对象行为型模式

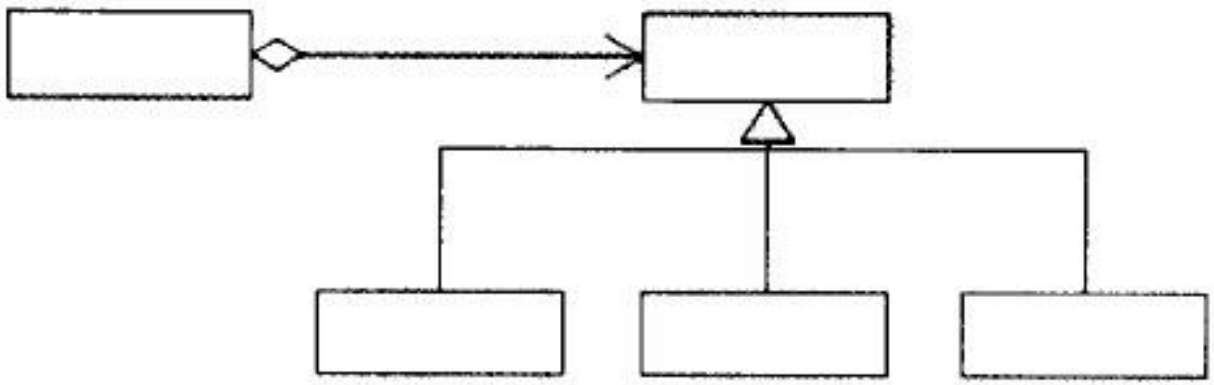
# 7、策略模式

STRATEGY—跟不同类型的MM约会，要用不同的策略，有的请电影比较好，有的则去吃小吃效果不错，有的去海边浪漫最合适，单目的都是为了得到MM的芳心，我的追MM锦囊中有好多Strategy哦。

策略模式：策略模式针对一组算法，将每一个算法封装到具有共同接口的独立的类中，从而使得它们可以相互替换。策略模式使得算法可以在不影响到客户端的情况下发生变化。策略模式把行为和环境分开。环境类负责维持和查询行为类，各种算法在具体的策略类中提供。由于算法和环境独立开来，算法的增减，修改都不会影响到环境和客户端。

# 什么是策略模式

- 策略模式是对算法的包装，是把使用算法的责任和算法本身分割开，委派给不同的对象管理。
- 策略模式通常把一个系统的算法包装到一系列的策略类里面，作为一个抽象策略类的子类。
- 策略模式的用意在于把可选的策略或方案封装在不同的类中，并在这些类中实现一个共同的操作。



