# 单例设计模式的要点：

1. 某个类只能有一个实例。

构造器私有化。

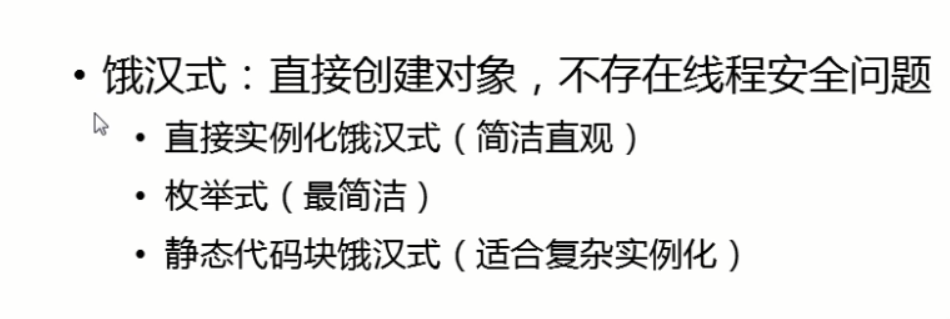
1. 它必须自行创建这个实例。

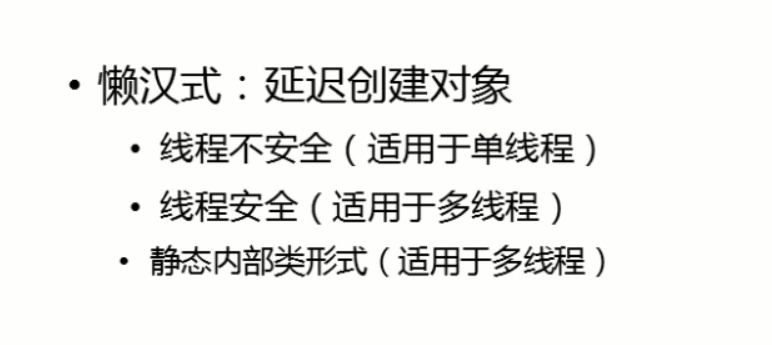
含有一个该类的静态变量来保存这个唯一的实例。

1. 它必须能向整个系统提供这个实例。

对外提供获取该实例对象的方式：

直接暴露。 用静态变量的get 方法获取。





## 饿汉式：

Class persion{

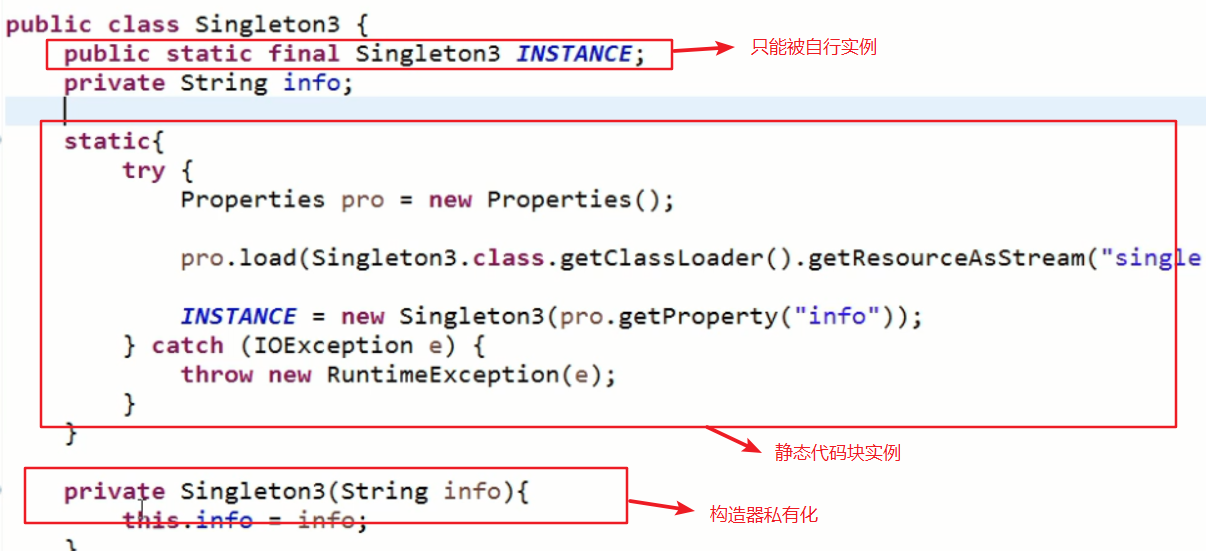
Public static final persion INNER;

