今日学习目标

* 能够理解面向对象的思想
* 能够明确类与对象关系
* 能够根据类的定义定义学生类
* 能够创建对象,并且使用对象访问类中的成员 对象.成员变量名 对象.成员方法(..)
* 能够完成手机类的练习
* 能够理解对象的内存图
* 能够说出成员变量和局部变量的区别
* 能够运用private关键字
* 能够用封装的思想定义一个标准类
* 能够说出this可以解决的问题
* 能够运用构造方法创建对象

# 面向对象

## 面向对象思想概述及举例

* 概述
  + 我们回想一下，这几天我们完成一个需求的步骤：首先是搞清楚我们要做什么，然后在分析怎么做，最后我们再代码体现。一步一步去实现，而具体的每一步都需要我们去实现和操作。这些步骤相互调用和协作，完成我们的需求。
  + 在上面的每一个具体步骤中我们都是参与者，并且需要面对具体的每一个步骤和过程，这就是面向过程最直接的体现。
  + 那么什么是面向过程开发呢? 面向过程开发，其实就是面向着具体的每一个步骤和过程，把每一个步骤和过程完成，然后由这些功能方法相互调用，完成需求。
  + 面向过程的代表语言：C语言
  + 当需求单一，或者简单时，我们一步一步去操作没问题，并且效率也挺高。可随着需求的更改，功能的增多，发现需要面对每一个步骤很麻烦了，这时就开始思索，能不能把这些步骤和功能在进行封装，封装时根据不同的功能，进行不同的封装，功能类似的封装在一起。这样结构就清晰了很多。用的时候，找到对应的类就可以了。这就是面向对象的思想。
* 面向对象的思想
  + 面向对象是基于面向过程的编程思想.
  + 面向过程:强调的是每一个功能的步骤
  + 面向对象:强调的是对象,然后由对象去调用功能
* 特点
  + 是一种更符合我们思考习惯的思想
  + 可以将复杂的事情简单化
  + 将我们从执行者变成了指挥者
    - 角色发生了转换
* \*举例
  + 买电脑
    - 面向过程：我要买电脑--我要明确买电脑的意义--上网查对应的参数信息--去中关村买电脑--讨价还价--买回电脑
    - 面向对象：我要买电脑--班长去给我买电脑--买回电脑
  + 洗衣服
    - 面向过程：把衣服脱下来--找一个盆--放点洗衣粉--加点水--浸泡10分钟--揉一揉--清洗衣服--拧干--晾起来
    - 面向对象：把衣服脱下来--打开全自动洗衣机--扔衣服--按钮--晾起来

## 类与对象及其使用

* \*类与对象关系
  + 我们学习编程语言，就是为了模拟现实世界的事物，实现信息化。比如：去超市买东西的计费系统，去银行办业务的系统。
  + 我们如何表示一个现实世界事物呢：
    - 属性 就是该事物的描述信息
    - 行为 就是该事物能够做什么
    - 举例： 学生事物
  + 我们学习的Java语言最基本单位是类，所以我们就应该把事物用一个类来体现。
  + 类：是一组相关的**属性**和**行为**的集合。
  + 对象：是该类事物的具体体现
  + 举例：
    - 类 学生
    - 对象 班长就是一个对象
* \*类的定义
  + 现实世界的事物
    - 属性 事物的描述信息
    - 行为 事物能做什么
  + Java中用class描述事物也是如此
    - 成员变量 就是事物的属性
    - 成员方法 就是事物的行为
  + 定义类其实就是定义类的成员(成员变量和成员方法)
    - 成员变量:和以前定义变量是一样的，只不过位置发生了改变。在类中，方法外。
    - 成员方法:和以前定义方法是一样的，只不过把static去掉，面向对象进阶在详细讲解static的作用。

**public class**  Student {

//成员变量

//姓名

String name;

//年龄

**int** age;

//成员方法

//学习的方法

**public void** study() {

System.***out***.println("好好学习，天天向上");

}

//吃饭的方法

**public void** eat() {

System.***out***.println("学习饿了要吃饭");

}

}

* \*对象的创建及其使用
  + 创建对象：

类名对象名 = new 类名();

* + 使用对象访问类中的成员:
    - 对象名.成员变量
    - 对象名.成员方法

**public class**  Demo01Student {

**public static void**  main(String[] args) {

//创建对象格式：类名对象名 = new 类名();

Student s = **new** Student();

System.***out***.println("s:"+s); //cn.itcast.demo02.Student@100363

//直接输出成员变量值

System.***out***.println("姓名："+s.name); //null

System.***out***.println("年龄："+s.age); //0

System.***out***.println("----------");

//给成员变量赋值

s.name = "赵丽颖";

s.age = 18;

//再次输出成员变量的值

System.***out***.println("姓名："+s.name); //赵丽颖

System.***out***.println("年龄："+s.age); //18

System.***out***.println("----------");

//调用成员方法

s.study();

s.eat();

}

}

* \*手机类练习
  + 定义手机类

public class Phone {

//品牌

String brand;

//价格

int price;

//颜色

String color;

