链滴

一个简单 JDK 动态代理的实例

作者：seanlee

原文链接：https://ld246.com/article/1503239250620

来源网站：链滴

<h3 id="动态代理的步骤">动态代理的步骤： </h3>

创建一个实现了InvocationHandler接口的类，必须重写接口里的invoke（）方法。
创建被代理的类和接口
通过Proxy的静态方法 newProxyInsatance(ClassLoader loader,Class[] interfaces,InvocationHandler,handler)来创建一个代理
通过代理来调用方法

<h3 id="简单的动态代理实例">简单的动态代理实例</h3>
<pre><code class="hljs">package com.sean.zzzjvm;
import java.lang.reflect.InvocationHandler;
import java.lang.reflect.Method;
import java.lang.reflect.Proxy;
/**
*

@Author Sean
@Date 2017/8/20 21:43.

@Version

*/
public class DynamicProxyTest {
<pre><code class="highlight-chroma">//定义一个接口
interface IHello</span&gt{
    void sayHello</spa>()</spa>;
}

//实现该接的类
static</san> class</span&t; Hello implements IHello{

    <span class=

hljs-annotation" style="box-sizing: border-box;">@Override
    <span class=
hljs-function" style="box-sizing: border-box;"><span class="hljs-keyword" style="box-si
ing: border-box; color: #333333; font-weight: bold;">public <span class="h
js-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">void</s
an> <span class="hljs-title" style="box-sizing: border-box; color: #990000; font-weight:
old;">sayHello<span class="hljs-params" style="box-sizing: border-box;"
gt;() {
        System.out.pr
ntln("hello wo
ld");
    }
}

<span class="hl
s-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">//创建一个
态代理类，实现InvocationHandler接口
<span class="hl
s-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">static</s
an> <span class="hljs-ke
word" style="box-sizing: border-box; color: #333333; font-weight: bold;">class</span&
t; <span class="hljs-title" style="box-sizing: border-box; color: #445588; font-weight: bold;
>DynamicProxy <span class="hljs-keyword" style="box-sizing: border-box;
color: #333333; font-weight: bold;">implements <span class="hljs-title" sty
e="box-sizing: border-box; color: #445588; font-weight: bold;">InvocationHandler</spa
>{
    Object original
bj;

    <span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">//创建一
代理的方法，在new DynamicProxy().bind(new Hello());执行
    <span class=
hljs-function" style="box-sizing: border-box;">Object <span class="hljs-title" style="box
sizing: border-box; color: #990000; font-weight: bold;">bind <span class=
hljs-params" style="box-sizing: border-box;">(Object originalObj)</span
gt;{
        <span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">this&lt
/span>.originalObj = originalObj;
        <span clas
="hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">//返回
个代理对象
        <span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">retur
 Proxy.newProxyInstance(originalObj.getClass().getClassLoader(),
                original
bj.getClass().getInterfaces(),<span class="hljs-keyword" style="box-sizing: border-box; color
 #333333; font-weight: bold;">this);
    }

    <span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">//默认重
的方法，在 hello.sayHello()执行

    <span class=
hljs-annotation" style="box-sizing: border-box;">@Override
    <span class=
hljs-function" style="box-sizing: border-box;"><span class="hljs-keyword" style="box-si
ing: border-box; color: #333333; font-weight: bold;">public Object <span c
ass="hljs-title" style="box-sizing: border-box; color: #990000; font-weight: bold;">invoke&
t;/span>(Object proxy, M
thod method, Object[] args) <span class="hljs-keyword" style="box-sizing: bo
der-box; color: #333333; font-weight: bold;">throws Throwable </span&gt
{
        System.out.pr
ntln("welcom
");
        <span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">retur
 method.invoke(originalObj,args);
    }
}

<span class="hl
s-function" style="box-sizing: border-box;"><span class="hljs-keyword" style="box-sizi
g: border-box; color: #333333; font-weight: bold;">public <span class="hlj
-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;">static</sp
n> <span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-wei
ht: bold;">void <span class="hljs-title" style="box-sizing: border-box; color
 #990000; font-weight: bold;">main<span class="hljs-params" style="box-
izing: border-box;">(String[] args){
    <span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">// IHello
ello = new Hello();
    <span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;">//调用动
代理的方法
    IHello hello = (
Hello) <span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-wei
ht: bold;">new DynamicProxy().bind(<span class="hljs-keyword" style="bo
-sizing: border-box; color: #333333; font-weight: bold;">new Hello());
    hello.sayHello();
}
</code></pre>
</code> <code class="hljs">}</code> </pre>