

【单例深思】枚举实现单例原理

2017年04月26日 20:49:58

阅读数：7293

标签：

java

单例

枚举

更多

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/Insert_day/article/details/70832185

单例的枚举实现在《Effective Java》中有提到，因为其**功能完整、使用简洁、无偿地提供了序列化机制、在面对复杂的序列化或者反射攻击**可以绝对防止多次实例化等优点，单元素的枚举类型被作者认为是实现Singleton的最佳方法。

其实现非常简单，如下：

```
public enum Singleton {  
    INSTANCE;  
    private Singleton() {}  
}
```

下面我们用一个枚举实现单个数据源例子来简单验证一下：

声明一个枚举，用于获取数据库连接。

```
public enum DataSourceEnum {  
    DATASOURCE;  
    private DBConnection connection = null;  
    private DataSourceEnum() {  
        connection = new DBConnection();  
    }  
    public DBConnection getConnection() {  
        return connection;  
    }  
}
```

模拟一个数据库连接类：

```
public class DBConnection {}
```

测试通过枚举获取的实例是否相同：

```
public class Main {  
    public static void main(String[] args) {  
        DBConnection con1 = DataSourceEnum.DATASOURCE.getConnection();  
        DBConnection con2 = DataSourceEnum.DATASOURCE.getConnection();  
        System.out.println(con1 == con2);  
    }  
}
```

输出结果为：**true** 结果表明两次获取返回了相同的实例。

下面深入了解一下为什么枚举会满足线程安全、序列化等标准。

在JDK5 中提供了大量的语法糖，枚举就是其中一种。

所谓 **语法糖 (Syntactic Sugar)**，也称糖衣语法，是由英国计算机学家 Peter.J.Landin 发明的一个术语，指在计算机语言中添加的某种语法，这种语法对语言的功能并没有影响，但是但是更方便程序员使用。只是在编译器上做了手脚，却没有提供对应的指令集来处理它。

就拿枚举来说，其实Enum就是一个普通的类，它继承自**java.lang.Enum**类。

```
public enum DataSourceEnum {  
    DATASOURCE;  
}
```

把上面枚举编译后的字节码反编译，得到的代码如下：

```
public final class DataSourceEnum extends Enum<DataSourceEnum> {  
    public static final DataSourceEnum DATASOURCE;
```

0
1

```
public static DataSourceEnum[] values();  
public static DataSourceEnum valueOf(String s);  
static {};  
}
```

由反编译后的代码可知，`DATASOURCE` 被声明为 `static` 的，根据在【单例深思】[饿汉式与类加载](#) 中所描述的类加载过程，可以知道虚拟机在验证一个类的 `<clinit>()` 方法在多线程环境中被正确的加锁、同步。所以，枚举实现是在实例化时是线程安全。

接下来看看序列化问题：

Java规范中规定，每一个枚举类型极其定义的枚举变量在JVM中都是唯一的，因此在枚举类型的序列化和反序列化上，Java做了特殊的规定。在序列化的时候Java仅仅是将枚举对象的name属性输出到结果中，反序列化的时候则是通过 `java.lang.Enum` 的 `valueOf()` 方法来根据名称来查找枚举对象。也就是说，下面枚举为例，序列化的时候只将 `DATASOURCE` 这个名称输出，反序列化的时候再通过这个名称，查找对于的枚举类型，因此反序列化后的实例也会和之前被序列化的对象实例相同。

```
public enum DataSourceEnum {  
    DATASOURCE;  
}
```

由此可知，枚举天生保证序列化单例。



想对作者说点什么



蓝色土耳其18: mark (03-28 10:41 #1楼)

enum 是实现单例最好的解决方案吗



1901

enum 是实现单例最好的解决方案吗 你一定听说过很多次，enum总是实现单例模式最好的选择。那enum是最好的选择吗？比起其它实...

为什么java中用**枚举实现单例模式**会更好

 4310

枚举单例是java中使用枚举提供一个实例对象来实现单例模式的一种新方法，虽然单例模式在java中早已存在，但枚举单例实际上从jav...

