



做不一样的码农

dboyaoao.blog.chinaunix.net

视野要开阔，创新灵感才能多

首页 | 博文目录 | 关于我



DBOYaoao

博客访问： 421004
博文数量： 83
博客积分： 821
博客等级： 军士长
技术积分： 1767
用户组： 普通用户
注册时间： 2011-10-23 16:17

加关注

短消息

论坛

加好友

个人简介

学校：上海交通大学软件工程 学历：
硕士 行业：从事流媒体移动开发 QQ：
412595942 邮
箱：yikail987910@gmail.com

文章分类

全部博文 (83)

网络 (1)

Android (3)

服务器大数据 (2)

设计模式 (8)

游戏开发 (2)

ffmpeg (7)

Webkit (2)

算法 (20)

网络/流媒体 (7)

C编程 (6)

硬件知识 (2)

Linux (9)

c++ (14)

未分配的博文 (0)

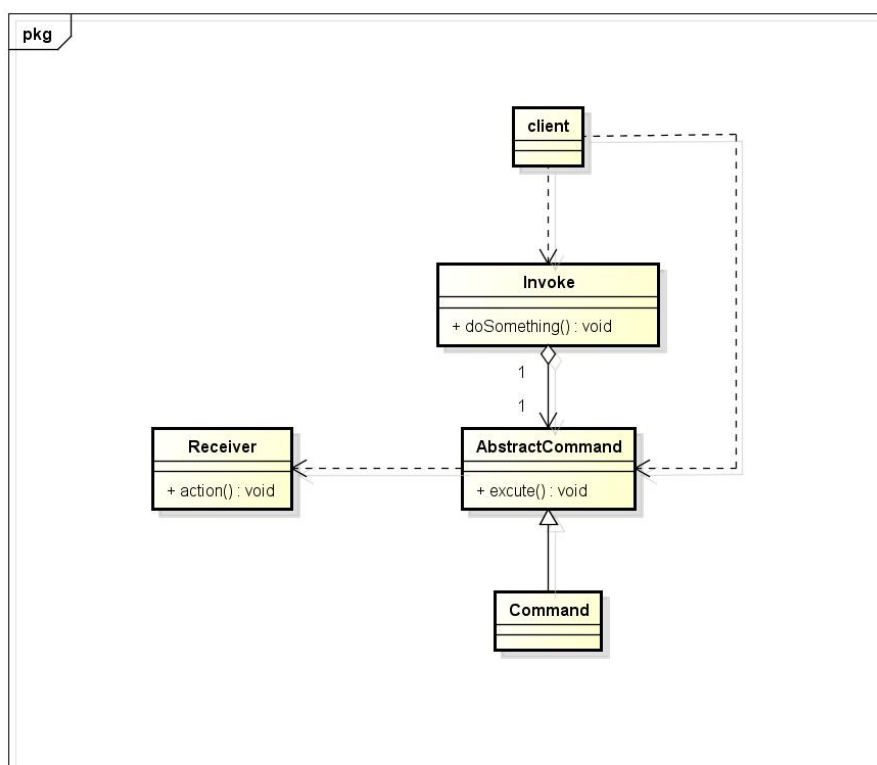
设计模式--命令模式

2014-02-17 18:34:44

分类： C/C++

命令模式其实和策略模式相似，都是将实现与调用解耦。

类图：



powered by Astah

而命令模式只是更形象的模拟了实现者和调用者之间的关系。来看一般情况下：

如果我们需要实现一个订餐的程序，一般简单的想法就是一个顾客类，一个服务员，一个厨师类。调用的情况一般可以理解为由顾客类调用服务员的点餐方法，而服务员的点餐方法又调用了厨师的烧菜方法，你会发现，这就出现了耦合的问题，依赖于实现编程了。所以该如何去改呢？

其实想想设计模式的核心思想就是一个词——抽象！！！！

我们需要将这个三个类之间抽象出来，是他们之间不相互依赖于各自的实现，而是依赖于各自的抽象接口。从这点出发，就引出了command模式，其实就是在三者之间分别建立命令抽象接口，以达到解耦的目的。

命令模式思想：

Command命令模式是一种对象行为型模式，它主要解决的问题是：在软件构建过程中，“行为请求者”与“行为实现者”通常呈现一种“紧耦合”的问题

是将实现者分装在一个类中，这个类一般就取名command类，随后通过这个command类进行调用者于实现者之间的联系。

文章存档

2015年 (16)
2014年 (18)
2013年 (12)
2012年 (28)
2011年 (9)

