设计模式

适配器模式

为了解决我们不兼容的问题，把一个类的接口换成我们想要的接口。

应用场景：

比如多数据来源，且数据格式不同，可以封装适配器，将这些不同格式的数据进行格式统一

发布-订阅模式

消息的发布者，不会将消息直接发送给特定的订阅者，而是通过消息通道广播出去，然后呢，订阅者通过订阅获取到想要的消息。

在异步编程中实现更深的解耦

一个发布者对应多个订阅者

应用场景：

- 各模块相互独立
- 存在一对多的依赖关系
- 依赖模块不稳定、依赖关系不稳定
- 各模块由不同的人员、团队开发

中介者模式

对象和对象之间借助第三方中介者进行通信

参考资料

https://juejin.cn/post/6844904138707337229#heading-16

工厂模式

工厂模式定义一个用于创建对象的接口，这个接口由子类决定实例化哪一个类。该模式使一个类的实例化延迟到了子类。而子类可以重写接口方法以便创建的时候指定自己的对象类型。

https://juejin.cn/post/6844904032826294286#heading-5

单例模式

一个类只能构造出唯一实例

比如弹框。只创建一个弹框，不需要时候隐藏，需要时候显示。这样就不用每次要用弹框的时候都新建了

https://github.com/MuYunyun/blog/blob/main/BasicSkill/%E8%AE%BE%E8%AE%A1%E6%A8%A1%E5%BC%8F/%E5%8D%95%E4%BE%8B%E6%A8%A1%E5%BC%8F.md

策略模式

根据不同参数可以命中不同的策略

比如年终奖的需求，不同的绩效对应不同的年终奖。利用高阶函数实现策略模式。减少大量的if判断

https://github.com/MuYunyun/blog/blob/main/BasicSkill/%E8%AE%BE%E8%AE%A1%E6%A8%A1%E5%BC%8F/%E7%AD%96%E7%95%A5%E6%A8%A1%E5%BC%8F.md

代理模式

代理对象和本体对象具有一致的接口，对使用者友好

比如图片懒加载。代理对象先加载loading图片，等到图片加载完成后，代理对象调用本体对象的同样的方法来更换图片地址。也可以不使用代理对象，直接使用本体对象。

https://github.com/MuYunyun/blog/blob/main/BasicSkill/%E8%AE%BE%E8%AE%A1%E6%A8%A1%E5%BC%8F/%E4%BB%A3%E7%90%86%E6%A8%A1%E5%BC%8F.md