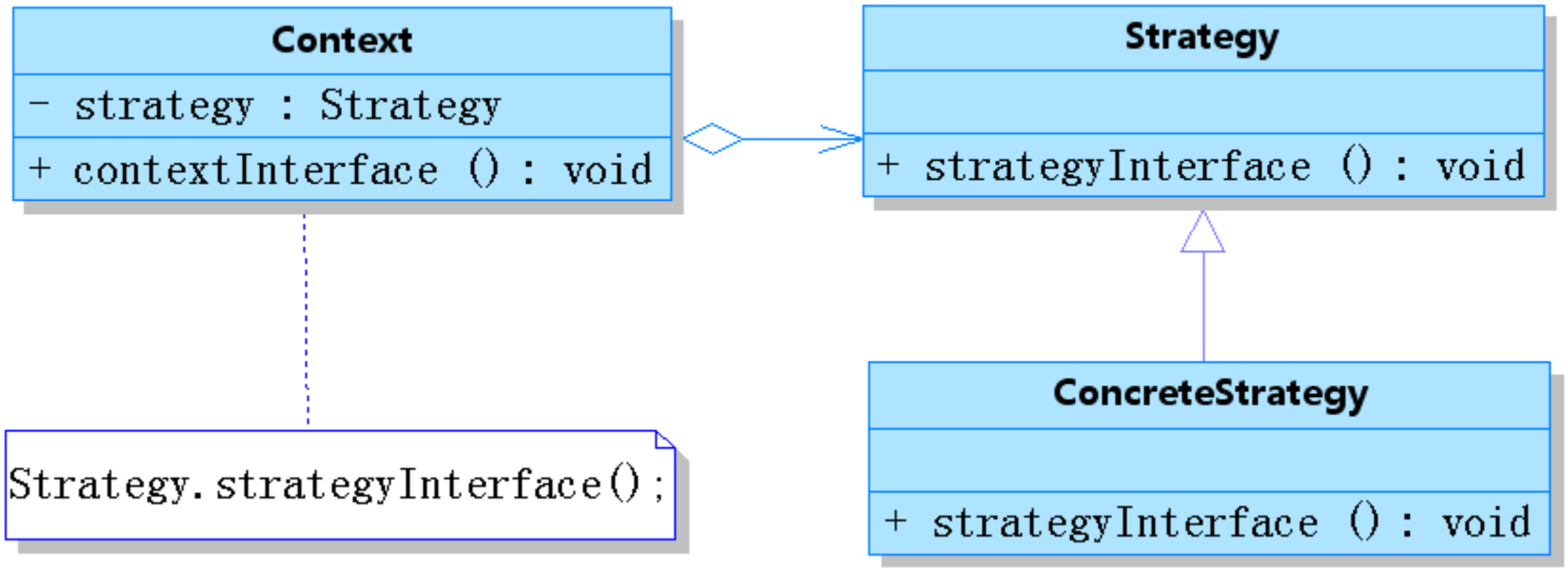
策略模式简略类图

# 策略模式的实现步骤

- 1、定义抽象角色类，定义好各个实现的共同抽象方法；
- 2、定义具体策略类，具体实现父类的共同方法；
- 3、定义环境角色类，私有化申明抽象角色变量，重载构造方法，执行抽象方法。

# 策略模式的结构

- 策略模式的类图：




这其中涉及到三个角色：

- 环境（Context）角色：持有一个Strategy类的引用。
- 抽象策略（Strategy）角色：这是一个抽象角色，通常由一个接口或抽象类实现。此角色给出所有的具体策略类所需的接口。
- 具体策略（ConcreteStrategy）角色：包装了相关的算法或行为。

# 使用场合

- 如果一系统里面有许多类，它们之间的区别仅在于它们的行为，那么使用策略模式可以动态地让一个对象在许多行为中选择一种行为。
- 一个系统需要动态地在几种算法中选择一种。那么这些算法可以包装到一个个的具体算法类里面，而这些具体算法类都是一个抽象算法类的子类。
- 一个系统的算法使用的数据不可以让客户端知道。策略模式可以避免让客户端涉及到不必要接触到的复杂的和只与算法有关的数据。
- 如果一个对象有很多种行为，如果不用恰当的模式，这些行为就只有使用多重的条件选择语句。

