设计模式之代理模式

最近《软件体系/设计模式》要汇报代理模式,查找了很多代理模式的资料,整理了下面的一些关于代理模式的文章.

（铁血肯德基-Eastmount 制作）

一.模式产生的原因

在面向对象系统中,有些对象由于某些原因,比如:对象开销太大、某些操作需要安全控制、或者要访问的对象在远程的机器上,直而采用接访问会给使用者或系统结构带来很多麻烦。

现在,我们可以在访问此对象时添加一个对此对象的访问层——代理。

例如:Web安全课程中讲述的浏览网页的代理工具WebScarab、购买火车票时的代售点、银行交易支付的支付宝等。上面这些例子都用到了代理的概念,那么究竟什么是代理模式呢?

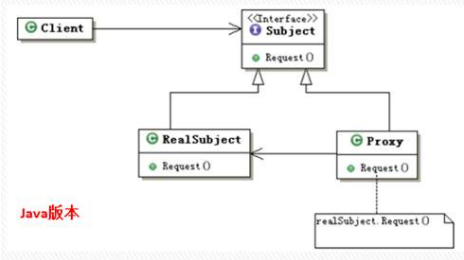
二.模式的定义和结构

模式定义

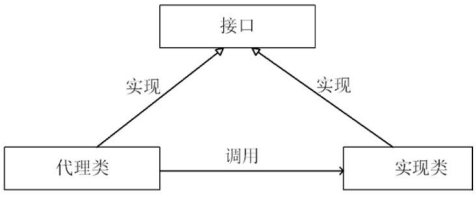
代理模式(Proxy Pattern/Surrogat)：给某一个对象提供一个代理，并由代理对象控制对原对象的引用。代理模式是一种对象结构型模式。

简而言之:一个客户不想或者不能直接引用一个对象,此时可以通过”代理”的第三者来实现间接引用,代理对象可以在客户端和目标对象之间起到中介作用,可以通过代理对象来去掉客户不能看到的内容和服务或者添加客户需要的额外服务.

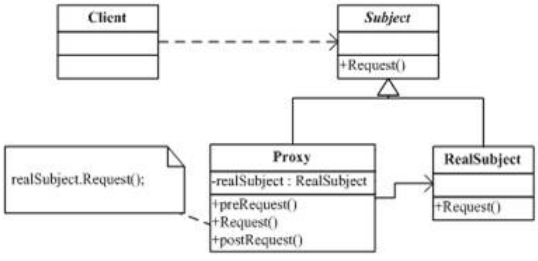
下面是一张代理模式的JAVA结构图:



代理模式结构图简化为如下:



代理模式抽象成类并设立一些方法后的结构图为:



代理模式包含如下角色:

Proxy:代理主题角色.内容包含对真实主题的引用,并提供了与真实主题角色相同的接口,以便在任何时候都可以替代真实主题.

Subject:抽象主题角色.声明真实主题和代理主题的共同接口,这样可以在任何使用真实主题的地方可以使用代理主题.

RealSubject:真实主题角色.定义真实的对象.

上面是代理模式的3个角色,下面介绍代理模式的常见几种模式及分类.

三.常见代理模式

根据《Java与模式》书中对代理模式的分类，代理模式分为8种，这里将几种常见的、重要的列举如下：

1.远程（Remote）代理：为一个位于不同的地址空间的对象提供一个局域代表对象。比如：你可以将一个在世界某个角落一台机器通过代理假象成你局域网中的一部分。

2.虚拟（Virtual）代理：根据需要将一个资源消耗很大或者比较复杂的对象延迟的真正需要时才创建。比如：如果一个很大的图片，需要花费很长时间才能显示出来，那么当这个图片包含在文档中时，使用编辑器或浏览器打开这个文档，这个大图片可能就影响文档的阅读，这时需要做个图片Proxy来代替真正的图片。

3.保护（Protector Access）代理：控制对一个对象的访问权限。比如：在论坛中，不同的身份登陆，拥有的权限是不同的，使用代理模式可以控制权限（当然，使用别的方式也可以实现）。

4.智能引用（SmartReference）代理：提供比对目标对象额外的服务。比如：纪录访问的流量（这是个再简单不过的例子），提供一些友情提示等等。

5.动态(DynamicProxy)代理：较为高级的代理模式,InvocationHandler接口和Proxy类(动态代理类).

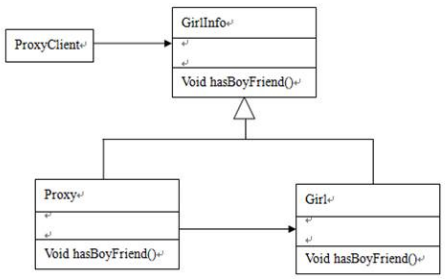
这里就不详细介绍各种代理模式分类了,详见设计模式书中.其中动态代理是较为特殊的代理模式.

四.实例分析

实例一:男孩女孩

一个男孩boy喜欢上了一个女孩girl，男孩一直想认识女孩，直接去和女孩打招呼吧，又觉得不好意思。于是男孩想出了一个办法，委托女孩的室友Proxy去帮他搞定这件事（获得一些关于女孩的信息，如QQ、手机号、人人号等，）。

其结构图为:



