public interface DrawAPI {

public void drawCircle(int radius, int x, int y);

}

---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

public class RedCircle implements DrawAPI {

@Override

public void drawCircle(int radius, int x, int y) {

System.out.println("Drawing Circle[ color: red, radius: " + radius + ", x: " + x + ", " + y + "]");

}

}

public class GreenCircle implements DrawAPI {

@Override

public void drawCircle(int radius, int x, int y) {

System.out.println("Drawing Circle[ color: green, radius: " + radius + ", x: " + x + ", " + y + "]");

}

}

public abstract class Shape {

protected DrawAPI drawAPI;

protected Shape(DrawAPI drawAPI){

this.drawAPI = drawAPI;

}

public abstract void draw();

}

public class Circle extends Shape {

private int x, y, radius;

public Circle(int x, int y, int radius, DrawAPI drawAPI) {

super(drawAPI);

this.x = x;

this.y = y;

this.radius = radius;

}

public void draw() {

drawAPI.drawCircle(radius,x,y);

}

}

public class BridgePatternDemo {

public static void main(String[] args) {

Shape redCircle = new Circle(100,100, 10, new RedCircle());

Shape greenCircle = new Circle(100,100, 10, new GreenCircle());

redCircle.draw();

greenCircle.draw();

}

}

**Result**

Drawing Circle[ color: red, radius: 10, x: 100, 100]

Drawing Circle[ color: green, radius: 10, x: 100, 100]