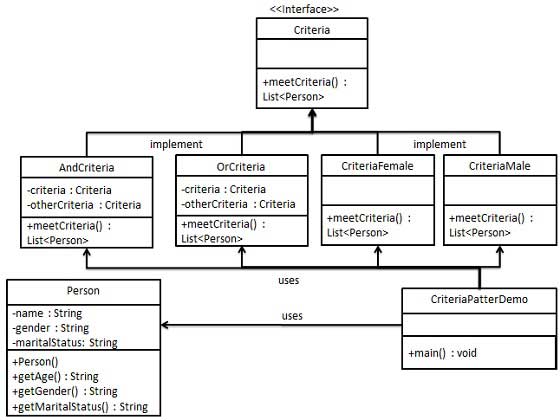
## 过滤器模式

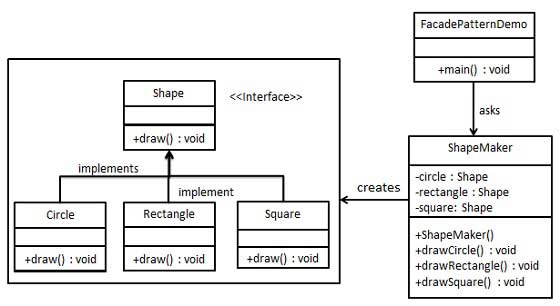
允许开发人员使用不同的标准来过滤一组对象，通过逻辑运算以解耦的方式把它们连接起来。



## 外观模式

向客户端提供一个可以访问系统的接口，从而隐藏系统的复杂性。

将一堆接口整合在一起的大杂脍



## 享元模式

主要用于减少创建对象的数量，以减少内存占用和提高性能。

在有大量对象时，有可能会造成内存溢出，我们把其中共同的部分抽象出来，如果有相同的业务请求，直接返回在内存中已有的对象，避免重新创建。

public class ShapeFactory {

private static final HashMap<String, Shape> circleMap = new HashMap<>();

public static Shape getCircle(String color) {

Circle circle = (Circle)circleMap.get(color);

if(circle == null) {

circle = new Circle(color);

circleMap.put(color, circle);

System.out.println("Creating circle of color : " + color);

}

return circle;

}

}

## 责任链模式

为请求创建了一个接收者对象的链。

在这种模式中，通常每个接收者都包含对另一个接收者的引用。如果一个对象不能处理该请求，那么它会把相同的请求传给下一个接收者，依此类推。

避免请求发送者与接收者耦合在一起，让多个对象都有可能接收请求，将这些对象连接成一条链，并且沿着这条链传递请求，直到有对象处理它为止。

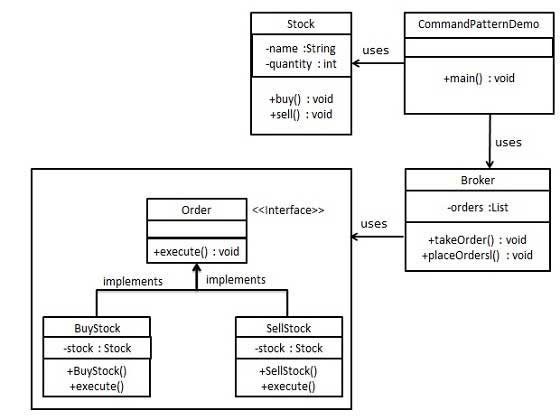
职责链上的处理者负责处理请求，客户只需要将请求发送到职责链上即可，无须关心请求的处理细节和请求的传递，所以职责链将请求的发送者和请求的处理者解耦了。

案例：Handler 里面聚合它自己，在 HandlerRequest 里判断是否合适，如果没达到条件则向下传递，向谁传递之前 set 进去。

命令模式

将一个请求封装成一个对象，从而使您可以用不同的请求对客户进行参数化。

定义三个角色：1、received 真正的命令执行对象 2、Command 3、invoker 使用命令对象的入口



解释器模式

给定一个语言，定义它的文法表示，并定义一个解释器，这个解释器使用该标识来解释语言中的句子。

1、可以将一个需要解释执行的语言中的句子表示为一个抽象语法树。 2、一些重复出现的问题可以用一种简单的语言来进行表达。 3、一个简单语法需要解释的场景。

参考：

《设计模式》菜鸟教程网站

《图解设计模式》【日】结城浩著