# Adapter Pattern



# Code:

# Interface:

public interface IShapesColors {  
 void Colored(String nameOfShape);  
 void Border(String nameOfShape);  
}

public interface IShapes {  
 void draw();  
}

public class FactoryShapes {  
  
 public IShapes getShapes(String name){  
 IShapes NumberOfShapes;  
 if (name == "Circle"){  
 NumberOfShapes = new Circle();  
 }else if (name == "Triangle") {  
 NumberOfShapes = new Triangle();  
 }else if (name == "Square") {  
 NumberOfShapes = new Square();  
 }  
 else {  
 return null;  
 }  
 return NumberOfShapes;  
 }  
  
 public void setShapeStyle(String style, String Shape) {  
 if (style == "Border") {  
 (new BorderShapes()).Border(Shape);;  
 } else if (style == "Colored") {  
 (new ColoredShapes()).Colored(Shape);  
 } else {  
  
 }  
 }  
}  
  
class Circle implements IShapes{  
 @Override  
 public void draw() {  
 System.*out*.println("Circle is Draw");  
 }  
}  
class Square implements IShapes{  
 @Override  
 public void draw() { System.*out*.println("Square is Draw"); }  
}  
class Triangle implements IShapes{  
 @Override  
 public void draw() {  
 System.*out*.println("Triangle is Draw");  
 }  
}

public class BorderShapes implements IShapesColors {  
  
 @Override  
 public void Colored(String nameOfShape) {  
// do nothings  
 }  
  
 @Override  
 public void Border(String nameOfShape) {  
 System.*out*.println("Give border to the shape of "+nameOfShape);  
 }  
}

public class ColoredShapes implements IShapesColors {  
  
 @Override  
 public void Colored(String nameOfShape) {  
 System.*out*.println("Give Colored to the shape of "+nameOfShape);  
 }  
  
 @Override  
 public void Border(String nameOfShape) {  
 // do nothings  
 }  
}



