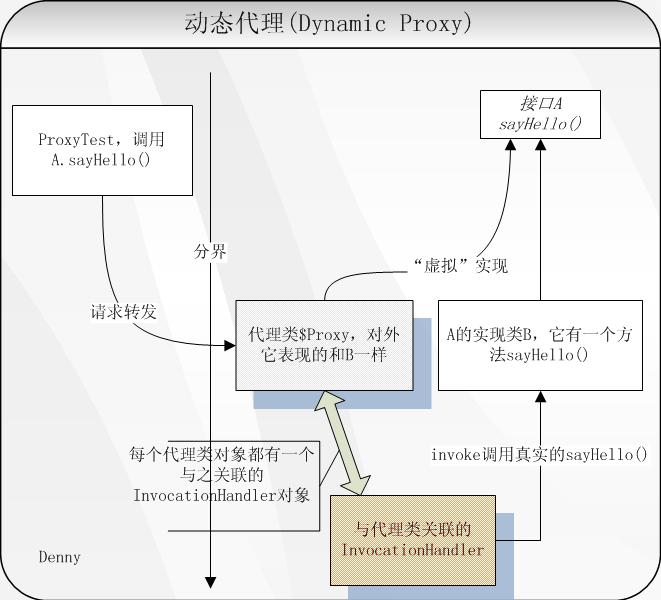
### [java动态代理学习笔记](http://langyu.iteye.com/blog/410071)

**博客分类：** [java](http://langyu.iteye.com/category/61048) [Java](http://www.iteye.com/blogs/tag/Java)[C](http://www.iteye.com/blogs/tag/C)[C++](http://www.iteye.com/blogs/tag/C++)[C#](http://www.iteye.com/blogs/tag/C%23)[Blog](http://www.iteye.com/blogs/tag/Blog)

没事的时候翻看lang.reflect包下的代码，发现有两部分内容:涉及反射和动态代理。   
很多地方都可以看到动态代理的影子，只是一直没仔细看下。   
  
在学习之前，先提出几个问题，带着问题来看代码:   
  
1.什么是动态代理?   
2.为什么使用动态代理?   
3.使用它有哪些好处?   
4.哪些地方需要动态代理?   
  
--------------------分隔线-----------------------------   
  
和动态代理有关的有两个类   
1.interface InvocationHandler   
Object invoke(Object proxy, Method method, Object[] args)   
只这一个方法，后面再说   
  
2.class Proxy   
真正表示动态代理的类，提供两个静态方法:   
  
Class<?> getProxyClass(ClassLoader loader, Class<?>[] interface)   
用来产生代理类，参数要提供interface数组，它会生成这些interface的“虚拟实现”，   
用来冒充真实的对象。   
  
Object newProxyInstance(ClassLoader loader, Class<?>[] interfaces, InvocationHandler h)   
产生代理对象，多了InvocationHandler参数(只是InvocationHandler接口的实现类)，   
它与代理对象关联，当请求分发到代理对象后，会自动执行h.invoke(...)方法，   
invoke方法就是我们用来做N多事情的地方 -\_-。   
  
--------------------分隔线-----------------------------   
  
看完上面的代码，大致明白动态代理的含义：   
A接口有c方法，类B实现A接口，原本应该是执行B类中的c方法，可现在不这样做；   
我声明产生B类的代理类B',由它来冒充B类的“兄弟”并“实现”A接口，   
对外界来说B'应该也有c方法，可当真正调用它的时候，   
它会去执行与它关联InvocationHandler的invoke()方法，   
在这个方法里面你可以做很多事情。这样，这个请求就被“代理”到其它地方去了。   
  
下面是根据我的理解画的一个说明图   
  
  
  
--------------------分隔线-----------------------------   
  
引用网上的一个例子来说明问题(有部分改动，转载自：http://callan.iteye.com/blog/161806)   
真实的接口:

Java代码  [收藏代码](javascript:void())

1. public interface Hello {
3. void sayHello(String to);
5. void print(String p);
7. }

它的真实实现类:

Java代码  [收藏代码](javascript:void())

1. public class HelloImpl implements Hello {
3. public void sayHello(String to) {
4. System.out.println("Say hello to " + to);
5. }
7. public void print(String s) {
8. System.out.println("print : " + s);
9. }
11. }

在这里生成与代理类相关联的InvocationHandler对象

Java代码  [收藏代码](javascript:void())

1. public class LogHandler implements InvocationHandler {
3. private Object dele;
5. public LogHandler(Object obj) {
6. this.dele = obj;
7. }
9. public Object invoke(Object proxy, Method method, Object[] args) throws Throwable {
10. doBefore();
11. //在这里完全可以把下面这句注释掉，而做一些其它的事情
12. Object result = method.invoke(dele, args);
13. after();
14. return result;
15. }
17. private void doBefore() {
18. System.out.println("before....");
19. }
21. private void after() {
22. System.out.println("after....");
23. }
24. }

最后是测试类:

Java代码  [收藏代码](javascript:void())

1. public class ProxyTest {
3. public static void main(String[] args) {
4. HelloImpl impl = new HelloImpl();
5. LogHandler handler = new LogHandler(impl);
6. //这里把handler与impl新生成的代理类相关联
7. Hello hello = (Hello) Proxy.newProxyInstance(impl.getClass().getClassLoader(), impl.getClass().getInterfaces(), handler);
9. //这里无论访问哪个方法，都是会把请求转发到handler.invoke
10. hello.print("All the test");
11. hello.sayHello("Denny");
12. }
14. }

这里是输出结果:

Java代码  [收藏代码](javascript:void())

1. before....
2. print : All the test
3. after....
4. before....
5. Say hello to Denny
6. after....

--------------------分隔线-----------------------------   
  
最后试着来回答之前提出的问题:   
1.什么是动态代理?   
一种用于转发请求，进行特殊处理的机制，“动态”应该指的是“运行期”。   
2.为什么使用动态代理?   
可以对请求进行任何处理(如事务，日志等，这都是网上说的，我当然可以做任何处理)   
3.使用它有哪些好处?   
如上   
4.哪些地方需要动态代理?   
不允许直接访问某些类；对访问要做特殊处理等，我只能想到这些。   
  
--------------------分隔线-----------------------------   
  
其它一些想法：   
  
1.如果想声明产生B类的代理类，那个B类必须要实现接口，如果没有接口，   
代理类就不能伪装成B类的“兄弟”，也就没有存在的意思，   
其实也可以伪装成B类的“孩子”，对外他们有共同的接口，可以这样做吧?   
  
2.当请求代理类的方法时，这个请求会被转到执行与代理类关联InvocationHandler   
的invoke方法。那InvocationHandler到底是什么?对它的理解可以是这样:   
它用来处理方法的调用，实现类也有同样的意义；与代理类对象相关联则表示，   
它就是负责处理代理类应该有的动作，把所有的方法请求分发到invoke这个方法上。   
  
  
  
--------------------分隔线-----------------------------   
学习后总结，既可以全面地观察分析，又能加深印象。   
如果我的理解有误，别人的指证会对我产生积极影响。   
如果我的理解正确，帮助其它人理解是我的荣幸。   
初学者，请多多指教。