## 自问自答：

谈谈面向对象吧？封装、继承、多态

一：抽象封装

答：嗯...（思考几秒）那我就举个例子吧：就用面向对象思维封装“老张开车去东北”吧。

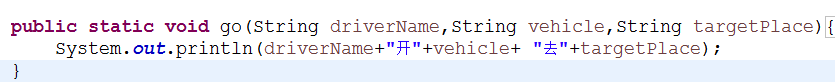
最原始的，就用一mian方法，打印：

|  |
| --- |
| System.***out***.println(**"老张开车去东北！"**); |

下一步封装，比如，把人物“老张”，交通工具“车”，目的地“东北”，抽取出来，提取成变量：

|  |
| --- |
| String driverName = **"老张"**; String vehicle = **"车"**; String targetPlace = **"东北"**; System.***out***.println(driverName+**"开"**+vehicle+ **"去"**+targetPlace); |

进一步封装，可以封装成一个方法：



这还不够面向对象，面向对象拿到一个问题，首先要考虑问题域里面有哪些对象。考虑类：一般问题域里面的名词，就是对象，或者对象的属性（不可脱离具体的应用环境），动词是方法。比如这个问题里面：

隐藏封装（降低耦合度）：属性定义为private 私有的，对外提供get、set方法，也是一种隐藏和封装，比如不想让别人修改某个属性，把set方法去掉就可以。比如可以在get方法里添加判断，看你是否有权利get该属性。

司机Driver：

属性：name

方法：drive（开车）

车Car ：

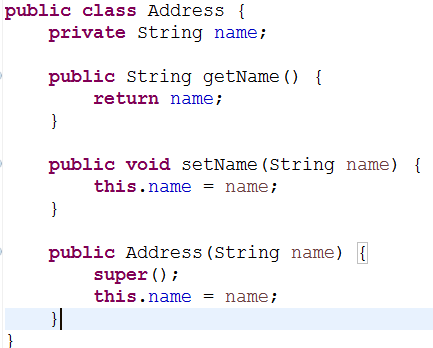
方法: go()

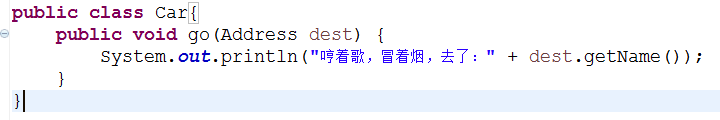
地址Address

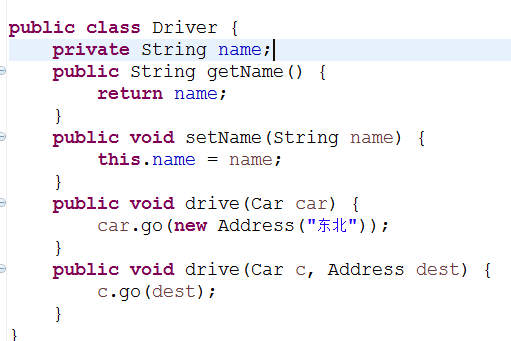
属性：name

要考虑 类和类之间的关系：司机和车之间是有关系的，司机可以开车，可以把车定义为司机的一个属性（这辆车就属于这个司机），也可以把车传个司机的开车方法drive(Car c)（司机可以开任何车）；

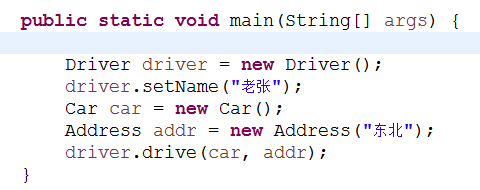
代码：



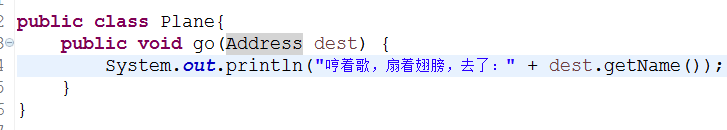




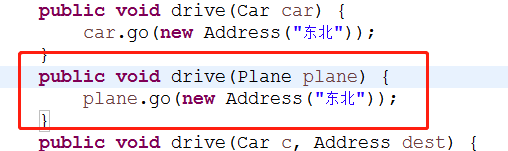
测试类：



现在如果不想开着车去，想开飞机，怎么办呢？再写个类：飞机Plane



司机再加个drive的重载方法：

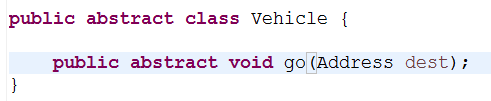


这样就可以了。这样的问题是可扩展性不好，随着交通工具的增多，司机类的drive方法需要继续重载，可不可以让司机只有一个drive方法，就能drive所有的交通工具呢？

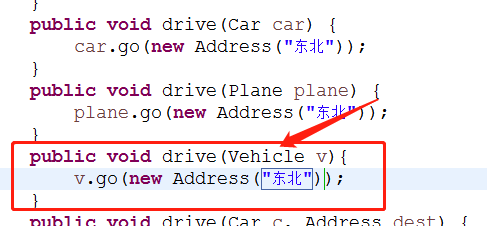
这就要用到继承了。

二：继承

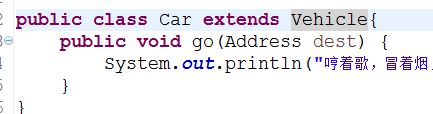
可以。写个交通工具类Vehicle，这个类是抽象的：

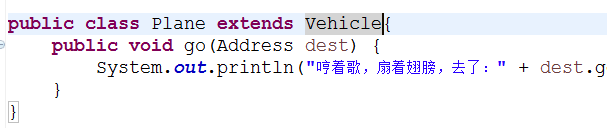


然后司机类里面，drive方法传入Vehicle交通工具，

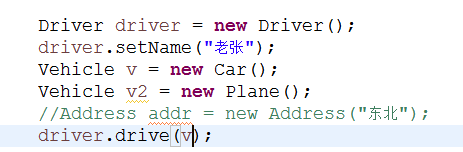


然后让Car和Plane都继承Vehicle，重写go方法：





测试类：



这样在用户使用的时候，传入是Car，就调用Car的go方法，传入的是Plane，就调用Plane的go方法，这就叫**多态**。

发生多态的三个条件：要有继承，要有重写，父类引用指向子类对象

多态的耦合性比较高，父类的修改会影响子类，需要慎重使用

多态的好处：**可扩展性高了，这就是面向对象的精华**

如果再添加其他交通工具，只需要添加类，然后继承Vehicle类就好了。

接口和抽象类的区别：

如果你脑子里有一个具体的事物，比如交通工具，就设计为抽象类。

如果你脑子里没有具体的事物，只有一类事物，或者几类事物的特征，比如说，会飞的、会跑的，就定义为接口，runnable、Moveable，比如上例中的Vehicle，就可以设计成Moveable，

然后里面有个方法，run(), 然后Car、Plane都实现这个接口即可。