Static {

INNER=new persion();

Private persion(){}

}



饿汉式在类初始化时直接创建实例对象，不管你是否需要这个对象都会创建。

## 懒汉式：

Public class Singleton{

Private Singleton(){}

Private static class final Inner{

Private static final Singleton INSTATICE= new Singleton();

}

Public static Singleton getInstance(){

Return Inner.INSATICE;

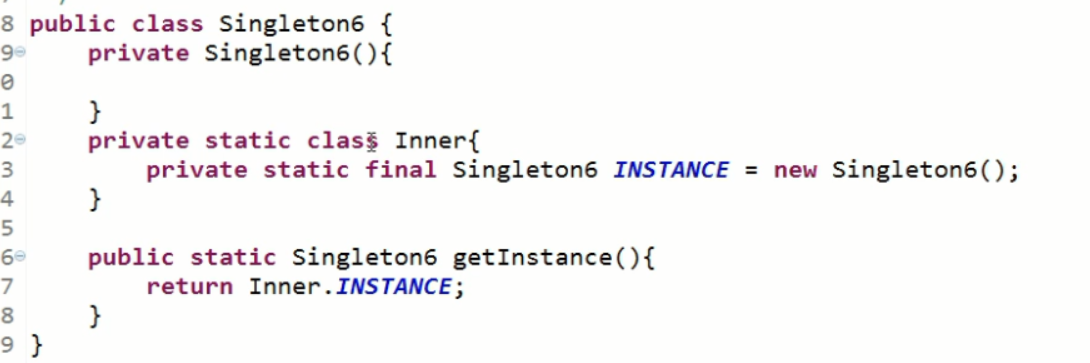
}

}

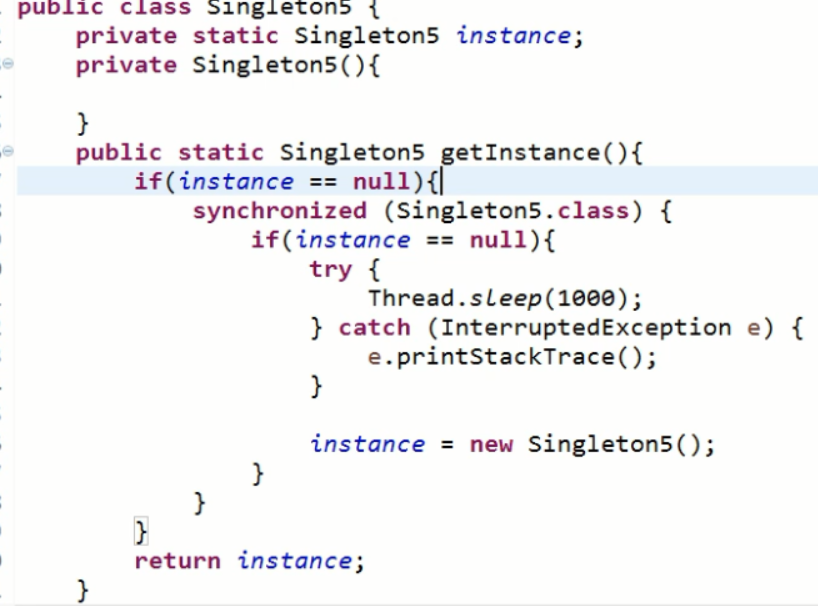
在内部类被加载和初始化时，才创建INSTATICE 实例对象。

静态内部类不会自动随着外部类的加载和初始化而初始化，它是要单独去加载和初始化的。

因为是在内部类加载和初始化时创建的，因此是线程安全的。



或者：



Public class Singleton{

Private Singleton(){}

Private static Singleton instatice;

Public static Singleton getInstatice(){

if(instatice == null){

synchronized(Singleton.class){

instatice = new Singleton();

}

}

Return instatice;

}

}