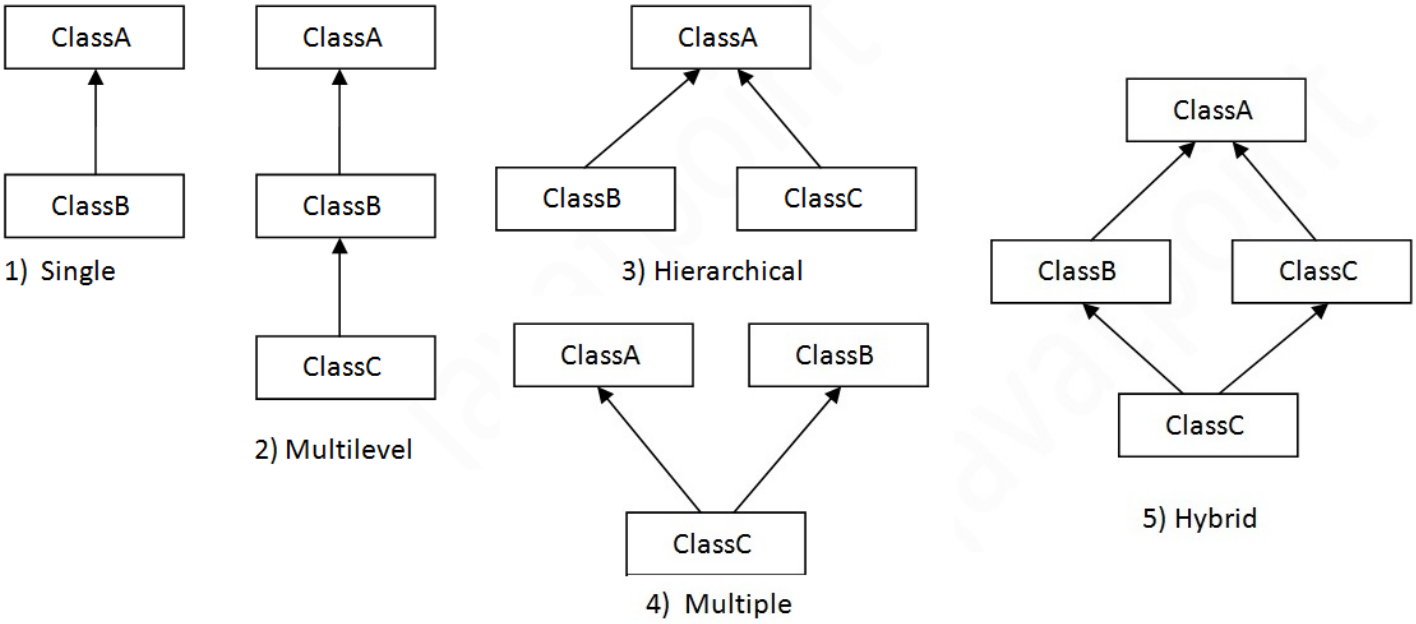
Constructor in inheritance

```java
class Person
{
    String name = "";
    int ssn = 0;
    Person(String name)
    {
        this.name = name;
    }
    Person(int ssn)
    {
        this.ssn = ssn;
    }
}
```

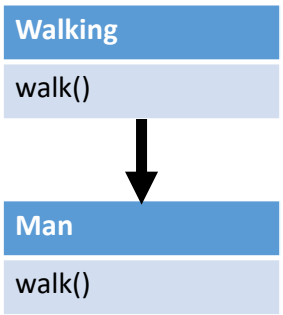
```java
class Doctor extends Person
{
    Doctor(String name)
    {
        super(name);
    }

    Doctor(int socialSecurityNumber)
    {
        super(socialSecurityNumber);
    }
}
```

```java
Doctor drwatson = new Doctor("Watson");
Doctor drpepper = new Doctor(823556789);
```

Types of Inheritance



Method overriding




```
Walking w=new Walking();
w.walk();
Man m = new Man();
m.walk();
```

```
class Walking
{
    void walk()
    {
    System.out.println("Fastly");
    }
}
class Man extends Walking
{
    @Override
    void walk()
    {
    System.out.println("Slowly");
    }
}
```

