



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](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<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 newProxyInstance(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"> <span class="hljs-keyword">package</span> com.sean.zzzjvm;
```

```
<p> <span class="hljs-keyword">import</span> java.lang.reflect.InvocationHandler;<br>
```

```
<span class="hljs-keyword">import</span> java.lang.reflect.Method;<br>
```

```
<span class="hljs-keyword">import</span> java.lang.reflect.Proxy;</p>
```

```
<p> <span class="hljs-comment">/**<br>
```

```
*</span></p>
```


 @Author Sean

 @Date 2017/8/20 21:43.

@Version


```
<p>*/<br>
```

```
<span class="hljs-keyword">public</span> <span class="hljs-class"><span class="hljs-keyword">class</span> <span class="hljs-title">DynamicProxyTest</span> </span>{</p>
```

```
<pre> <code class="highlight-chroma"><span class="highlight-line"><span class="highlight-cl">&lt;span class="hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//定义一个接口&lt;/span&gt;
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hljs-class" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-sizing: order-box; color: #333333; font-weight: bold;"&gt;interface&lt;/span&gt; &lt;span class="hljs-title" style="box-sizing: border-box; color: #445588; font-weight: bold;"&gt;Hello&lt;/span&gt;
```

```
&lt;/span&gt;{<br></span></span><span class="highlight-line"><span class="highlight-cl"> &lt;span class="hljs-function" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;void&lt;/span&gt; &lt;span class="hljs-title" style="box-sizing: border-box; color: #990000; font-weight: bold;"&gt;sayHello&lt;/span&gt;
```

```
&lt;span&gt;&lt;span class="hljs-params" style="box-sizing: border-box;"&gt;()&lt;/span&gt;&lt;/span&gt;&lt;/span&gt;&lt;/span></span><span class="highlight-line"><span class="highlight-cl">&lt;/span>&lt;/span></span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hljs-class" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//实现该接
```

```
的类&lt;/span&gt;
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;static&lt;/span&gt;
```

```
&lt;span&gt; &lt;span class="hljs-class" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;class&lt;/span&gt;
```

```
&lt;span&gt; &lt;span class="hljs-title" style="box-sizing: border-box; color: #445588; font-weight: bold;"&gt;Hello&lt;/span&gt;
```

```
&lt;span&gt;&lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;implements&lt;/span&gt; &lt;span class="hljs-title" style="box
```

```

hljs-annotation" style="box-sizing: border-box;"&gt;@Override&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl"> &lt;span class=
hljs-function" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-si
ing: border-box; color: #333333; font-weight: bold;"&gt;public&lt;/span&gt; &lt;span class="h
js-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;void&lt;/s
an&gt; &lt;span class="hljs-title" style="box-sizing: border-box; color: #990000; font-weight:
old;"&gt;sayHello&lt;/span&gt;&lt;span class="hljs-params" style="box-sizing: border-box;"
gt;()&lt;/span&gt; &lt;/span&gt;{
</span></span><span class="highlight-line"><span class="highlight-cl">      System.out.pr
ntln(&lt;span class="hljs-string" style="box-sizing: border-box; color: #dd1144;"&gt;"hello wo
ld"&lt;/span&gt;);
</span></span><span class="highlight-line"><span class="highlight-cl">  }
</span></span><span class="highlight-line"><span class="highlight-cl">}&lt;/span>
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hl
s-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//创建一个
态代理类，实现InvocationHandler接口&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hl
s-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;static&lt;/s
an&gt; &lt;span class="hljs-class" style="box-sizing: border-box;"&gt;&lt;span class="hljs-ke
word" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;class&lt;/span&
t; &lt;span class="hljs-title" style="box-sizing: border-box; color: #445588; font-weight: bold;
&gt;DynamicProxy&lt;/span&gt; &lt;span class="hljs-keyword" style="box-sizing: border-box;
color: #333333; font-weight: bold;"&gt;implements&lt;/span&gt; &lt;span class="hljs-title" sty
e="box-sizing: border-box; color: #445588; font-weight: bold;"&gt;InvocationHandler&lt;/spa
&gt;&lt;/span&gt;{
</span></span><span class="highlight-line"><span class="highlight-cl">  Object original
bj;
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl"> &lt;span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//创建一
代理的方法，在new DynamicProxy().bind(new Hello());执行&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl"> &lt;span class=
hljs-function" style="box-sizing: border-box;"&gt;Object &lt;span class="hljs-title" style="box
sizing: border-box; color: #990000; font-weight: bold;"&gt;bind&lt;/span&gt; &lt;span class=
hljs-params" style="box-sizing: border-box;"&gt;(Object originalObj)&lt;/span&gt;&lt;/span
gt;{
</span></span><span class="highlight-line"><span class="highlight-cl">      &lt;span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;this&lt;/
span&gt;.originalObj = originalObj;
</span></span><span class="highlight-line"><span class="highlight-cl">      &lt;span clas
="hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//返回
个代理对象&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">      &lt;span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;retur
&lt;/span&gt; Proxy.newProxyInstance(originalObj.getClass().getClassLoader(),
</span></span><span class="highlight-line"><span class="highlight-cl">          original
bj.getClass().getInterfaces(),&lt;span class="hljs-keyword" style="box-sizing: border-box; color
#333333; font-weight: bold;"&gt;this&lt;/span&gt;);
</span></span><span class="highlight-line"><span class="highlight-cl">  }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl"> &lt;span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//默认重
的方法，在hello.sayHello()执行&lt;/span&gt;

```

