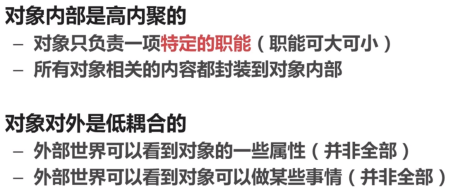
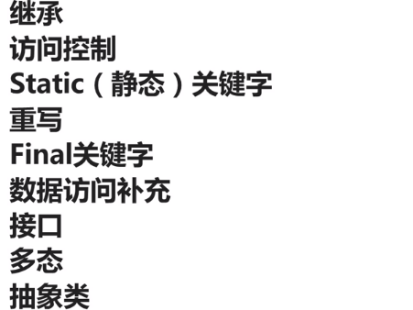
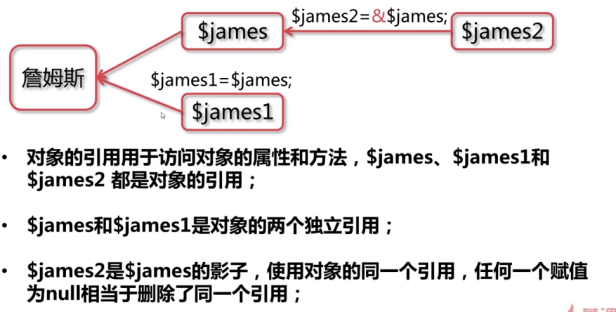
# 目标



# \*&



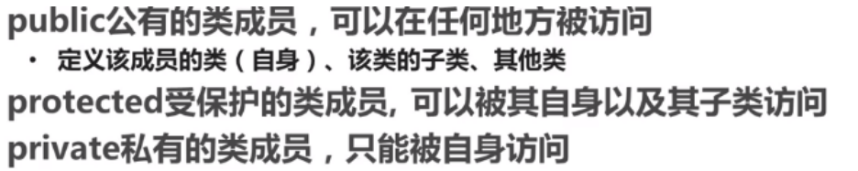
# 继承

子类对象可以访问父类

父类对象不可以访问子类

一次只可以拼接一个类，但是可以在父类中再继承加上类

# 权限



**//外面里面只能使用public方法  
//protected与private都是只能在类的里面进行访问  
//protected是在自身与子类中可以进行访问  
//private是在自身类中可以访问**

# Static

例子是如果一个对象进行修改了属性值，另外一个对象不会被修改，也就是属性值没有被保存，所以应该使用static，静态类对象设置被保存之后，

这个类多有对应的对象都发生了变化。相当于引用传值

普通的就是值传递

**// \* 1. 静态属性用于保存类的公有数据,使得类的对象static属性一致  
// \* 2. 静态方法里面只能访问静态属性  
// \* 3. 静态成员不需要实例化对象就可以访问  
// \* 4. 类内部，可以通过self或者static关键字访问自身的静态成员  
// \* 5. 可以通过parent关键字访问父类的静态成员  
// \* 6. 可以通过类名称在外部访问类的静态成员**

# Final

**\* 重写和Final  
\* 1. 子类中编写跟父类完全一致的方法可以完成对父类方法的重写  
\* 2. 对于不想被任何类继承的类可以在class之前添加final关键字  
\* 3. 对于不想被子类重写（overwrite, 修改）的方法，可以在方法定义前面添加final关键字**

# 数据访问

**\* 1. parent关键字可以用于调用父类被重写的类成员  
\* 2. self关键字可以用于访问类自身的成员方法，也可以用于访问自身的静态成员和类常量；不能用于访问类自身的属性；访问类常量时不用在常量名称前面加$符号  
\* 3. static关键字用于访问类自身定义的静态成员，访问静态属性时需要在属性名前面添加$符号**

