链滴

常见的几种java单例模式代码解析

作者：wanglei0622

原文链接：https://ld246.com/article/1462527974547

来源网站：链滴

<h2 style="text-align: left;">单例模式</h2>
是一种常见的设计模式，单例模式分三种：懒汉式单例、饿汉单例、登记式单例三种。
　　单例模式有一下特点：
　　1、单例类只能有一个实例。
　　2、单例类必须自己自己创建自己的唯实例。
　　3、单例类必须给所有其他对象提供这一实例
单例模式确保某个类只有一个实例，而且自行实例化向整个系统提供这个实例。在计算机系统中，线程池、缓存、日志对象、对话框、打印机、显卡的驱程序对象常被设计成单例。这些应用都或多或少具有资源管理器的功能。每台计算机可以有若干个打机，但只能有一个Printer Spooler，以避免两个打印作业同时输出到打印机中。每台计算机可以有若通信端口，系统应当集中管理这些通信端口，以避免一个通信端口同时被两个请求同时调用。总之，择单例模式就是为了避免不一致状态，复用对象节省资源。
<h2 style="text-align: left;">懒汉单例模式实践</h2>
<pre class="brush: java">/**
 *
 * @ClassName: singleton
 * @Description: 懒汉单例模式，调用getInstance方法时才创建实例，实现懒加载
 * @author Lei Wang
 * @date 2016-5-6 下午4:46:54
 */
public class Singleton
{

private static Singleton single = null;

/**
 * 私有化构造方法
 *
 * @author Lei Wang
 */
private Singleton()
{

}

/**
 *
 * @Title: getInstance
 * @Description: 判断是否已经有Singleton实例存在，获得Singleton实例
 * @author Lei Wang
 * @return Singleton实例
 * @throws
 */
public static Singleton getInstance()
{

    if (null != single)
    {
        return single;
    }
    else
    {
        single = new Singleton()

```
        return single;

    }

}

}</pre>

<p><span><span>这种是比较常见的单例实现代码，<span>但是以上实现没有考虑线程安全问
。在<span>在并发环境下很可能出现多个Singleton实例。可以用<span>synchronized来解决线
安全问题，但是会牺牲效率。</span></span></span></span></span></p>
<p><span><span><span><span><span>还可以用静态内部类实现，解决线程安全问题，不会
牲效率</span></span></span></span></span></p>
<pre class="brush: java">  /**
         *
         * @ClassName: LazyHolder
         * @Description: 静态内部类，这种方式同样利用了classloder的机制来保证初始化instance时
有一个线程，不同的是staic修饰的变量只要Singleton类被装载了，那么instance就会被实例化（没
达到lazy loading效果），而这种方式是Singleton类被装载了，instance不一定被初始化。因为Singl
tonHolder类没有被主动使用，只有显示通过调用getInstance方法时，才会去装载LazyHolder类，
而实例化instance。
         * @author &lt;a href="http://www.wanglay.com"&gt;Lei Wang&lt;/a&gt;
         * @date 2016-5-6 下午5:21:46
         */
        private static class LazyHolder
        {
            private static final Singleton INSTANCE = new Singleton();
        }</pre>
<p><span><span><span><span><span> </span></span></span></span></span
</p>
<h2><span><span><span><span><span>饿汉式单例实践</span></span></span></span
</span></h2>
<pre class="brush: java">/**
 *
 * @ClassName: singleton
 * @Description: TODO(懒汉单例模式.在类加载时实例化自己 )
 * @author &lt;a href="http://www.wanglay.com"&gt;Lei Wang&lt;/a&gt;
 * @date 2016-5-6 下午4:46:54
 */
public class Singleton
{
    /**
     * 在类加载时实例化对象
     */
    private final static Singleton INSTANCE = new Singleton();


/**
 * 私有化构造方法
 *
 * @author &lt;a href="http://www.wanglay.com"&gt;Lei Wang&lt;/a&gt;
 */
private Singleton()
{
```

```
}

/**
 *
 * @Title: getInstance
 * @Description: 获得Singleton实例
 * @author &lt;a href="http://www.wanglay.com"&gt;Lei Wang&lt;/a&gt;
 * @return Singleton实例
 * @throws
 */
public static Singleton getInstance()
{
    return INSTANCE;

}

}</pre>

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