我的朋友



青丘凤九



小尾巴鱼



伊Eva



shisl030



crucify_

最近访客



ColdWater



garyv



lyl19



asmaster



jknuloop



Part_Kin



debuger



liuyingj



ljia696

微信关注



IT168企业级官微

微信号: IT168qiye



系统架构师大会

微信号: SACC2013

订阅

推荐博文

- A DB2 Performance Tuning Ro...
- IRQ全称为Interrupt Request...
- Hive 1.2.1&Spark&Sqoop安装...
- Swift开发iOS应用(1)列表的实...
- 向量时钟算法简介
- 最近的几个技术问题总结和答...
- 【MySQL】ibdata文件增大的原...
- 【MySQL】数据库性能测试...
- 关于 Oracle 分区索引的正确...
- 服务器进程异常的原因分析(第...

热词专题

- Qt数字软键盘
- bootargs参数设置

现在我们就看看，在客户点餐的过程中如何使用命令模式

点击[此处](#)折叠或打开

```
1. class clientorder
2. {
3. public:
4.     clientorder();
5.     ~clientorder();
6.
7.     void setorder(cookcommand* cook);
8. };
```

客户类，里面有一个订餐的接口，但是这个订餐的接口需要的是抽象的command类

点击[此处](#)折叠或打开

```
1. class cookcommand
2. {
3. public:
4.     cookcommand(){}
5.     ~cookcommand(){}
6.
7.     virtual void excute() = 0;
8. };
```

command类就是所谓的抽象层，用于调用实现者的功能。

接下来看下实现者是如何被封装的

点击[此处](#)折叠或打开

```
1. class chinesefood : public cookcommand
2. {
3. public:
4.     chinesefood(){}
5.     ~chinesefood(){}
6.
7.     void excute()
8.     {
9.         printf("cook chinese food\n");
10.    }
11. };
```

其实实现者有可能是别的独有的类，这时就要用到组合的模式将实现者对象包含在分装类中（继承当然也可以，但是设计模式中推荐少用继承，多用组合：P）

这样就会发现，调用者和实现这分离了，点餐的顾客不需要知道厨师是怎么做菜的，只要知道自己想吃什么菜，命令厨师去做就可以了

点击[此处](#)折叠或打开

```
1. int _tmain(int argc, _TCHAR* argv[])
2. {
3.     clientorder client;
4.     client.setorder(new westfood());
5.     client.setorder(new chinesefood());
6.     system("pause");
7.     return 0;
8. }
```

看到没有，命令模式的实现方法和策略模式是很相似的，只不过策略模式是偏好于方法的抽象封装，而命令模式是对于实现类的封装，不过我感觉大体上还是差不多的

阅读(2410) | 评论(0) | 转发(3) |

[上一篇：设计模式—观察者模式](#)

- linux下安装mysql
- CentOS 7:如何安装Gnome GUI ...
- Bash内置命令exec和重定向...

下一篇: ffmpeg 最简单的水印功能



相关文章

- | | |
|------------------------|---------------------------|
| test123 | linux dhcp peizhi roc |
| 编写安全代码——小心有符号数... | 关于Unix文件的软链接 |
| 使用openssl api进行加密解密... | 求教这个命令什么意思，我是新... |
| 一段自己打印自己的c程序... | sed -e "/grep/d" 是什么意思... |
| sql relay的c++接口 | 谁能够帮我解决LINUX 2.6 10... |

给主人留下些什么吧!~~

评论热议

登录后评论。
[登录](#) [注册](#)

- | | | | |
|-----------|-----------|------------|----------|
| 1 厦门人才招聘网 | 4 芭提雅旅游攻略 | 7 web前端工程师 | 10 网页游戏 |
| 2 信息发布系统 | 5 美容 | 8 宝马3系促销 | 11 嵌入式培训 |
| 3 存款理财技巧 | 6 德国开元旅行社 | 9 个人房源出租 | 12 巴厘岛自 |

[关于我们](#) | [关于IT168](#) | [联系方式](#) | [广告合作](#) | [法律声明](#) | [免费注册](#)

Copyright 2001-2010 ChinaUnix.net All Rights Reserved 北京皓辰网域网络信息技术有限公司. 版权所有

感谢所有关心和支持过ChinaUnix的朋友们
京ICP证041476号 京ICP证060528号