面向对象编程–Java篇

面向对象万物皆对象
Java中的三大特性封装、继承、多态

封装

1. 隐藏实现细节，提供公共的访问方式
2. 好处：
   A:隐藏实现细节，提供公共的访问方式
   B:提高代码的复用性
   C:提高代码的安全性
3. 设计原则
   把不想让外界知道的实现细节给隐藏起来，提供公共的访问方式
4. private是封装的一种体现。
   封装：类，方法，private修饰成员变量

继承

1. 把多个类中相同的成员给提取出来定义到一个独立的类中。然后让这多个类和该独立的类产生一个关系，
   这多个类就具备了这些内容。这个关系叫继承。
2. Java中如何表示继承呢?格式是什么呢?
   A:用关键字extends表示
   B:格式：
   class 子类名 extends 父类名 {}
3. 继承的好处：
   A:提高了代码的复用性
   B:提高了代码的维护性
   C:让类与类产生了一个关系，是多态的前提
4. 继承的弊端：
   A:让类的耦合性增强。这样某个类的改变，就会影响其他和该类相关的类。
   原则：低耦合，高内聚。
   耦合：类与类的关系

内聚：自己完成某件事情的能力
B:打破了封装性

5. Java中继承的特点
   A:Java中类只支持单继承
   B:Java中可以多层(重)继承(继承体系)
6. 继承的注意事项：
   A:子类不能继承父类的私有成员
   B:子类不能继承父类的构造方法，但是可以通过super去访问
   C:不要为了部分功能而去继承
7. 什么时候使用继承呢?
   A:继承体现的是：is a的关系。
   B:采用假设法
8. Java继承中的成员关系
   1). 成员变量
   a:子类的成员变量名称和父类中的成员变量名称不一样，这个太简单
   b:子类的成员变量名称和父类中的成员变量名称一样，这个怎么访问呢?
   子类的方法访问变量的查找顺序：
   在子类方法的局部范围找，有就使用。
   在子类的成员范围找，有就使用。
   在父类的成员范围找，有就使用。
   找不到，就报错。
   B:构造方法
   a:子类的构造方法默认会去访问父类的无参构造方法
   是为了子类访问父类数据的初始化
   b:父类中如果没有无参构造方法，怎么办?
   子类通过super去明确调用带参构造
   子类通过this调用本身的其他构造，但是一定会有一个去访问了父类的构造
   让父类提供无参构造
   C:成员方法
   a:子类的成员方法和父类中的成员方法名称不一样，这个太简单
   b:子类的成员方法和父类中的成员方法名称一样，这个怎么访问呢?
   通过子类对象访问一个方法的查找顺序：
   在子类中找，有就使用
   在父类中找，有就使用
   找不到，就报错

多态

1. 同一个对象在不同时刻体现出来的不同状态。
2. 多态的前提：
   A:有继承或者实现关系。
   B:有方法重写。
   C:有父类或者父接口引用指向子类对象。

```
多态的分类：
    a:具体类多态
        class Fu {}
        class Zi extends Fu {}

        Fu f = new Zi();
    b:抽象类多态
        abstract class Fu {}
        class Zi extends Fu {}

        Fu f = new Zi();
    c:接口多态
        interface Fu {}
        class Zi implements Fu {}

        Fu f = new Zi();
```

3. 多态中的成员访问特点
   A:成员变量
   编译看左边，运行看左边
   B:构造方法
   子类的构造都会默认访问父类构造
   C:成员方法
   编译看左边，运行看右边
   D:静态方法
   编译看左边，运行看左边
   为什么?
   因为成员方法有重写。
4. 多态的好处：
   A:提高代码的维护性(继承体现)
   B:提高代码的扩展性(多态体现)
5. 多态的弊端：
   父不能使用子的特有功能。
   现象：
   子可以当作父使用，父不能当作子使用。
6. 多态中的转型
   A:向上转型
   从子到父
   B:向下转型
   从父到子