**单例模式**

单例模式有以下特点：

　　1、单例类只能有一个实例。

　　2、单例类必须自己创建自己的唯一实例。

　　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，而且自行实例化并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、显卡的驱动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。

懒汉式单例类.在第一次调用的时候实例化自己

但是以上懒汉式单例的实现没有考虑线程安全问题，它是线程不安全的，并发环境下很可能出现多个Singleton实例，要实现线程安全，有以下三种方式，都是对getInstance这个方法改造，保证了懒汉式单例的线程安全，

1、在getInstance方法上加同步

2、双重检查锁定

3、静态内部类

//饿汉式单例类.在类初始化时，已经自行实例化

饿汉式在类创建的同时就已经创建好一个静态的对象供系统使用，以后不再改变，所以天生是线程安全的。

饿汉式和懒汉式区别

从名字上来说，饿汉和懒汉，饿汉就是类一旦加载，就把单例初始化完成，保证getInstance的时候，单例是已经存在的了，而懒汉比较懒，只有当调用getInstance的时候，才回去初始化这个单例。

另外从以下两点再区分以下这两种方式：

1、线程安全：

饿汉式天生就是线程安全的，可以直接用于多线程而不会出现问题，懒汉式本身是非线程安全的，为了实现线程安全有几种写法，分别是上面的1、2、3，这三种实现在资源加载和性能方面有些区别。

2、资源加载和性能：

饿汉式在类创建的同时就实例化一个静态对象出来，不管之后会不会使用这个单例，都会占据一定的内存，但是相应的，在第一次调用时速度也会更快，因为其资源已经初始化完成，而懒汉式顾名思义，会延迟加载，在第一次使用该单例的时候才会实例化对象出来，第一次调用时要做初始化，如果要做的工作比较多，性能上会有些延迟，之后就和饿汉式一样了。