Spring中的设计模式

1. 代理模式

通过代理对象来访问目标对象，同时代理对象可以在不影响目标对象的情况下，对功能进行自己的扩展。

静态代理：代理对象与目标对象实现相同的接口，比如两者都有request()方法，然后通过持有一个目标对象A，在调用request()方法时，进行一系列扩展的操作后，再调用A.request()，来完成用户的操作，这样就可以在目标对象A无知觉的情况下，扩展功能。

\*但是，静态代理的缺点在于，要同时维护代理类和目标类两份代码，而且随着接口的变多，代码非常复杂，难以管理。

动态代理：使用jdk的反射功能和Java.reflect.Proxy类库，在运行时动态地构建出代理类，完成代理操作。使用proxy类库提供的静态方法，Proxy.newInstance方法，传入类加载器、对应的接口、需要扩展的目标类的方法等，动态地生成代理类。

\*静态代理和动态代理都需要目标类实现过一个接口才能使用，对于单独的，没有实现接口的类，就需要使用CGLib的方法创建代理类。