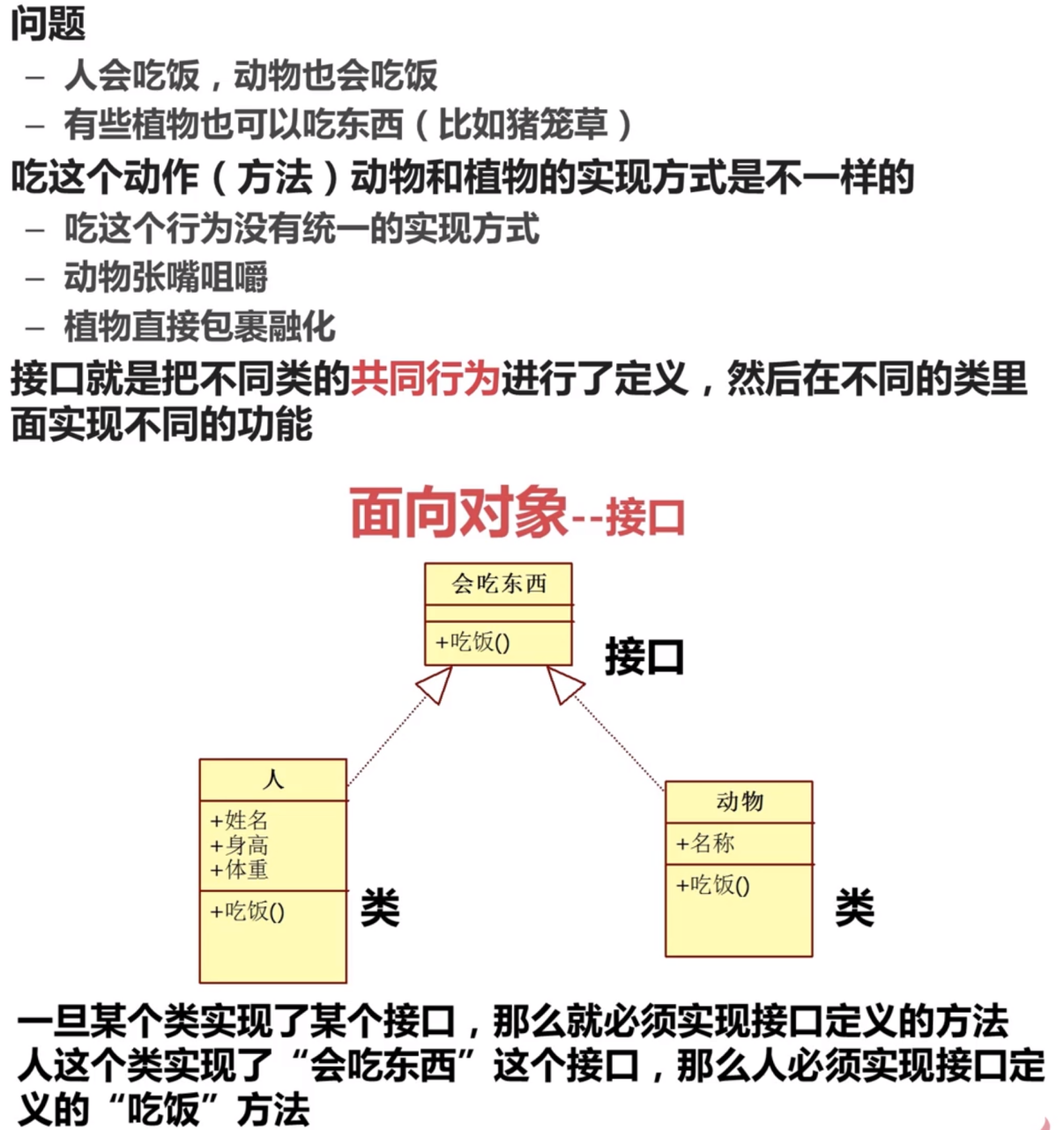
# 接口

## 说明

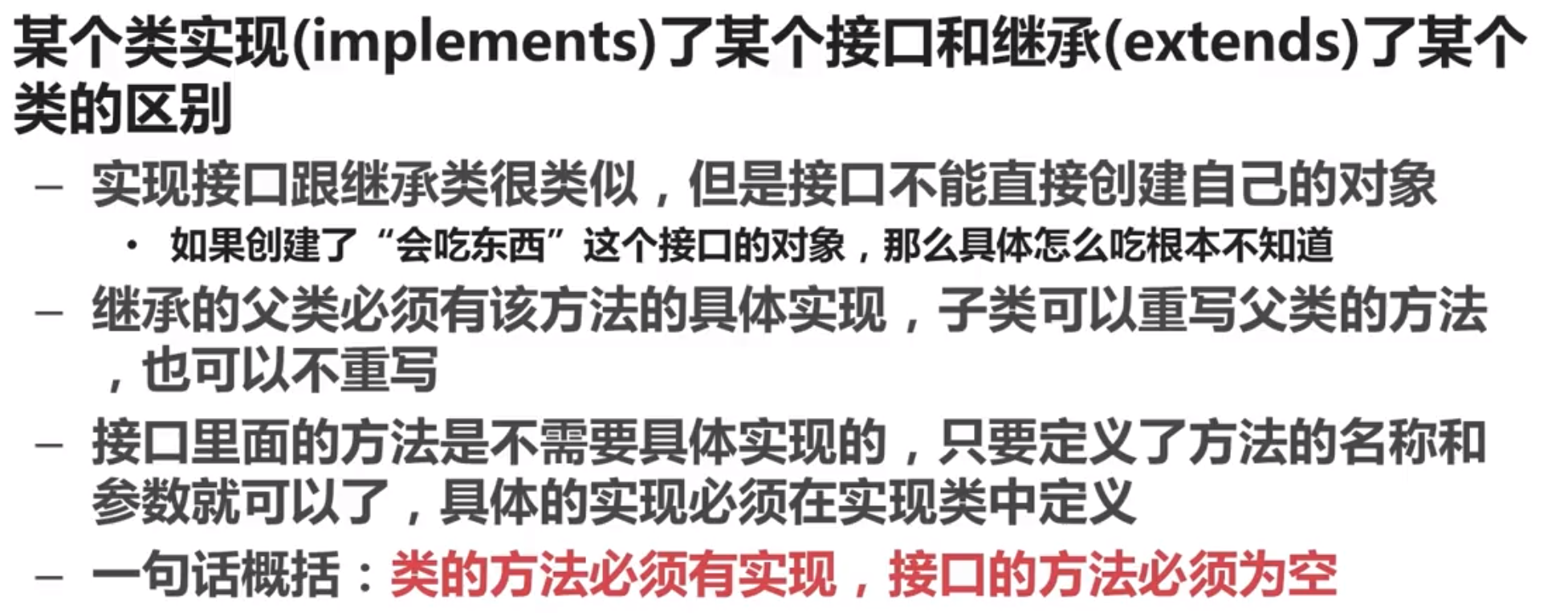
因为不同的类生成的对象实现接口定义的方法都有自己的一套规则

所以就有了接口，

