零声学院出品 Mark 老师 QQ: 2548898954

书籍

- 《设计模式-可复用面向对象软件的基础》
- 《重构与模式》

设计模式

> 设计模式是指在软件开发中，经过验证的，用于解决在特定环境下，重复出现的，特定问题的解决方案；

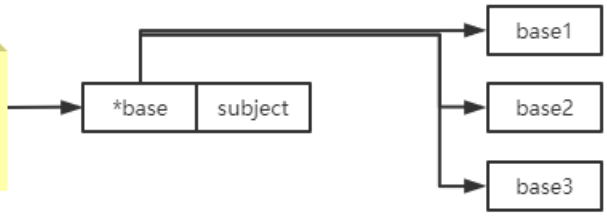
基础

类模型

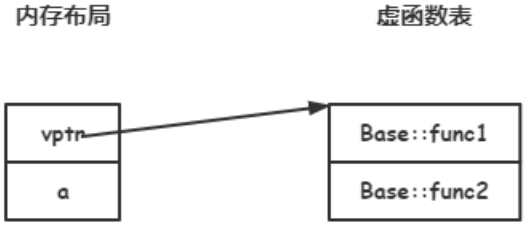
```
class Subject : public Base {

};
```

```
class Subject{
private:
    Base base;
};
```

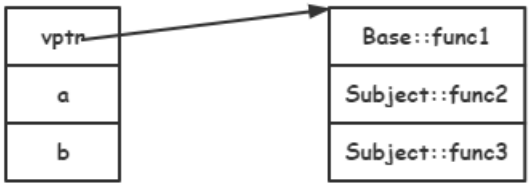

```
class Subject{
private:
    Base* base;
};
```




```
class Base{
public:
    virtual void func1(){}
    virtual void func2(){}
    int a;
};
```




```
class Subject : public Base {
public:
    virtual void func2(){}
    virtual void func3(){}
    int b;
};
```



类关系

```
class  Stranger {
};
```

```
class TonyFather {
    friend class TonyMother;
    friend class Beauty;
public:
    int a;
protected:
    int b;
private:
    int c;
};
```

```
class  TonyMother {
    friend class TonyFather;
private:
    int d;
};
```

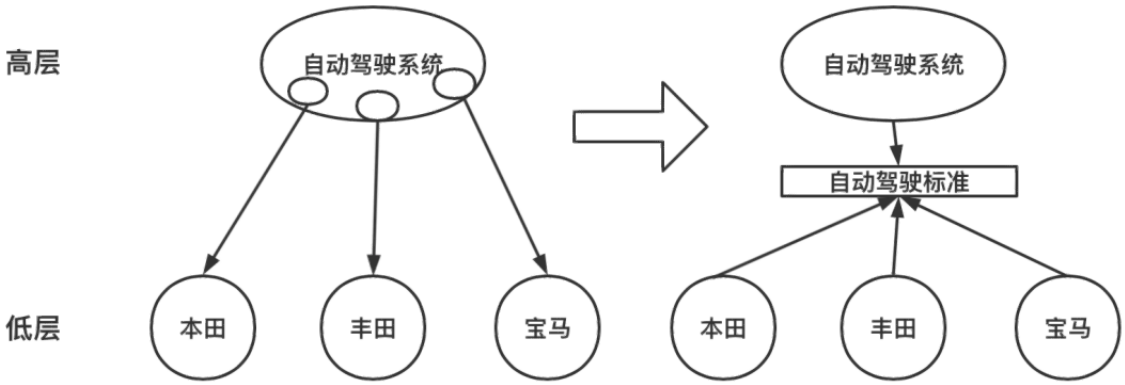
```
class  Beauty {
private:
    int e;
};
```

```
class Tony : public TonyFather
{
};
```