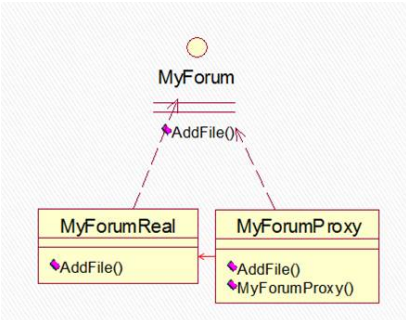
它的抽象主题角色是GirlInfo,代理主题角色是Proxy,真实主题角色是Girl,通过Client实例化.代码实现如下:

```java
//抽象主题角色:GirlInfo
interface GirlInfo{
    public void hasBoyFriend();
}
//真实主题角色: Girl
class Girl implements GirlInfo {
    public void hasBoyFriend(){
        System.out.println("还没有男朋友");
    }
}
//代理主题角色: Proxy
class Proxy implements GirlInfo {
    private GirlInfo _girl;
    public Proxy(GirlInfo girl) {
        _girl=girl;
    }
    public void hasBoyFriend() {
      _girl.hasBoyFriend();
    }
}
//Client实例化
public class ProxyClient {
    public static void main(String[] args) {
        GirlInfo girl=new Girl();
        Proxy proxy=new Proxy(girl);
        proxy.hasBoyFriend();
    }
}
```

实例二:论坛

在论坛中已注册用户和游客的权限不同,已注册的用户拥有发帖,修改自己的注册信,修改自己的帖子等功能;而游客只能看到别人发的帖子,没有其他权限.为了简化代码,更好的显示出代理模式的骨架,我们这里只实现发帖权限的控制.

其结构图为:



它的抽象主题角色是MyForum,代理主题角色是MyForumProxy,真实主题角色是MyForumReal.代码实现如下:

```java
//抽象主题角色:MyForum
public interface MyForum
{
    public void AddFile();  //发帖功能
}

//真实主题角色: MyForumReal
public class MyForumReal implements MyForum
{
    public void AddFile() {
        System.out.println("可以增加帖子!");
    }
}
```

```java
//代理主题角色：Proxy
public class MyForumProxy implements MyForum
{
    private MyForumReal forum ;
    private int permission;         //权限
    public MyForumProxy(int permission){
        forum =new MyForumReal();
        this.permission=permission;
    }
    public void AddFile() {
        if(this.permission==1){
            forum.AddFile();
        }else{
            System.out.println("不可以发帖！");
        }
    }
}

//Client实例化
public class ProxyClient
{
    public static void main(String[] args) {
        MyForumProxy proxy =new MyForumProxy(0);
        proxy.AddFile();
    }
}
```

五.代理模式的优点

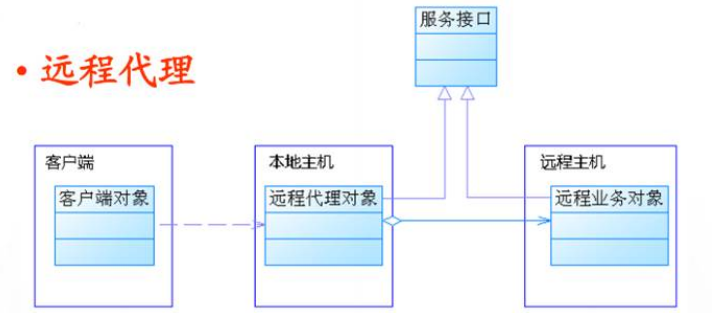
分析问题:

(1).当我们在网站上浏览图文信息时,进场使用代理模式,访问网页时调用的不是真实装载的图片,而是在代理对象方法中先使用一个线程向浏览器装载一个缩小的图片.

如果要仔细查看这个图片,则点击缩小的图片(代理图片)并调用响应的方法,在后台使用一个线程来调用真实的装载大图片的方法将图片加载到本地,并在网页中显示出来.

这就一个通过代理模式实现图片的加载放在后台操作,使其不影响前台的浏览.

(2).远程代理可以将网络的细节隐藏起来,使得客户端不必考虑网络的存在,使计算机具有更快的响应和处理速度.



综上所述:

代理模式的常见优点如下:

1.代理模式能够协调调用者和被调用者,在一定程度上降低了系统的耦合度;如加载图片中后台操作不影响前台效果;

2.在远程代理中使得客户端可以访问在远程机器上的对象,远程机器具有更好的计算性能和处理速度,响应客户端请求;

3.代理模式通过使用一个小对象来替代一个大对象,以减少系统资源的消耗,对系统进行优化并提高运行速度,主要表现在虚拟代理;

4.保护代理可以控制对真实对象的使用权限;

5.copy-on-write可以大幅度的降低拷贝庞大实体时的开销.

六.代理模式的缺点

1.由于在客户端和真实主题之间增加代理对象,因此有些类型的代理模式可能会造成请求的处理速度慢;

2.实现代理模式需要额外的工作,有些代理模式的实现非常复杂;

3.如果是一个很小的系统,功能也不是很繁杂,那么使用代理模式可能就显得臃肿,需要修改原有的方法.

七.总结

代理模式是一个常用的模式,代理模式能够协调调用者和被调用者，能够在一定程度上降低系统的耦合度。在项目中比如打开网页中的图片缩略图,远程管理系统都经常使用到代理模式。当我们直接访问一个对象很困难,或者说不能访问,此时只能是找个代理去访问,然后把结果反馈给自己。代理模式中各种分类这里就不具体介绍它们的实现代码和实例了.想学习的同学建议看看清华大学出版社刘伟编写的代理模式书及课件.

最后说明一下,该文章的知识来自于：自己的分析与理解,《大话设计模式》,《Head First 设计模式》,还有清华大学出版社刘伟编写的相关课件,还有3个博客的相关阅读.建议可以看看下面的3个作者的理解.

http://tech.ddvip.com/2008-10/122362574376324.html

http://blog.csdn.net/hguisu/article/details/7542143

http://blog.csdn.net/ai92/article/details/216424

在这里感谢上面所有的书籍,课件,博主.由于作者才接触CSDN博客,可能格式内容不是很好,还请海涵!文章中有不足之处或错误的地方,作者能力有限,请见谅.希望文章能帮助大家更好的了解代理模式.（铁血肯德基-Eastmount制作）