0

Java**枚举实现单例模式**

 2.5万

单例模式约束一个类只能实例化一个对象。在Java中，为了强制只实例化一个对象，最好的方法是使用一个枚举量。这个优秀的思想直...

1

Java 利用**枚举实现单例模式**

 3.2万

引言单例模式比较常见的实现方法有懒汉模式，DCL模式公有静态成员等，从Java 1.5版本起，单元素枚举实现单例模式成为最佳的方...

Java**枚举单例**

 1751

注意enum不是Enum，有Java基础的同学们应该都不会把二者混淆了。简单来说，enmu只是jdk1.5引入的语法糖，它不是java中的新...

关于**枚举式单例**的一些详解

 844

之前写过一篇单例模式,没有说明为什么枚举可以反反射,反序列化!这里详细解释下, 首先枚举都是默认集成java中Enum类的,而在枚举类...



没有时间学英语？看看胡歌怎么学！

10秒注册，免费领取【每日英语】资料

单例模式之使用enum**枚举数据类型实现**

 2139

枚举enum和静态代码块的特性想死，在使用枚举类时，构造方法会被自动调用，也可以应用这个特性实现单例模式。 package test; p...

JAVA单例模式的各种写法分析，最优为**枚举**

 5292

作用 单例模式（Singleton）：保证一个类仅有一个实例，并提供一个访问它的全局访问点 适用场景 应用中某个实例对象需要频繁的被...

java **枚举与单例**

 140

转载<http://blog.csdn.net/javazejian/article/details/71333103> 出自【zejian的博客】理解枚举类型枚举类型是Java 5中新增特性的一部分...

用枚举实现单例

 223

最近看同事的老代码有这么个类,一时之间真没看明白.找同事一解释才知道这使用枚举实现的一个单例. 忽然感觉就这么简单的单例就水...

文章热词

java word打开 java 模板+参数 java收集控制台一行 java代码抽奖 java 对象动态堆

相关热词

单例 单例注入非单例 ood单例 guice单例 idea单例

个人资料



InkWestSour...

关注

原创	粉丝	喜欢	评论
118	12	3	13

等级:  访问: 10万+

积分: 2225 排名: 2万+

勋章:  

0

1

最新文章

- Java new 背着我们干了什么
- 【单例深思】单例与序列化
- 【单例深思】静态内部类实现详解
- 【单例深思】双重检测锁与Java内存模型
- 【单例深思】懒汉式改进版与内置锁

博主专栏



Java单例深思

阅读量：10626 7 篇

个人分类

POJ	83篇
算法	10篇
Java	21篇
Android	3篇
Reflect as You Work	1篇

展开

归档

2017年4月	19篇
2016年1月	7篇
2015年11月	1篇
2015年5月	3篇
2015年4月	2篇

展开

0
1

热门文章

【Android学习笔记】 点击穿透（Click Through）
阅读量：7590

【单例深思】枚举实现单例原理
阅读量：7258

Winform 自定义TabControl实现浏览器标签
阅读量：4613

WebBrowser（IE）与 JS 相互调用
阅读量：4253

在C#客户端用HTTP上传文件到Java服务器
阅读量：3670

最新评论

2015年书单
ls0111： 作者是前端还是后端 --！

Java new 背着我们干了什么
qw59794464： 赞！

【单例深思】枚举实现单例原理
qq_27605885： mark

使用Inno SetUp脚本打包W...
Insert_day： 自己写脚本，控制起来容易些！


2015年书单
Insert_day： thanks

联系我们



请扫描二维码联系客服
✉ webmaster@csdn.net
☎ 400-660-0108
👤 QQ客服 🗣 客服论坛

0
1

[关于](#) [招聘](#) [广告服务](#) [网站地图](#)
©2018 CSDN版权所有 京ICP证09002463号
 百度提供搜索支持

经营性网站备案信息
网络110报警服务
中国互联网举报中心
北京互联网违法和不良信息举报中心

0
1