1,单例设计模式。

解决的问题：就是可以保证一个类在内存中的对象唯一性。

必须对于多个程序使用同一个配置信息对象时，就需要保证该对象的唯一性。

如何保证对象唯一性呢？

1，不允许其他程序用new创建该类对象。

2，在该类创建一个本类实例。

3，对外提供一个方法让其他程序可以获取该对象。

步骤：

1，私有化该类构造函数。

2，通过new在本类中创建一个本类对象。

3，定义一个公有的方法，将创建的对象返回。

//饿汉式

class Single//类一加载，对象就已经存在了。

**{**

private static Single s **=** **new** Single**();**

private Single**(){}**

public static Single getInstance**()**

**{**

**return** s**;**

**}**

**}**

//懒汉式

class Single2//类加载进来，没有对象，只有调用了getInstance方法时，才会创建对象。

//延迟加载形式。

**{**

private static Single2 s **=** **null;**

private Single2**(){}**

public static Single2 getInstance**()**

**{**

**if(**s**==null)**

s **=** **new** Single2**();**

**return** s**;**

**}**

**}**