单例模式

1.单例模式(不适用于集群环境)分为

- 饿汉式
- 懒汉式

主要步骤:

- 创建一个实例的属性(用于后面获取实例调用)
- 构造方法私有化(是核心步骤,只有将构造方法私有化了,就是不对外公开了,这时才能控制住外部程序new的操作)
- 静态的公开的getInstance的方法(全局唯一的用来获取实例对象的方法)

两种模式主要区别是类实例的生成时机
饿汉式:类装载的时候就创建
懒汉式:实例将要被使用的时候创建
2.总结:
懒汉式: 体现了延迟加载思想(Lazy Load),缓存思想(getInstance方法),线程不安全
饿汉式:线程安全的(虚拟机保证只会装载一次),但是不具备延迟加载特性
可以看出以上的两种方式都是有缺点的
3.升华
3.1解决懒汉式线程安全问题

首先可能会想到使用synchronized 修饰getInstance方法即可,但是这样会引出另外的问题--访问速度降低.而且每次都要判断实例是否存在.

3.2 双重检查加锁

为了解决上述问题我们可以使用"双重检查加锁"

即

```
public class Singleton {
    /**
     * 对保存实例的变量添加 volatile 的修饰
     */
    private volatile static Singleton instance = null;
    private Singleton(){
    }
    public static  Singleton getInstance(){
        //先检查实例是否存在，如果不存在才进入下面的同步块
        if(instance == null){
            //同步块，线程安全地创建实例
            synchronized(Singleton.class){
                //再次检查实例是否存在，如果不存在才真正地创建实例
                if(instance == null){
                    instance = new Singleton();
                }
            }
        }
        return instance;
    }
}
```

这样可以大幅度的减少synchronized带来的访问速度慢的问题,但是并未解决根本问题

3.3 静态初始化器

先简单地看看类级内部类相关的知识。

- 什么是类级内部类?
  简单点说，类级内部类指的是，有 static 修饰的成员式内部类。如果没有 static 修饰的成员式内部类被称为对象级内部类。
- 类级内部类相当于其外部类的 static 成分，它的对象与外部类对象间不存在依赖关系，因此可直接创建。而对象级内部类的实例，是绑定在外部对象实例中的。
- 类级内部类中，可以定义静态的方法。在静态方法中只能够引用外部类中的静态成员方法或者成员变量。
- 类级内部类相当于其外部类的成员，只有在第一次被使用的时候才会被装载。

再来看看多线程缺省同步锁的知识。

大家都知道，在多线程开发中，为了解决并发问题，主要是通过使用 synchronized 来加互斥锁进行同步控制。但是在某些情况中，JVM 已经隐含地为您执行了同步，这些情况下就不用自己再来进行同步控制了。这些情况包括：

- 由静态初始化器（在静态字段上或 static{} 块中的初始化器）初始化数据时
- 访问 final 字段时
- 在创建线程之前创建对象时
- 线程可以看见它将要处理的对象时

(注:TODO上面的最后一条不懂)

所以我们可以在饿汉式中的静态本例属性放在一个静态内部类中

当调用getInstance方法时再去通过该内部类加载生成实例(延迟加载,并且是线程安全的)

3.4 终极

按照《高效 Java 第二版》中的说法：单元素的枚举类型已经成为实现 Singleton 的最佳方法。

```
public enum Singleton {
    /**
     * 定义一个枚举的元素，它就代表了 Singleton 的一个实例
     */
    uniqueInstance;

    /**
     * 示意方法，单例可以有自己的操作
     */
    public void singletonOperation(){
        //功能处理
    }
}
```

使用枚举来实现单实例控制会更加简洁，而且无偿地提供了序列化的机制，并由 JVM 从根本上提供保障，绝对防止多次实例化，是更简洁、高效、安全的实现单例的方式。

- Java 的枚举类型实质上是功能齐全的类，因此可以有自己的属性和方法。
- Java 枚举类型的基本思想是通过公有的静态 final 域为每个枚举常量导出实例的类。
- 从某个角度讲，枚举是单例的泛型化，本质上是单元素的枚举。