**代理模式**

1. 解決的問題：



1. 舉例：

**interface** NetWork{

**public** **void** browse();

}

//被代理類

**class** Server **implements** NetWork{

@Override

**public** **void** browse() {

System.***out***.println("真實的服務器訪問網絡");

}

}

//代理類

**class** ProxyServer **implements** NetWork{

**private** NetWork work;

**public** ProxyServer(NetWork work){

**this**.work = work;

}

**public** **void** check(){

System.***out***.println("聯網之前的檢查工作");

}

@Override

**public** **void** browse() {

check();

work.browse();

}

1. 應用場景



**工廠模式**

1. 解決的問題

实现了创建者与调用者的分离，即将创建对象的具体过程屏蔽隔离起来，达到提高灵活性的目的。

1. 具體模式

簡單工廠模式： 用來生產同一等級結構中的任意產品。（對於增加新的產品,需要修改已有產品）

工廠方法模式： 用來生產同一等級結構中的固定產品。（支持增加任意產品

抽象工廠模式： 用來生產不同品族的全部產品。（對於增加新，無能為力;支持增加產品族。）