参考地址：<https://www.cnblogs.com/goodboy-heyang/p/5037786.html>

# **[iOS设计模式 - 命令模式](https://www.cnblogs.com/goodboy-heyang/p/5037786.html)**

 前言：

命令对象封装了如何对目标执行指令的信息，因此客户端或调用者不必了解目标的任何细节，却仍可以对他执行任何已有的操作。通过把请求封装成对象，客 户端可 以把它参数化并置入队列或日志中，也能够支持可撤销操作。命令对象将一个或多个动作绑定到特定的接收器。命令模式消除了作为对象的动作和执行它的接收器之 间的绑定。

正文内容大纲：

1. 电视机、遥控器与接收器之间的关系  
   2、改变一个视图的明暗程度（未使用命令模式）  
   3、命令模式介绍  
   4、改变一个视图的明暗程度（使用命令模式）  
   5、附录

### **1.电视机、遥控器与接收器之间的关系**

电视机没有遥控器当然也能够换台  
如果要用遥控器，电视机上面肯定要有红外线接收器，专门用来接受遥控器的信息。  
红外线接收器专门接受遥控器的信息，并转化成电视机里面的信息，让电视机识别。

遥控器与接收器并非必要的设备

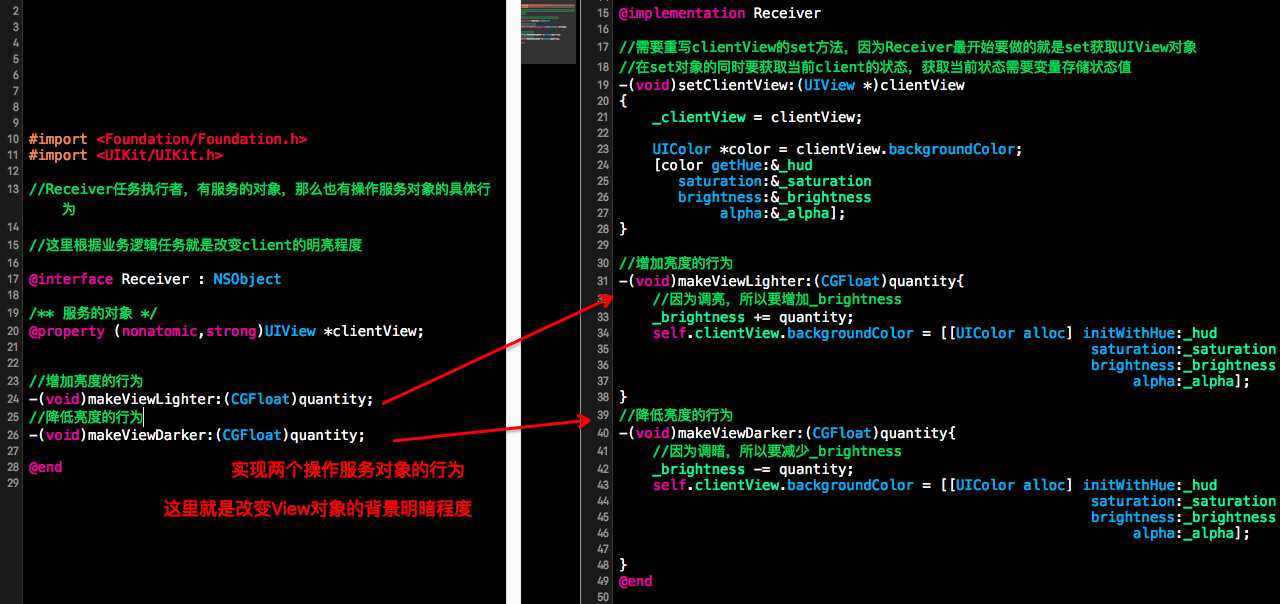
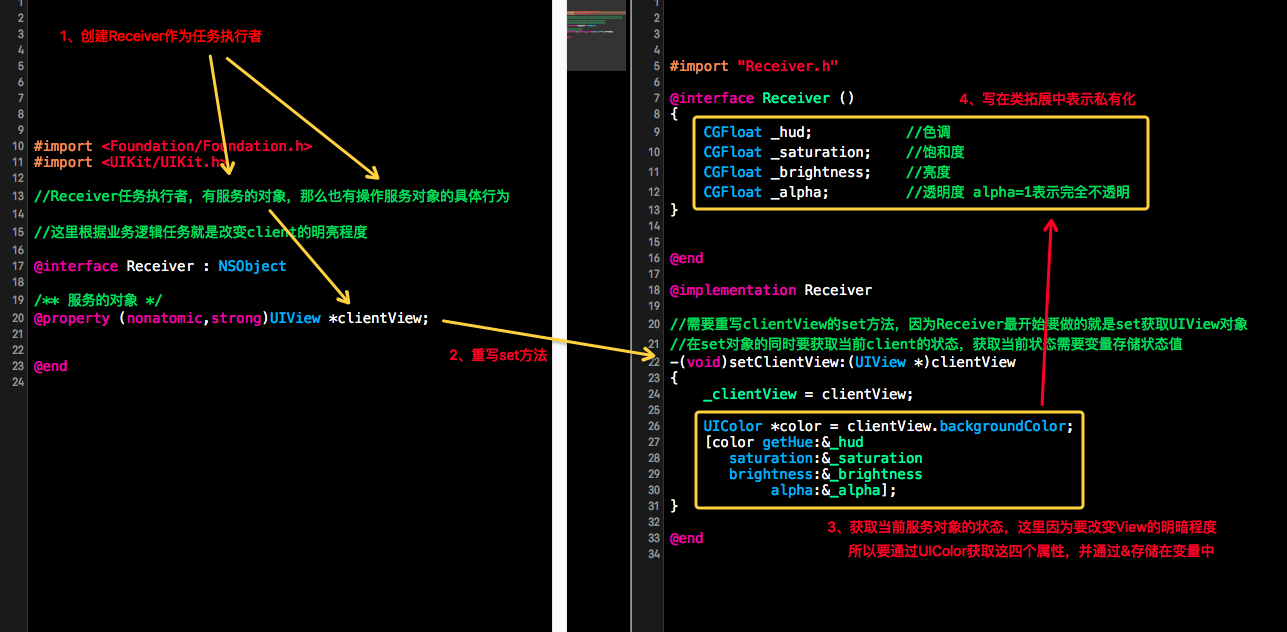
接收器转换遥控器的信号