# 策略模式的优点

（1）策略模式提供了管理相关的算法族的方法。

定义了一系列算法，使这些算法可以相互替换，自由切换。

（2）策略模式提供了可以替换继承关系的方法。

（3）使用策略模式可以避免使用多重条件转移语句。

（4）更好的扩展性

# 策略模式的缺点

（1）所有的策略模式都需要对外暴露

客户端必须知道所有的策略类，并自行决定使用哪一个策略类。

（2）策略模式造成很多的策略类

（3）只适合扁平的算法结构，不适合于处理同时嵌套多于一个算法的情形。

装饰模式

享元模式

工厂方法模式
门面模式

# 策略模式与其他模式的关系

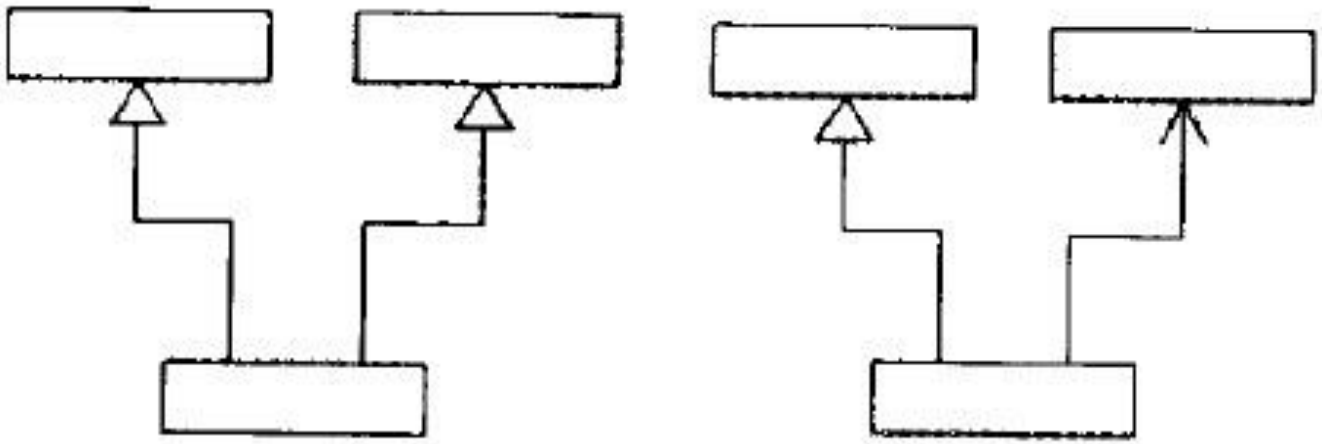
- 与建造者（Builder）模式的关系

二者结构上相似。事实上建造模式是策略模式的一种特殊情况。
这两种模式之间的区别是它们用意不同。
建造者模式中，核心功能是一步一步的方式创立一个产品，它的Director角色一遍一遍调用建造者对象来把零件增加到产品上，最后返还整个产品对象。
策略模式，Strategy角色在Context角色的调用下，不仅可以提供新对象的创立，还可以提供任何一种服务。

