1. 單例模式的說明  
   1.1 理解  
   設計模式是在大量的實踐中總結和理論化之后優的代碼結構、編程風格、1.2 常用設計模式 ---- 23種經典的設計模式

創建型的設計模式有：單例模式、抽象工廠、工廠方法、建造模式、原型模式

行為型的設計模式有：疊代器模式、觀察者模式、模板方法、命令模式、狀態模式、

策略模式、職責鏈模式、中介模式、訪問者模式、解釋器模式、備忘錄模式

結構型的設計模式有：組合模式、外觀模式、代理模式、適配器模式、裝飾模式、橋模式、享元模式

1. 單例模式  
   2.1 要解決的問題：

所謂類的單例設計模式，就是採取一定的方法保證在整個的軟件系統中，對某個類只能存在一個對象實例。

* 1. 具體代碼的實現：

餓漢式1：

class Order{

//1.私有化類的構造器

private Order(){

}

//2.內部創建類的對象

//4.要求此對像也必須聲明為靜態的

private static Order instance = new Order();

//3.提供公共的靜態的方法，返回類的對象

public static Order getinstance(){

return instance;

}

餓漢式2：使用了靜態代碼塊

class Order{

//1.私有化類的構造器

private Order(){

}

//2.內部創建類的對象

//4.要求此對像也必須聲明為靜態的

private static Order instance = null;

static{

instance = new Order();

}

//3.提供公共的靜態的方法，返回類的對象

public static Order getinstance(){

return instance;

}

懶漢式：

class Order{

//1.私有化類的構造器

private Order(){

}

//2.聲明當前類對象，沒有初始化

//4.此對象也必須聲明為static的

private static Order instance = null;

//3.聲明public、static的返回當前類對象的方法

public static Order getinstance(){

if ( instance == null){

instance = new Order();

}

return instance;

}

* 1. 兩種方式的對比：

餓漢式：

壞處：對象加載時間過長。

好處：餓漢式是線程安全的

懶漢式：好處：延遲對象的創建。

目前的寫法壞處：線程不安全。--->到多線程內容時，再修改