设计模式详解

引言

什么是设计模式?

> 什么是设计模式，是一套被反复使用，多数人知晓的，经过分类编目的、代码设计经验的总结。使用设计模式是为了可重用代码，让代码更加容易被人理解、保证代码可靠性、程序的重用性。

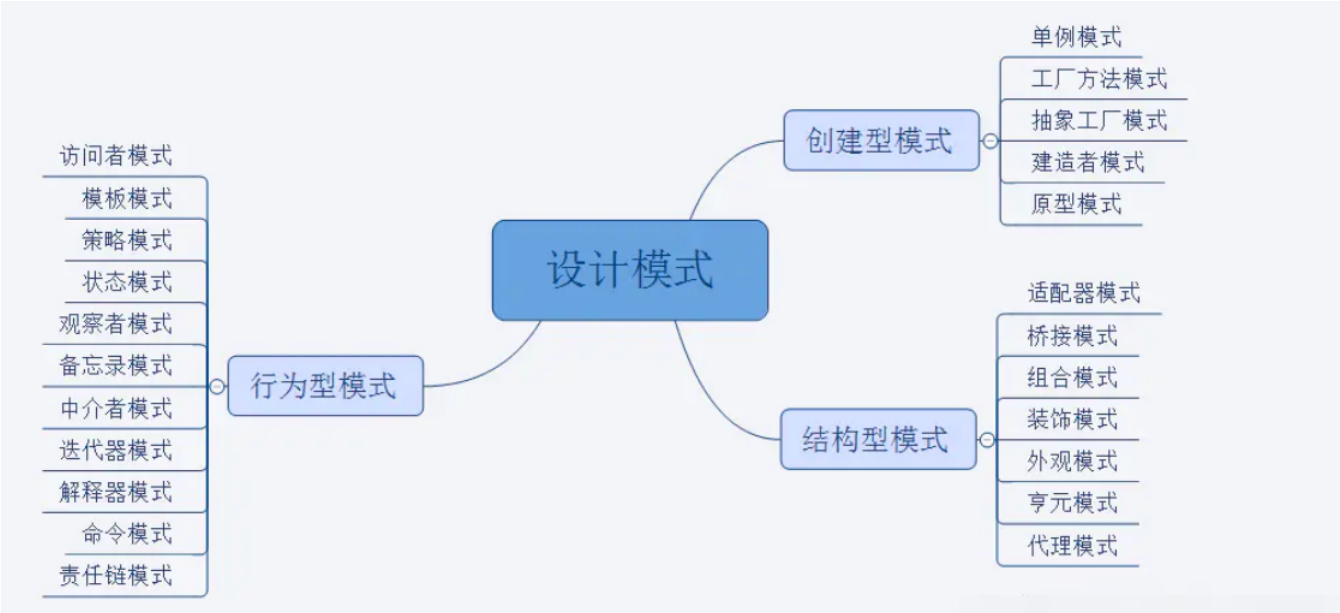
为什么学习设计模式?