电视机、遥控器与接收器之间的关系：电视机可以独立，不用遥控器和红外线接收器

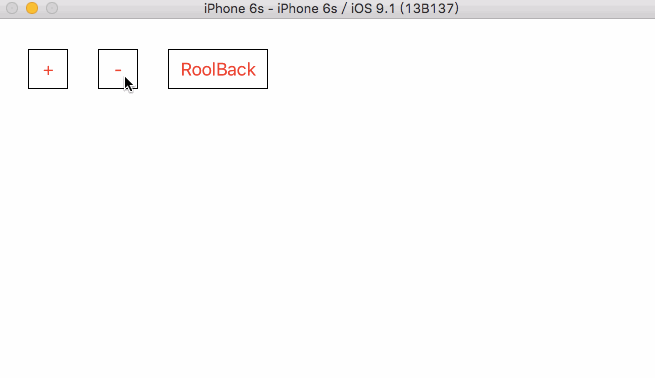
### **2.改变一个视图的明暗程度的小实例项目（未使用命令模式）**

 首先在ViewController添加三个按钮，并设置好相关属性和监听点击事件：

然后创建任务执行者对象Receiver类，创建的实例对象负责执行调整View背景明亮暗度

因为在非命令模式下，撤销逻辑不好实现，这里就不提供实现了

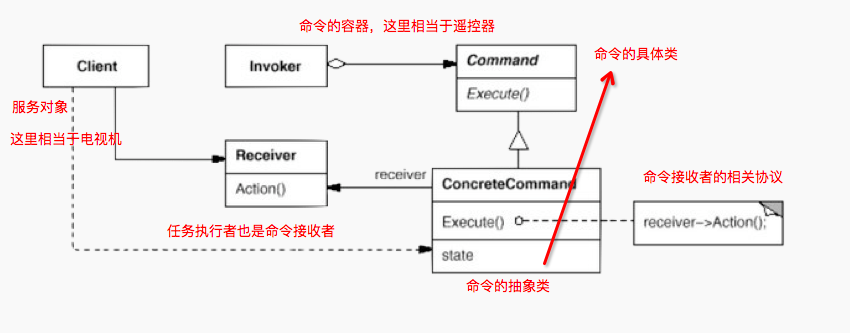
最后展示效果：

 其实到这里，Receive对象相当于ViewController的代理，代理完成控制属于Viewcontroller管理和控制的View 的背景明亮暗度调整。只不过这个代理不是那么抽象而且也不是遵循了某个协议的。是具体而直接完成逻辑业务的。这个简单的模式，在项目或者不考虑拓展性或者 某个模块功能固定了，可以使用这个模式。

但是有的业务需求会需要记录存储和取出执行任务的或者信息传递命令的状态，像这里如果要添加撤销操作，就需要记录之前的操作，然后根据之前的操作回 退反过来操作，这时候，命令模式就能比较方便的实现这个逻辑了。命令模式其实就是将发送的命令信息抽象成一个类，然后具体信息就创建具体的类，并通过回调 者添加并执行命令对象的执行命令涉及到的任务方法，同时存储记录这个命令，那么这时候因为每次操作都能存储下来，所以再来设计撤销操作就很容易了。这就是 命令模式的一个优点。

### **3.命令模式介绍**

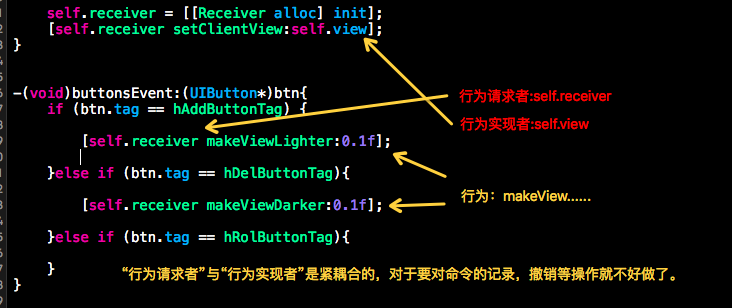
命令模式的UML图：

概述

在软件系统中，“行为请求者”与“行为实现者”通常呈现一种“紧耦合”。但在某些场合，比如要对行为进行“记录、撤销/重做、事务”等处理，  
这种无法抵御变化的紧耦合是不合适的。在这种情况下，如何将“行为请求者”与“行为实现者”解耦？将一组行为抽象为对象，实现二者之间  
的松耦合。这就是命令模式（Command Pattern）。