设计原则

依赖倒置

- 高层模块不应该依赖低层模块，两者都应该依赖抽象；
- 抽象不应该依赖具体实现，具体实现应该依赖于抽象；



- 自动驾驶系统公司是高层，汽车生产厂商为低层，它们不应该互相依赖，一方变动另一方也会跟着变动；而应该抽象一个自动驾驶行业标准，高层和低层都依赖它；这样以来就解耦了两方的变动；自动驾驶系统、汽车生产厂商都是具体实现，它们应该都依赖自动驾驶行业标准（抽象）；

开放封闭

- 一个类应该对扩展（组合和继承）开放，对修改关闭；

面向接口

- 不将变量类型声明为某个特定的具体类，而是声明为某个接口；
- 客户程序无需获知对象的具体类型，只需要知道对象所具有的接口；
- 减少系统中各部分的依赖关系，从而实现"高内聚、松耦合"的类型设计方案；

封装变化点

- 将稳定点和变化点分离，扩展修改变化点；让稳定点和变化点的实现层次分离；

单一职责

- 一个类应该仅有一个引起它变化的原因；

里氏替换

- 子类型必须能够替换掉它的父类型；主要出现在子类覆盖父类实现，原来使用父类型的程序可能出现错误；覆盖了父类方法却没有实现父类方法的职责；

接口隔离

- 不应该强迫客户依赖于它们不用的方法；
- 一般用于处理一个类拥有比较多的接口，而这些接口涉及到很多职责；

组合优于继承

- 继承耦合度高，组合耦合度低；

模板方法

定义

> 定义一个操作中的算法的骨架 ，而将一些步骤延迟到子类中。 Template Method使得子类可以不改变一个算法的结构即可重定义该算法的某些特定步骤。 ——《 设计模式》 GoF