1、更好的看懂源代码

2、看看前辈们的代码（去公司学习时看的代码）

3、编写属于自己的代码，别人看更加清楚易懂

设计模式分类



设计模式六大原则

1、开放封闭原则

2、里氏替换原则

3、依赖倒置原则

4、接口隔离原则

5、迪米特法则（最少知道原则）

6、单一职责原则

单例模式

什么是单例模式

> 保证一个类只有一个实例，并且提供一个访问该全局访问点

哪些地方用到单例模式

1、网站的计数器

2、应用程序的日志应用

3、Windows的任务管理器就是典型的单例模式，不能打开两个

4、Windows的回收站，在整个系统运行过程，回收站只维护一个实例。

单例优缺点

> 优点

1、在单例模式中，活动的单例只有一个实例，对单例类的所有实例化得到的都是相同的一个实例。这 样就防止其它对象对自己的实例化，确保所有的对象都访问一个实例

2、单例模式具有一定的伸缩性，类自己来控制实例化进程，类就在改变实例化进程上有相应的伸缩 性

3、避免对共享资源的多重占用

4、由于在系统内存中只存在一个对象，因此可以节约系统资源，当需要频繁创建和销毁的对象时单例 模式无疑可以提高系统的性能

> 缺点

1、不适用于变化的对象，如果同一类型的对象总是要在不同的用例场景发生变化，单例就会引起数据 的错误，不能保存彼此的状态

2、由于单利模式中没有抽象层，因此单例类的扩展有很大的困难

单例模式创建（懒汉式和饿汉式）

1、饿汉式:类初始化时,会立即加载该对象，线程天生安全,调用效率高。

2、懒汉式: 类初始化时,不会初始化该对象,真正需要使用的时候才会创建该对象,具备懒加载功能。

3、静态内部方式:结合了懒汉式和饿汉式各自的优点，真正需要对象的时候才会加载，加载类是线程 安全的。

4、枚举单例: 使用枚举实现单例模式 优点:实现简单、调用效率高，枚举本身就是单例

懒汉式

饿汉式

工厂方法模式

> 定义:

定义一个用于创建对象的接口，让子类决定实例化哪一个类。 Factory Method使得一个类的实例化 延迟到子类

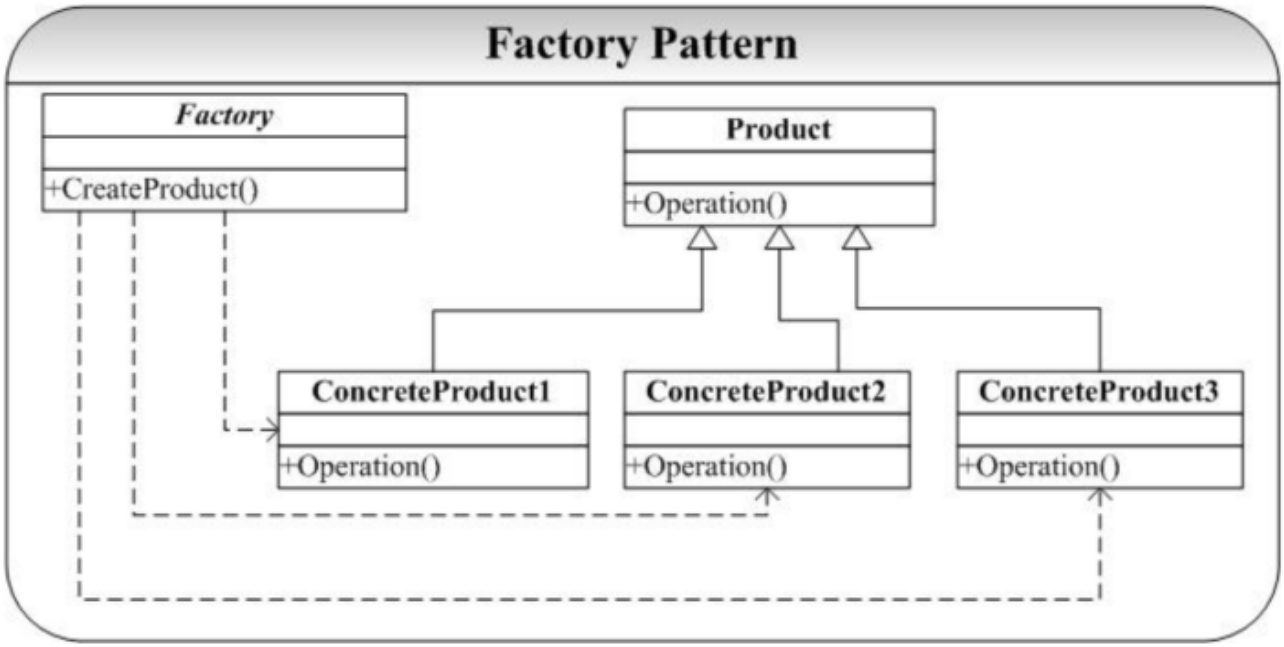
> 实例

实现一个导出数据的接口，让客户选择数据的导出方式.

