#### Java并发编程

##### 之单例模式：枚举类型

问题描述

JDK 8

Java中“单例模式”有以下几种实现方式：

1. 饿汉式
2. 懒汉式
3. 双重检查加锁（线程安全）
4. 静态内部类
5. 单元素的枚举类型

由单元素的枚举类型实现的单例模式由JVM保证其线程安全性，且可以规避序列化/反序列化操作对单例模式的破坏

解决

public class SingletonClass {  
 public static void main(String[] args) {  
 ProductSingleton singleton = ProductSingleton.*INSTANCE* .setId(1001L)  
 .setName("Product A")  
 .setDescription("Product A description")  
 .setPrice(1.0D);  
 System.*out*.println(JSON.*toJSONString*(singleton)); // INSTANCE.  
 }  
}  
  
  
*/\*\*  
 \* 单元素的枚举类型已经成为实现Singleton的最佳方式  
 \* —— 《Effective Java》  
 \*/*enum ProductSingleton {  
 *INSTANCE*;  
  
 */\*\*  
 \* Java Bean  
 \*/* private Long id;  
 private String name;  
 private String description;  
 private Double price;  
  
 */\*\*  
 \* 机器代码：生成的Getter、Setter方法（链式调用）.  
 \** ***@return*** *\*/* public Long getId() {  
 return id;  
 }  
  
 public ProductSingleton setId(Long id) {  
 this.id = id;  
 return this;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public ProductSingleton setName(String name) {  
 this.name = name;  
 return this;  
 }  
  
 public String getDescription() {  
 return description;  
 }  
  
 public ProductSingleton setDescription(String description) {  
 this.description = description;  
 return this;  
 }  
  
 public Double getPrice() {  
 return price;  
 }  
  
 public ProductSingleton setPrice(Double price) {  
 this.price = price;  
 return this;  
 }  
}

笔记

1. 单元素的枚举类型已成为实现单例的最佳方式

□