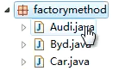
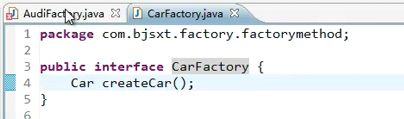
工厂方法模式可以修正简单工厂模式违反了开闭原则的问题

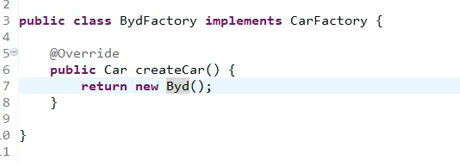


还是简历一个CarFactory 这个是接口 不是类



【实际上 可以看到 CarFactory和具体的实现类解耦合 仅仅和Car这个接口有关联了】

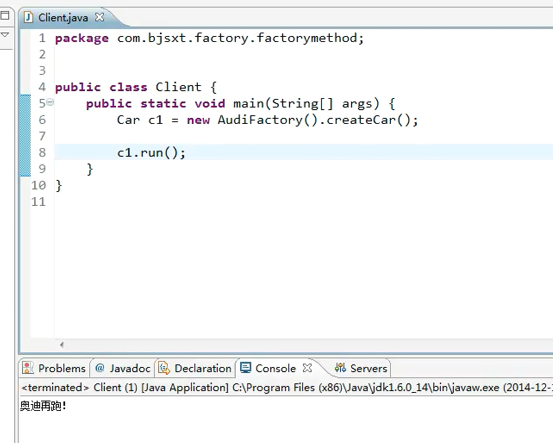
如果建立奥迪车 ---- 先建立奥迪车的工厂



【实际上可以看懂啊 工厂方法模式 就是在简单工厂的第二种实现模式上面 扩展出来的 你不是不满足开闭么 那就每一个方法都独立到一个类 然后这些类向上抽象出CarFactory】

现在的调用方式

客户端需要奥迪 通过奥迪的工厂区new

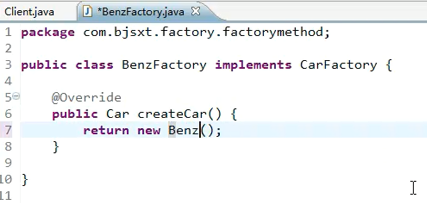


这样客户端和具体的工厂类 和 Car产生耦合

现在要增加一个新的车 

我现在不需要修改已有的代码 ------ CarFactory哪里都不需要修改

直接增加一个BenzFactory即可



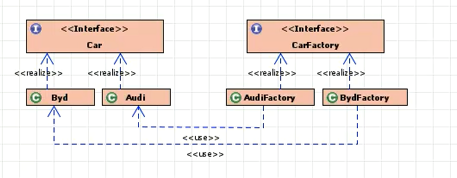
【虽然Client增加了依赖比简单工厂多了 但是 满足了项目的开闭原则 ---- 我认为知道了更多的底层细节 反而违背了迪米特原则】

这就是工厂方法模式

和简单工厂类似 就是修正了不能满足开闭原则的问题

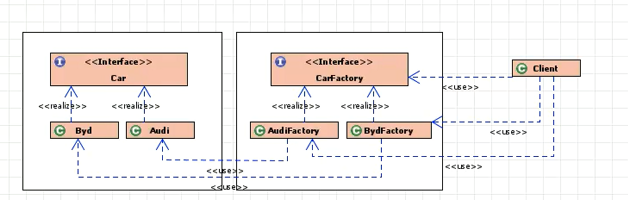
工厂方法模式虽然符合了开闭原则 ----- 类变多了

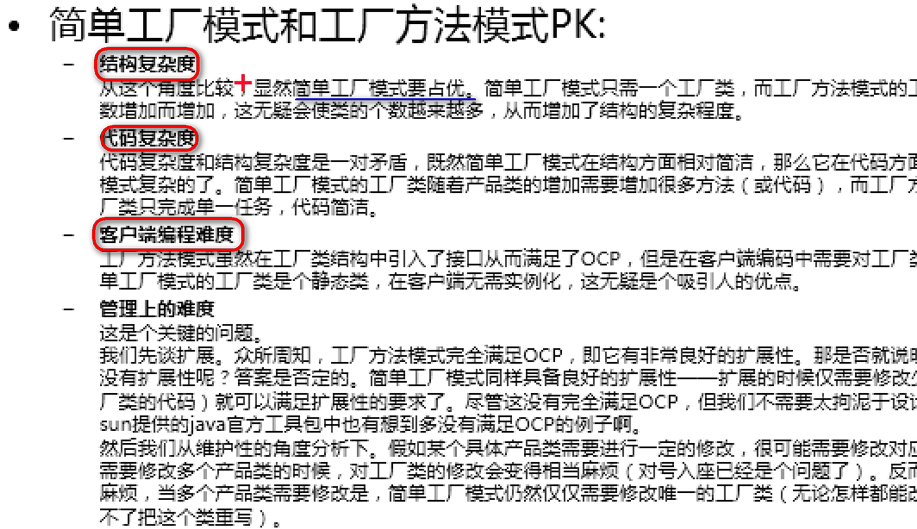
项目开发 简单工厂多一些



画一下uml

画一个矩形框就说明这个是一个继承 或者 实现的体系 对外透明





结构复杂度 客户端 和 管理上的难度 都是简单工厂占优

代码复杂度 工厂方法模式占优

