《Head First 设计模式》：与设计模式相处

作者：jingqueyimu
原文链接：https://ld246.com/article/1604064251330
来源网站：链滴

正文

一、设计原则

1、封装变化

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。

2、针对接口编程，不针对实现编程

“针对接口编程”真正的意思是“针对超类型编程”。

超类型可以是抽象类或者接口，关键是要利用多态，这样程序执行时会根据实际状况执行到真正的行，不会被绑死在超类型的行为上。

“针对超类型编程”可以更明确地说成：变量的声明类型应该是超类型，如此，只要是具体实现此超型的类所产生的对象，都可以指定给这个变量。这也意味着，声明类时不用理会以后执行时的真正对类型。

3、多用组合，少用继承

“有一个”一般比“是一个”更好。换句话说，通过在类中持有其他类来获得新的行为一般会比通过承来获得新的行为好。

“组合”就是将多个类组合起来使用，即在一个类中持有其他类的引用。

4、为了交互对象之间的松耦合而努力

“松耦合”使对象之间的互相依赖降到了最低。如此一来，当一个对象发生改变时，对其他对象的影也降到了最低。

5、开闭原则：类应该对扩展开放，对修改关闭

应该让类在不修改现有代码的情况下，就可搭配新的行为。

6、依赖倒置原则：要依赖抽象，不要依赖具体类

不能让高层组件依赖低层组件，而且，不管高层或低层组件，都应该依赖于抽象。

所谓“高层”组件，是指由其他低层组件定义其行为的类。

避免违反依赖倒置原则的指导方针：

- 变量不可以持有具体类的引用。
- 不要让类派生自具体类。
- 不要覆盖基类中已实现的方法。

7、最少知识原则：减少对象之间的交互，只留下几个“密友”

不要让太多的类耦合在一起，免得修改系统中的一部分，会影响到其他部分。

为了避免违反最少知识原则，在对象的方法内，我们只应该调用属于以下范围的方法：

- 该对象本身。
- 被当作方法的参数而传递进来的对象。
- 此方法所创建或实例化的对象。
- 对象的组件，即被实例变量所引用的对象。

8、好莱坞原则：别调用（打电话给）我们，我们会调用（打电话给你

在好莱坞原则下，我们允许低层组件将自己挂钩到系统上，但是高层组件会决定什么时候和怎样使用些低层组件。

换句话说，高层组件对待低层组件的方式是“别调用我们，我们会调用你”。

9、单一责任原则：一个类应该只有一个引起变化的原因

我们知道要避免类内的改变，因为修改代码很容易造成许多潜在的错误。

如果一个类具有两个改变的原因，会使得将来该类的变化机率上升，而当它真的改变时，你的设计中时有两个方面将会受到影响。

二、定义设计模式

模式是在某情境下，针对某问题的某种解决方案。

- 情境：应用某个模式的情况。这应该是会不断出现的情况。
- 问题：你想在某情境下达到的目标，或者某情景下的约束。
- 解决方案：你所追求的一个通用的设计，它可以用来解决约束、达到目标。

三、设计模式分类

1、创建型模式

创建型模式涉及到将对象实例化，这类模式都提供一个方法，将客户从所需要实例化的对象中解耦。

- 工厂方法模式：由子类决定要创建的具体类是哪一个。
- 抽象工厂模式：允许客户创建对象的家族，而无需指定它们的具体类。
- 单件模式（单例模式）：确保有且只有一个对象被创建。
- 生成器模式（建造者模式）：封装一个复杂对象的创建过程。
- 原型模式：通过复制现有的实例来创建新的实例。

2、行为型模式

行为型模式涉及到类和对象如何交互及分配职责。

- 策略模式：封装可以互换的行为，并使用委托来决定要使用哪一个。
- 观察者模式：让对象能够在状态改变时被通知。
- 命令模式：封装请求成为对象。
- 模板方法模式：由子类决定如何实现一个算法中的步骤。
- 迭代器模式：在对象的集合之中游走，而不暴露集合的实现。
- 状态模式：封装了基于状态的行为，并使用委托在行为之间切换。
- 责任链模式：为某个请求创建一个对象链。
- 解释器模式：将每一个语法规则表示成一个类。
- 中介者模式：封装一系列对象之间的交互。
- 备忘录模式：在对象外部存储对象的某个状态。
- 访问者模式：通过访问数据结构中的每个元素，来对元素进行各种操作。

3、结构型模式

结构型模式涉及到类和对象如何被组合以建立新的结构或新的功能。

- 装饰者模式：包装一个对象，以提供新的行为。
- 适配器模式：封装对象，并提供不同的接口。
- 外观模式：简化一群类的接口。
- 组合模式：客户用一致的方式处理对象集合和单个对象。
- 代理模式：包装对象，以控制对此对象的访问。

- 桥接模式：分离抽象与实现，使它们可以独立变化。
- 蝇量模式（享元模式）：运用共享技术，减少对象的创建。

相关文章

《Head First 设计模式》：策略模式
《Head First 设计模式》：观察者模式
《Head First 设计模式》：装饰者模式
《Head First 设计模式》：工厂方法模式
《Head First 设计模式》：抽象工厂模式
《Head First 设计模式》：单件模式
《Head First 设计模式》：命令模式
《Head First 设计模式》：适配器模式
《Head First 设计模式》：外观模式
《Head First 设计模式》：模板方法模式
《Head First 设计模式》：迭代器模式
《Head First 设计模式》：组合模式
《Head First 设计模式》：状态模式
《Head First 设计模式》：代理模式
《Head First 设计模式》：剩下的模式