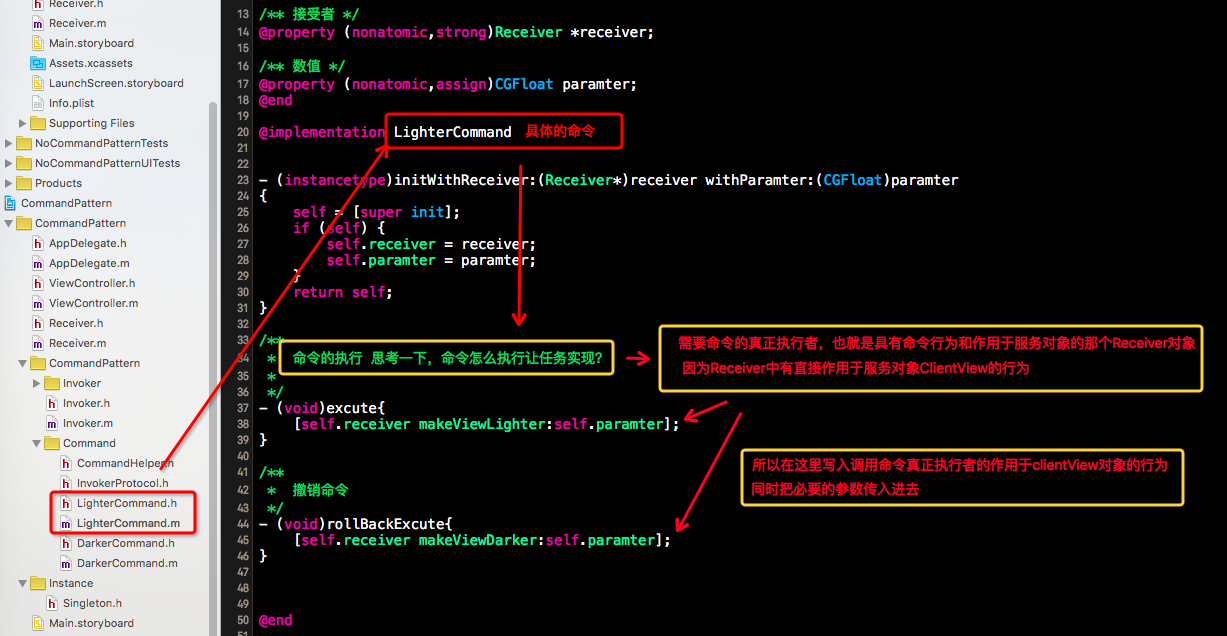
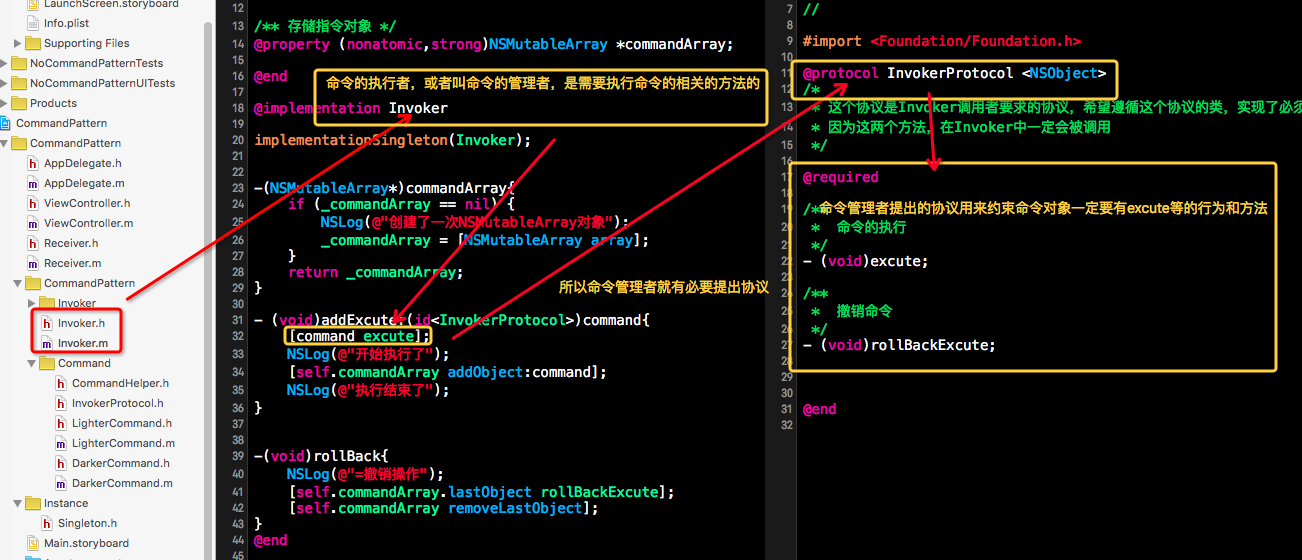
### **4.改变一个视图的明暗程度（使用命令模式）**

分析之前的代码：

为了能够行为进行“记录、撤销/重做、事务”等处理，我们可以使用命令模式：

* 设定接受者
* 将改变明暗的操作抽象成对象
* 撤销操作

分析：

最后命令的发起者和命令的执行者是解耦的。

### **5、附录**

IOS命令模式的学习网站：[http://www.cnblogs.com/YouXianMing/p/4888402.html](http://www.cnblogs.com/YouXianMing/p/4888402.html" \t "/Users/mac/Documents\\x/_blank)

 大神项目源码编程细节学习积累，所谓编程经验就是在实际代码应用中，代码的用法，代码的风格，代码表述逻辑的方式，分析和积累真正的有几年开发经验的大神写的代码，可以让你更快的提高编程经验值：

使用枚举设置控件的Tag值：