接口定义了类要实现的方法

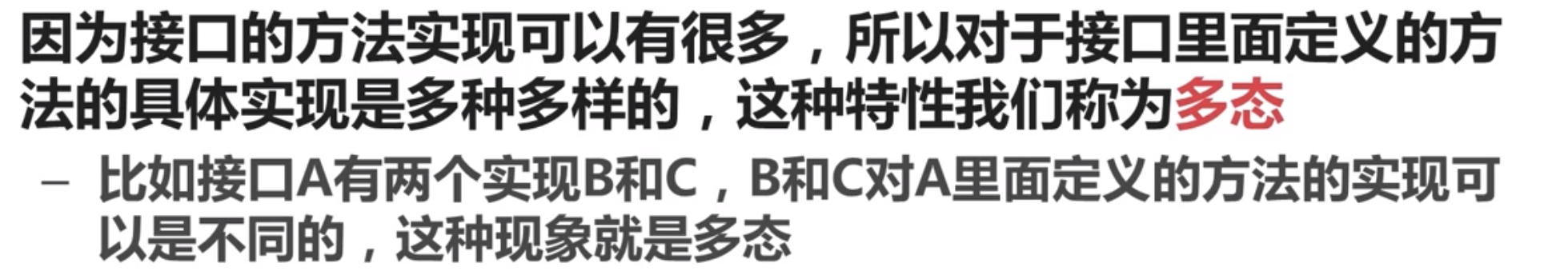


## 接口继承区别



# 多态

使用使用同一个接口，不同的类实现的方法不一样

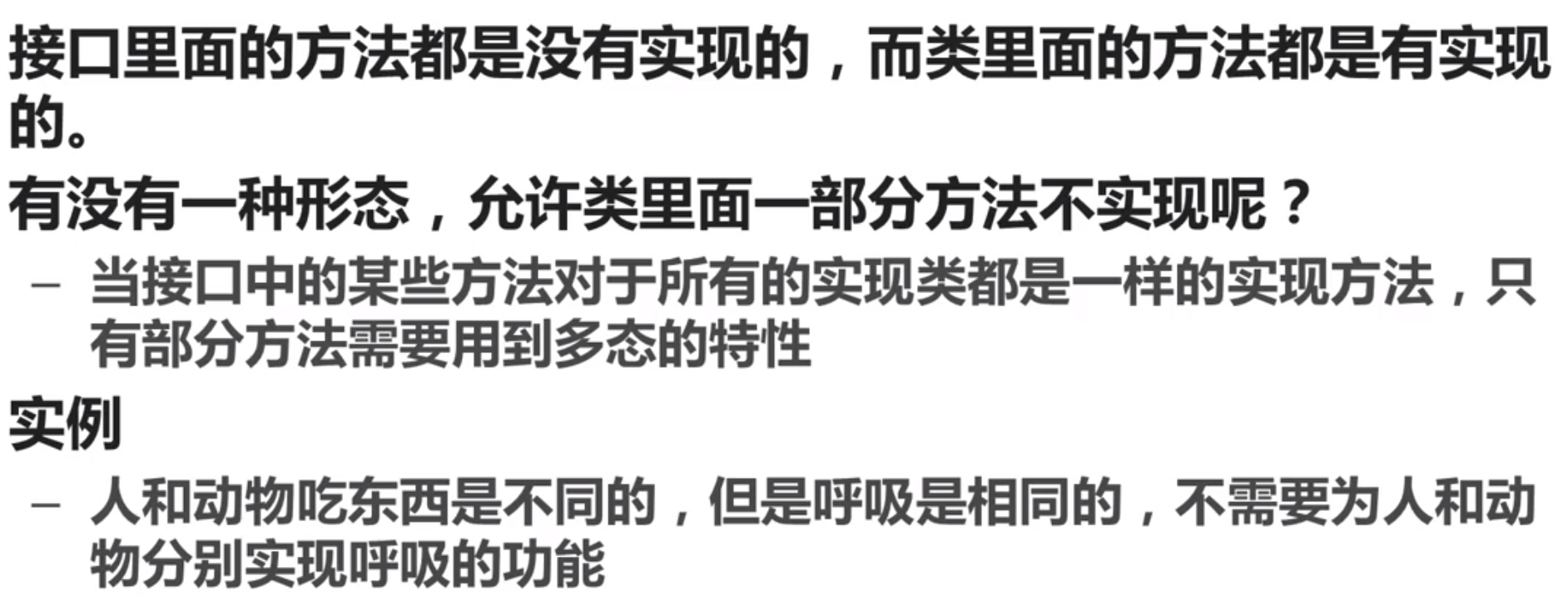


# 抽象类

继承是全部不实现

接口是全部实现

抽象类是部分实现



**\* 抽象类  
\* 1. 抽象类允许类里面的部分方法暂时没有具体实现，这些方法我们成为抽象方法  
\* 2. 一旦类里面有抽象方法，这个类就必须是抽象类  
\* 3. 抽象类跟接口一样，不能直接实例化为对象**