> 本质

延迟子类选择实现

> 结构图



抽象工厂模式

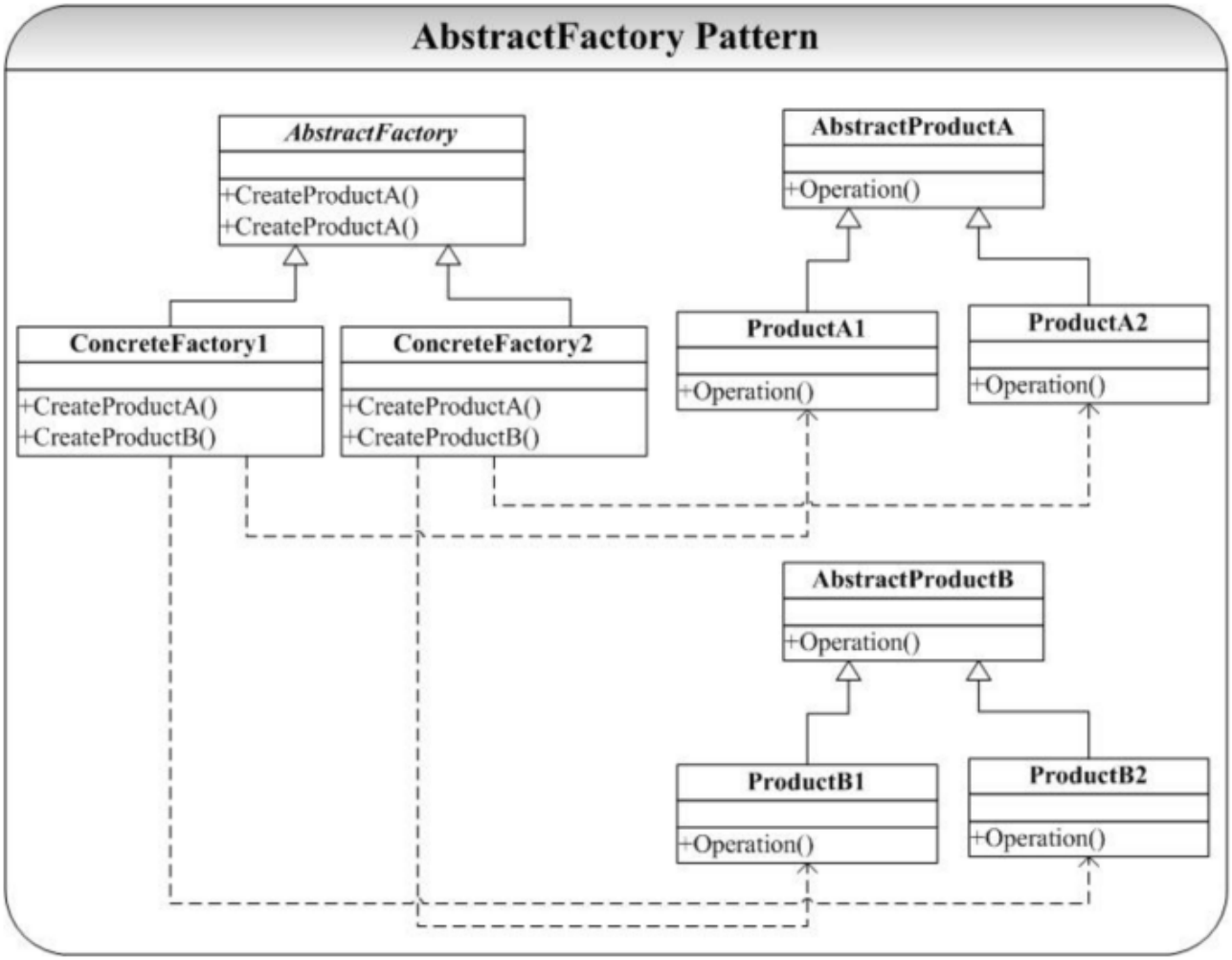
> 定义：

提供一个接口，让该接口负责创建一系列“相关或者相互依赖的对象”，无需指定它们具体的类

> 实例

实现一个拥有导出导入数据的接口，让客户选择数据的导出导入方式 （更加抽象了，客户只要能导入导出就行了，不管你怎么实现，工厂方法是具体的实现导出的方式是哪种）

> 结构图



装饰器模式

> 定义:

