## 简介

将原本不兼容的接口能够合作无间。

## 角色

1. 对象适配器模式

1、 Target1接口。功能1的接口

2、 Target1实现类。具有功能1的target1接口的具体实现

3、 adapter类。实现了target1接口，并封装了target2接口作为属性的适配器类。

4、 target2接口。功能2的接口

5、 target2实现类。具有功能2的target2接口的具体实现。

1. 类适配器模式

1、 Target1接口。功能1的接口

2、 Target1实现类。具有功能1的target1接口的具体实现

3、 target2接口。功能2的接口

4、 target2实现类。具有功能2的target1接口的具体实现，并继承target1的实现类。作为适配器类。

## 优缺点

优点：

1、将目标类和适配者类解耦

2、增加了类的透明性和复用性，将具体的实现封装在适配者类中，对于客户端类来说是透明的，而且提高了适配者的复用性

3、灵活性和扩展性都非常好，符合开闭原则

类适配器还有的优点：

1、由于适配器类是适配者类的子类，因此可以再适配器类中置换一些适配者的方法，使得适配器的灵活性更强。

类适配器的缺点：

1、对于[**Java**](http://lib.csdn.net/base/java)、C#等不支持多重继承的语言，一次最多只能适配一个适配者类，而且目标抽象类只能为接口，不能为类，其使用有一定的局限性，不能将一个适配者类和他的子类同时适配到目标接口。

对象适配器还有的优点：

1、把多个不同的适配者适配到同一个目标，也就是说，同一个适配器可以把适配者类和他的子类都适配到目标接口。

对象适配器的缺点：

1、与类适配器模式相比，要想置换适配者类的方法就不容易

## 适用场景

1. 已经存在的场景不符合需求
2. 创建一个可以复用的类，使该类可以和其他不相关的类或不可预见得类协同工作
3. 在不对每一个都进行子类化以匹配他们接口的情况下，使用一些已经存在的子类

## 参考

http://www.cnblogs.com/V1haoge/p/6479118.html