|  |  |
| --- | --- |
| 工厂方法/抽象工厂 | 我出钱，你给货 ，我不用管你是怎么做的；工厂方法：只能获取单个；抽象工厂：是获取一套，一整组对象。 |
| 单例模式 | 独一无二：整个系统只有这样的一个对象。 |
| 原型模式 | 与单例刚好相反，可以有无数个复制品 |
| 委派模式 | 领导不干活，把活都分给小弟做， |
| 策略模式 | 选择权交给客户：领导不管你怎么上班：走路，坐车，骑车。。  只要你上班就行。 |
| 装饰模式 | 装饰：是为了让被装饰的对象有更多职责，而且装饰完后还是原来的类型。  例子：给人添加不同的角色，但追后他还是一个人。不能变成宁外的东西。 |
| 适配器模式 | 把两个不兼容对象进行组合到一起，可以一起完成工作，电源适配 |
| 代理模式 | 由专人代替自己去做某件事，和委派不同的是，代理会在做这这件事之前或之后 会做一些方法增强 |
| 观察者模式 | 一个对象发生变化后，另外一个人会做出相应的改变， |

Spring Aop : 事务，BeanFactoryTransactionAttributeSourceAdvisor

TransactionAttributeSourceAdvisor

Spring ioc：@bean 或者 @service @controller,@component..

**protected** Object doCreateBean(**final** String beanName, **final** RootBeanDefinition mbd, **final** @Nullable Object[] args)  
 **throws** BeanCreationException {}

**创建 对象 有spring 来完成**

**Spring di : @**Autowire @Resource(name=”xxx”),

**private** Optional<?> createOptionalDependency(  
 DependencyDescriptor descriptor, @Nullable String beanName, **final** Object... args) {