Final Keyword

- Stop Value Change
- Stop Method Overriding
- Stop Inheritance

Final Variable

```
class Demo
{
final int MAX_VALUE=99;
void myMethod()
      {
      MAX_VALUE=101;
      }
public static void main(String args[])
      {
      Demo obj=new Demo();
      obj.myMethod();
      }
}
```

Stop to change the value

Final Method

```
class First {
  final void display() {
    System.out.println("This is first class");
  }
}
class Second extends First {
  void display()    {
   System.out.println("This is second class");
  }
}
```

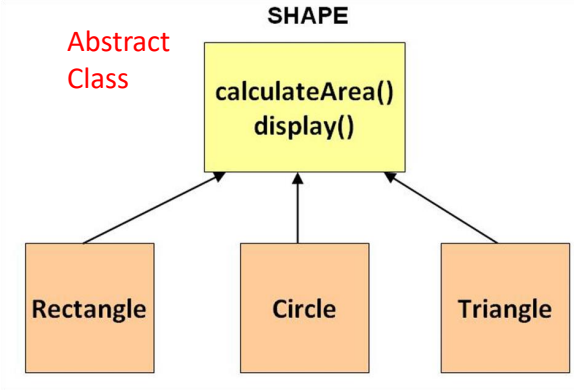
error **To stop method overriding**

Final class

```
final class Rafi
{
  void show()
  {
    System.out.println("Rafi on trustingeeks.com");
  }
}
public class Afridi extends Rafi
{
   public static void main(String[] args)
  {
    Afridi obj = new Afridi();
    obj.show();
  }
}
```

Error - To stop inheritance

Abstract Class



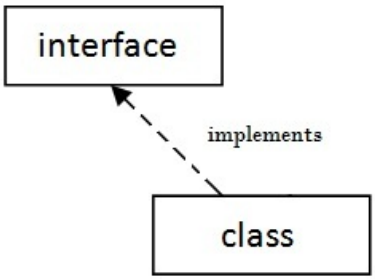

```
public abstract class Shape {
    private String color;

    public Shape(){}

    public String getColor() {
        return color;
    }
    public void setColor(String color) {
        this.color = color;
    }
    public abstract double getArea();
    public abstract double getPerimeter();
}
```

Creation and Implementation of an interface

```
interface Printable{
void print();
}
```

```
class A7 implements Printable {
  public void print(){
        System.out.println("Hello");
        }
}
```

```
Class DemoClass
{
public static void main(String args[]){

A7 obj = new A7();
obj.print();

 }
}
```



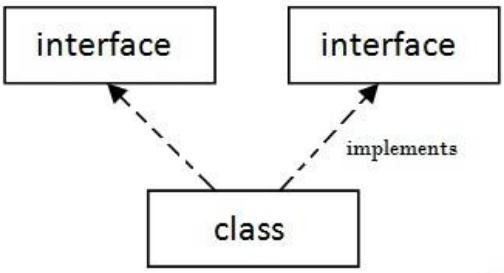
Multiple Inheritance using Interface

```
interface Printable{
void print();
}
```

```
interface  Showable{
  void show();
}
```

```
class A7 implements Printable,Showable{
  public void print(){
        System.out.println("Hello");
        }
public void show(){
        System.out.println("Welcome");
        }
}
```

```
Class DemoClass
{
public static void main(String args[]){

A7 obj = new A7();
obj.print();
obj.show();
 }
}
```



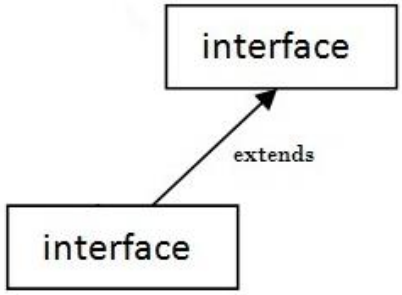
Interface inheritance

```
interface Printable{
void print();
}
```

```
interface Showable extends Printable
{ void show();
}
```

```
class A7 implements Showable{
  public void print(){
        System.out.println("Hello");
        }
public void show(){
        System.out.println("Welcome");
        }
}
```

```
Class DemoClass
{
public static void main(String args[]){

A7 obj = new A7();
obj.print();
obj.show();
 }
}
```