- 与享元（Flyweight Pattern）的关系

如果有多个客户端对象需要调用同样的一些策略类的话，就可以使它们实现享元模式，这样客户端可以共享这些策略类。可组合使用。

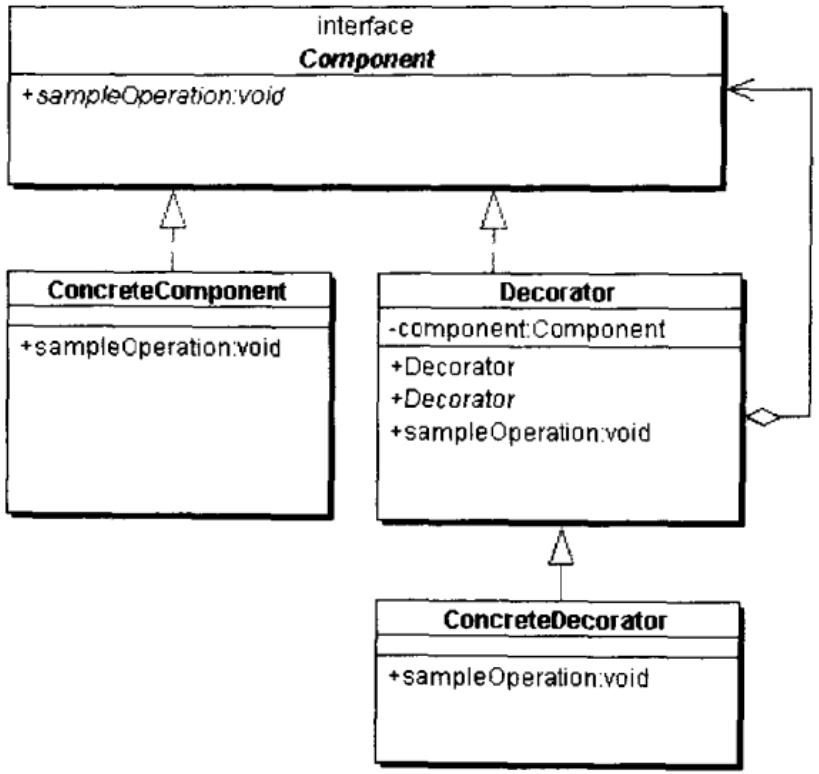
- 与适合器（Adapter）模式的关系



左边是类的适配器模式，右边是对象的适配器模式

二者在结构上相似。它们的区别在于它们的用意不同。
适配器模式的用意是允许一个客户对象通过调用一个配备着完全不同的接口的对象来完成它原来所要做的功能。
策略模式的用意是使用统一的接口为客户端提供不同的算法。

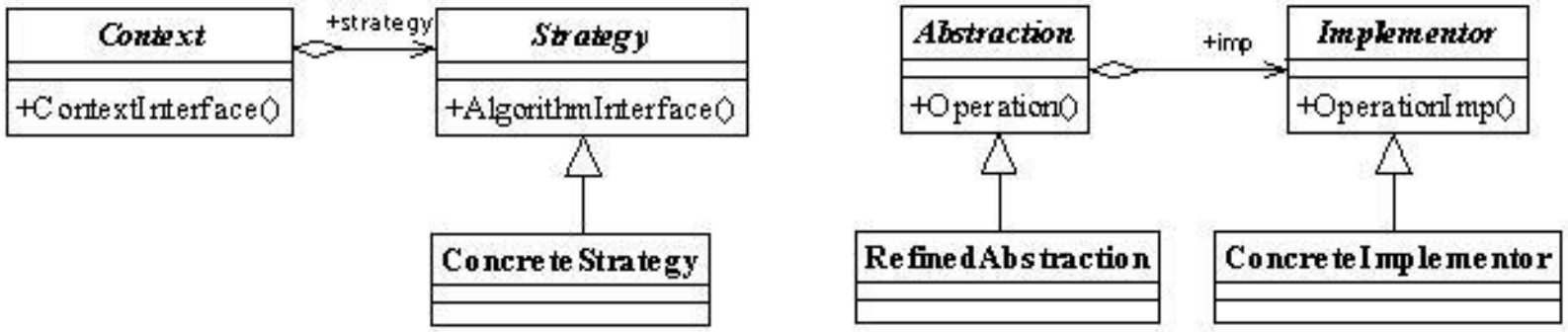
- 与装饰（Decorator）模式的关系




装饰模式类图

装饰模式的用意在于不改变接口的情况下，增强一个对象的功能。
策略模式在保持接口不变的情况下，使具体算法可以互换。策略模式只能处理客户端从几个具体算法中选择一个的情形，而不能处理客户端同时选用一种一上算法的情形。为处理后者，可以组合使用。

- 与桥梁（Bridge）模式的关系




两种模式结构图比较

策略模式是一个行为模式，旨在封装一系列的行为 （自由切换算法）

桥梁模式则是解决在不破坏封装的情况下如何抽取出它的抽象部分和实现部分（独立扩展）

- 与模板方法（Template）模式的关系

模板方法模式与策略模式的不同在于，策略模式使用委派的方法提供不同的算法行为，而模板方法模式使用继承的方法提供不同的算法行为。
可以组合使用，模板方法重在封装算法骨架，策略模式重在分离并封装算法的实现。

- 与状态（State）模式的关系

策略模式封装不同的算法，算法之间没有交互，以达到算法自由切换。
状态模式封装不同的状态，以达到状态切换导致行为发生改变的目的。