背景

> 某个品牌动物园，有一套固定的表演流程，但是其中有若干个表演子流程可创新替换，以尝试迭代更新表演流程；

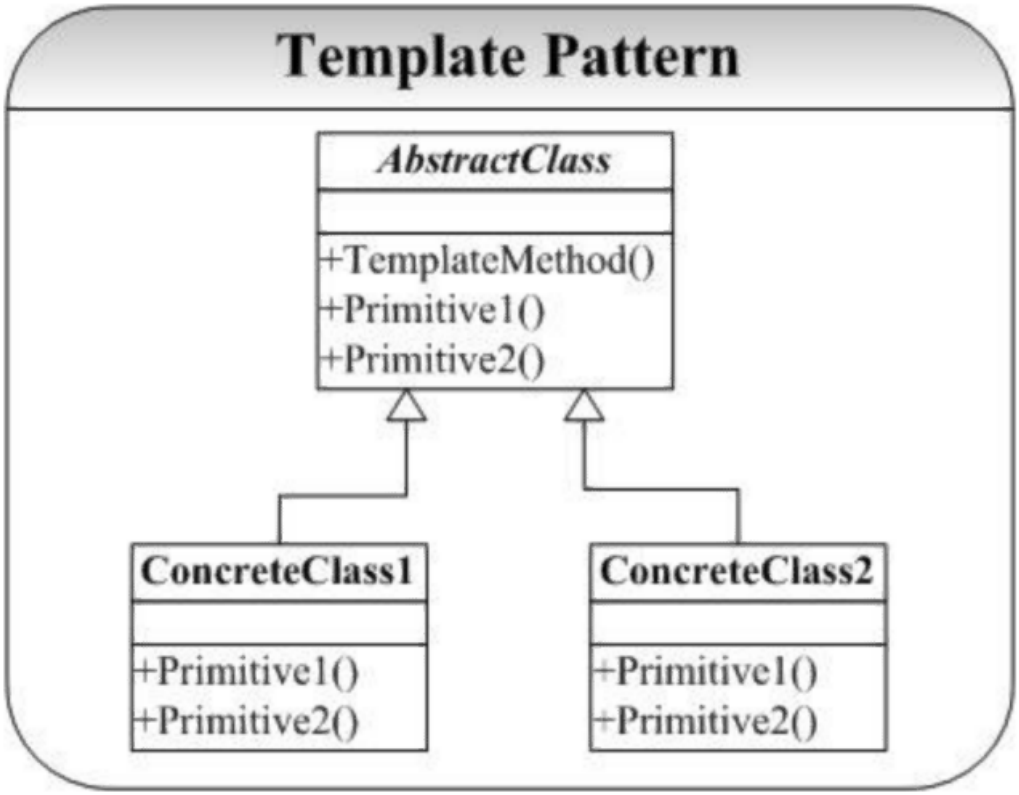
要点

- 最常用的设计模式，子类可以复写父类子流程，使父类的骨架流程丰富；
- 反向控制流程的典型应用；
- 父类 protected 保护子类需要复写的子流程；这样子类的子流程只能父类来调用；

本质

- 通过固定算法骨架来约束子类的行为；

结构图



观察者模式

定义

> 定义对象间的一种一对多（变化）的依赖关系，以便当一个对象(Subject)的状态发生改变时，所有依赖于它的对象都得到通知并自动更新。 ——《 设计模式》 GoF

背景

> 气象站发布气象资料给数据中心，数据中心经过处理，将气象信息更新到两个不同的显示终端（A和B）；

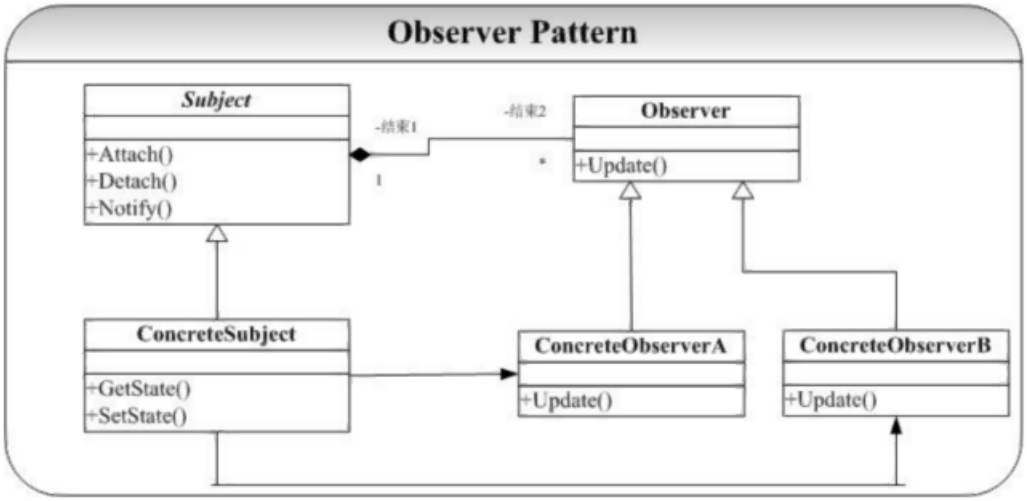
要点

- 观察者模式使得我们可以独立地改变目标与观察者，从而使二者之间的关系松耦合；
- 观察者自己决定是否订阅通知，目标对象并不关注谁订阅了；
- 观察者不要依赖通知顺序，目标对象也不知道通知顺序；
- 常用在基于事件的ui框架中，也是 MVC 的组成部分；
- 常用在分布式系统中、actor框架中；

本质

- 触发联动

结构图



策略模式

定义

> 定义一系列**算法**，把它们一个个封装起来，并且使它们可互相替换。该模式使得算法可独立于使用它的客户程序而变化。 ——《设计模式》 GoF

背景

> 某商场节假日有固定促销活动，为了加大促销力度，现提升国庆节促销活动规格；

要点

- 策略模式提供了一系列可重用的算法，从而可以使得类型在运行时方便地根据需要在各个算法之间进行切换；
- 策略模式消除了条件判断语句；也就是在解耦合；

本质

- 分离算法，选择实现；

结构图