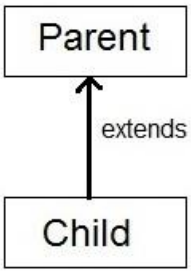
Dynamic Method Dispatch - Polymorphism

```
class Bike{
  void run(){
System.out.println("Bike");
}
}
```

```
class Splender extends Bike{
  void run(){
System.out.println("Splender");
        }
}
```

```
class DemoMain{
    public static void main(String args[]){
        Bike b = new Bike();
        b.run();
        b  =  new  Splender();//upcasting
        b.run();
  }
}
```

instance of Operator



Parent p = new Child( );

Upcasting

Child c = new Parent( );

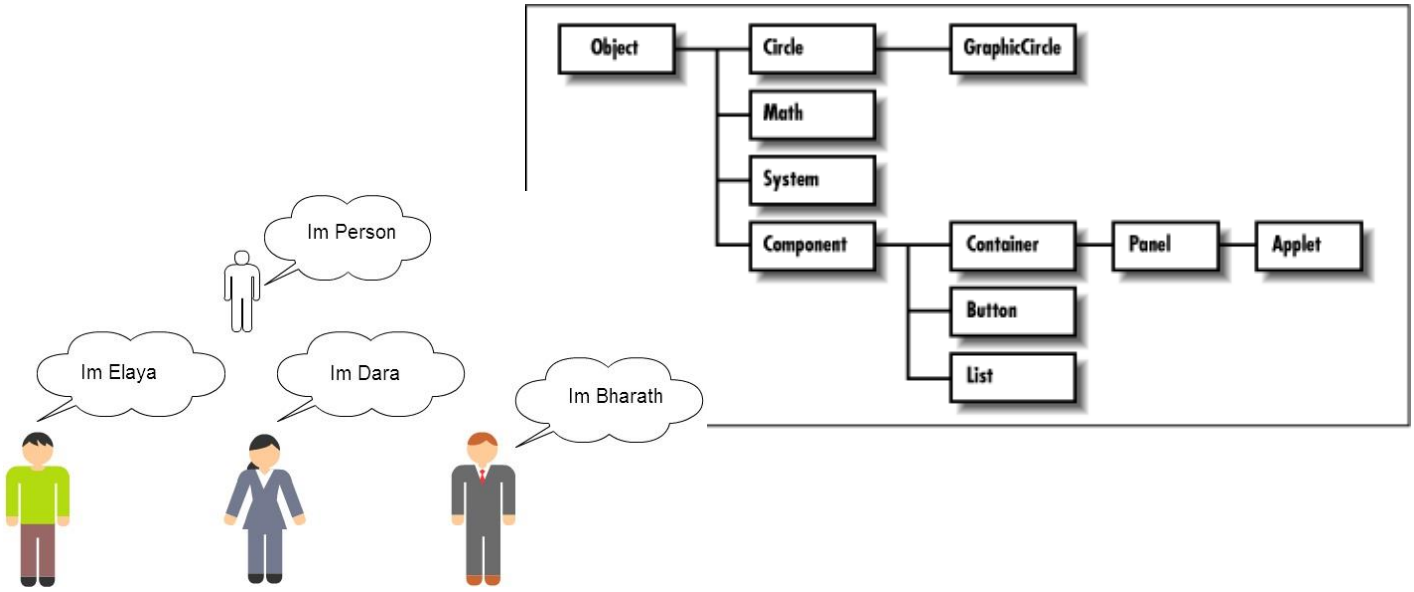
Compile time error

Child c = (Child) new Parent( );

Downcasting but throws ClassCastException at runtime.

```
if (p instanceof Child)
{
    Child c=p;
}
```

Understanding of Java Object Class



Abstract vs Interface

Feature	Interface	Abstract Class
Multiple Inheritance	Yes	No
Object Instantiated	No	No
Instance Member Function (Methods)	No	Yes
Instance Data Member	No	Yes
Inheritance	Implements	Extends
Access Modifier	Public	All
Constructor	No	Yes

References:

- http://www.write-technical.com/126581/session7/session7.htm
- http://sourcecodemania.com/inheritance-in-java/
- http://www.sitesbay.com/java/java-method-overloading

Questions/Comments


THANK
YOU