**面向对象设计原则**：

* 1. 找出应用中可能需要变化之处，把它们独立出来，不要和那些不需要变化的代码混在一起。
     + 把变化的部分“封装”起来，好让其他部分不会受到影响；代码变化引起的不经意后果变少，系统变得更有弹性。
  2. 针对接口编程，针对超类型编程
  3. 多用组合，少用继承
  4. 类应该对修改关闭，对扩展开放，允许系统在不修改代码的情况下，进行功能扩展

1. **单一实例模式（Singleton Pattern）**

**保证一个类仅有一个实例存在，而且易于被访问。**

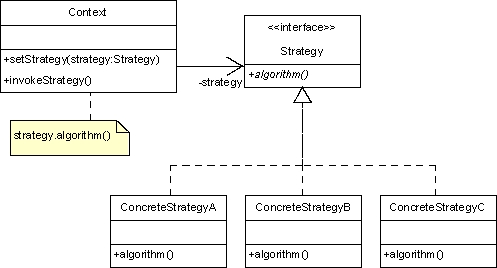


（1）构造函数私有；

（2）提供一个返回该类唯一实例的静态公开的方法；

（3）拥有一个该类类型的静态私有变量。

1. **策略模式（ Strategy Pattern）**

  
在该策略模式中，有一个公共的接口Strategy，该接口提供了一个策略算法的抽象方法，继承该接口的类ConcreteStrategyA、ConcreteStrategyB和ConcreteStrategyC分别封装了具体的策略算法，类Context是应用策略算法的类，它包含了一个Strategy接口类型的对象变量strategy，通过方法setStrategy为变量strategy赋予具体的类对象（即ConcreteStrategyA、ConcreteStrategyB或ConcreteStrategyC的对象）；方法invokeStrategy包含语句strategy.algorithm（）,通过多态实现对相应类的算法的调用。