动态地给一个对象增加一些额外的职责。就增加功能而言，装饰器模式比生成子类更为灵活

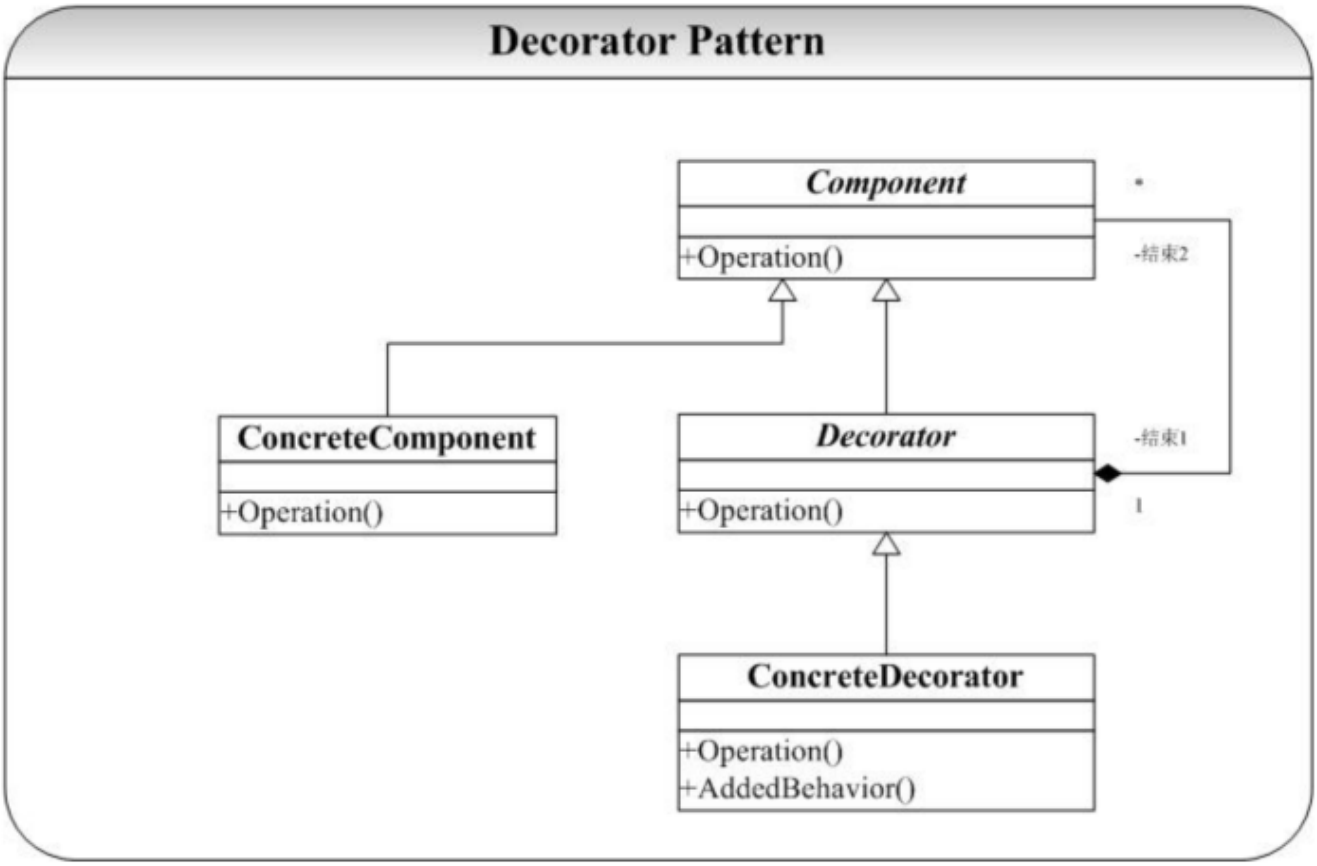
> 实例

普通员工有销售奖金，累计奖金，部门经理除此之外还有团队奖金；后面可能会添加环比增长奖 金，同时可能针对不同的职位产生不同的奖金组合 （让对象能有更多功能）

> 本质

动态组合

> 结构图



代理模式

> 定义

为其他对象提供一种代理以控制对这对象的访问

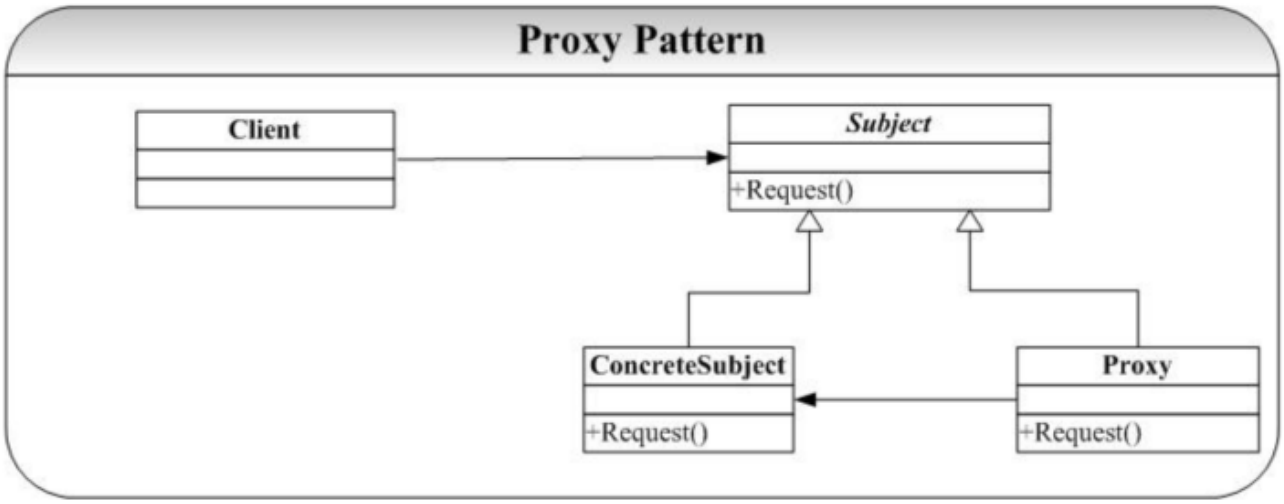
> 实例

在有些系统中，为了某些对象的纯粹性，只进行了功能相关封装（稳定点），后期添加了其他功能 需要对该对象进行额外操作（变化点），为了隔离变化点（也就是不直接在稳定点进行修改，这样 会让稳定点也变得不稳定），可以抽象一层代理层

> 本质

控制对象访问

> 结构图



适配器模式

> 定义：

将一个类的接口转换成客户希望的另一个接口。 Adapter模式使得原本由于接口不兼容而不能一起 工作的那些类可以一起工作

> 实例

日志系统，原来是通过写磁盘的方式进行存储，后来因为查询不便，需要额外添加往数据库写日志 的功能（写文件和数据库并存）（适配适配，让新的方式适配旧的接口）

> 本质

转换匹配，复用功能

> 结构图