```

</span></span><span class="highlight-line"><span class="highlight-cl">    &lt;span class=
hljs-annotation" style="box-sizing: border-box;"&gt;@Override&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">    &lt;span class=
hljs-function" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-si
ing: border-box; color: #333333; font-weight: bold;"&gt;public&lt;/span&gt; Object &lt;span c
ass="hljs-title" style="box-sizing: border-box; color: #990000; font-weight: bold;"&gt;invoke&
t;/span&gt;&lt;span class="hljs-params" style="box-sizing: border-box;"&gt;(Object proxy, M
thod method, Object[] args)&lt;/span&gt; &lt;span class="hljs-keyword" style="box-sizing: bo
der-box; color: #333333; font-weight: bold;"&gt;throws&lt;/span&gt; Throwable &lt;/span&gt;
{
</span></span><span class="highlight-line"><span class="highlight-cl">        System.out.pr
ntln(&lt;span class="hljs-string" style="box-sizing: border-box; color: #dd1144;"&gt;"welcom
"&lt;/span&gt;);
</span></span><span class="highlight-line"><span class="highlight-cl">        &lt;span clas
="hljs-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;retur
&lt;/span&gt; method.invoke(originalObj,args);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">}&lt;/span></span>
</span></span><span class="highlight-line"><span class="highlight-cl">&lt;span class="hl
s-function" style="box-sizing: border-box;"&gt;&lt;span class="hljs-keyword" style="box-sizi
g: border-box; color: #333333; font-weight: bold;"&gt;public&lt;/span&gt; &lt;span class="hlj
-keyword" style="box-sizing: border-box; color: #333333; font-weight: bold;"&gt;static&lt;/sp
n&gt; &lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-wei
ht: bold;"&gt;void&lt;/span&gt; &lt;span class="hljs-title" style="box-sizing: border-box; color
#990000; font-weight: bold;"&gt;main&lt;/span&gt;&lt;span class="hljs-params" style="box-
izing: border-box;"&gt;(String[] args)&lt;/span&gt;&lt;/span&gt;{
</span></span><span class="highlight-line"><span class="highlight-cl">    &lt;span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;// IHello
ello = new Hello();&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">    &lt;span class=
hljs-comment" style="box-sizing: border-box; color: #999988; font-style: italic;"&gt;//调用动
代理的方法&lt;/span&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">    IHello hello = (
Hello) &lt;span class="hljs-keyword" style="box-sizing: border-box; color: #333333; font-wei
ht: bold;"&gt;new&lt;/span&gt; DynamicProxy().bind(&lt;span class="hljs-keyword" style="bo
-sizing: border-box; color: #333333; font-weight: bold;"&gt;new&lt;/span&gt; Hello());
</span></span><span class="highlight-line"><span class="highlight-cl">    hello.sayHello();
</span></span><span class="highlight-line"><span class="highlight-cl">}&lt;/span></span>
</span></span></code></pre>
</code><p><code class="hljs"></code></p></pre><p></p>

```