//打电话

public void call(String name) {

System.*out*.println("给"+name+"打电话");

}

//发短信

public void sendMessage() {

System.*out*.println("群发短信");

}

}

* + 手机类的测试类

**public class**  Demo02Phone {

**public static void**  main(String[] args) {

//创建对象

Phone p = **new** Phone();

//输出成员变量值

System.***out***.println("品牌："+p.brand);//null

System.***out***.println("价格："+p.price);//0

System.***out***.println("颜色："+p.color);//null

System.***out***.println("------------");

//给成员变量赋值

p.brand = "锤子";

p.price = 2999;

p.color = "棕色";

//再次输出成员变量值

System.***out***.println("品牌："+p.brand);//锤子

System.***out***.println("价格："+p.price);//2999

System.***out***.println("颜色："+p.color);//棕色

System.***out***.println("------------");

//调用成员方法

p.call("唐嫣");

p.sendMessage();

}

}

## 对象的内存图

* 定义Phone类

public class Phone {

String brand;

int price;

String color;

public void call(String name) {

System.*out*.println("给"+name+"打电话");

}

public void sendMessage() {

System.*out*.println("群发短信");

}

}

* 一个对象的内存图

**public class**  Demo01Phone {

**public static void**  main(String[] args) {

Phone p = **new** Phone();

System.***out***.println(p.brand + "---" + p.price + "---" + p.color);

p.brand = "锤子";

p.price = 2999;

p.color = "棕色";

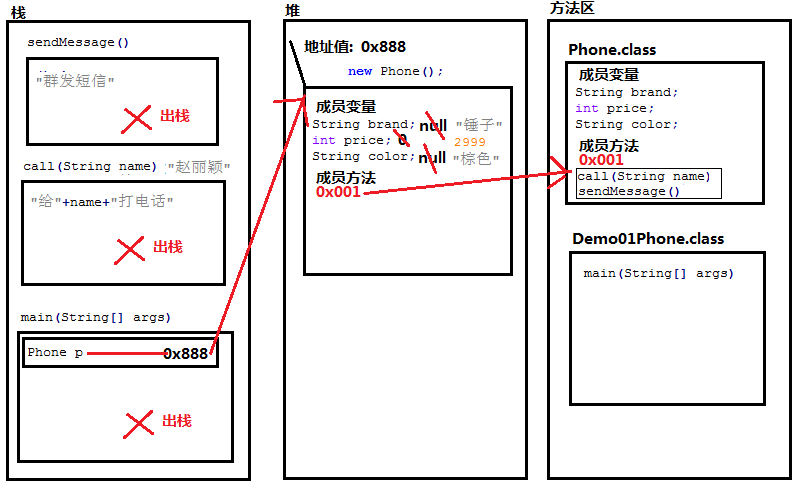
System.***out***.println(p.brand + "---" + p.price + "---" + p.color);

p.call("林青霞");

p.sendMessage();

}

}



* 两个引用，指向同一个对象的内存图

**public class**  Demo03Phone {

**public static void**  main(String[] args) {

Phone p1 = **new** Phone();

p1.brand = "OPPO";

p1.price = 2999;

p1.color = "白色";

System.***out***.println(p1.brand+"---"+p1.price+"---"+p1.color);

Phone p2 = p1;

p2.brand = "魅族";

p2.price = 1999;

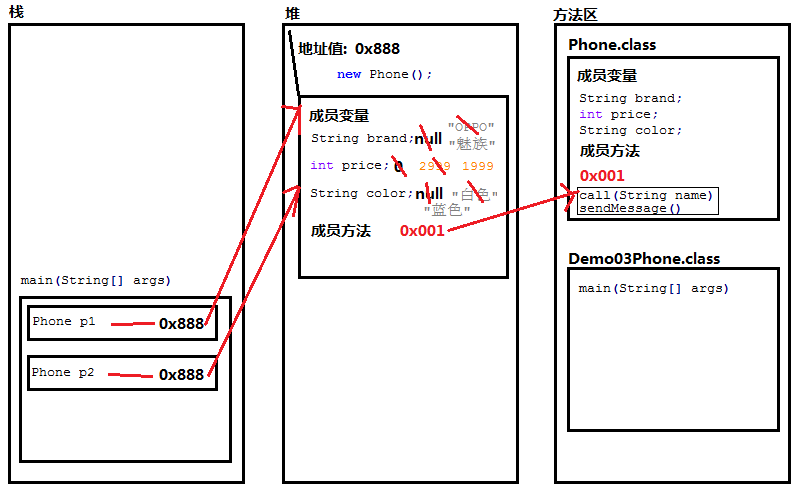
p2.color = "蓝色";

System.***out***.println(p1.brand+"---"+p1.price+"---"+p1.color);

System.***out***.println(p2.brand+"---"+p2.price+"---"+p2.color);

}

}



## 成员变量和局部变量的区别

* 在类中的位置不同
  + 成员变量：类中，方法外
  + 局部变量：方法中或者方法声明上(形式参数)
* 在内存中的位置不同
  + 成员变量：堆内存
  + 局部变量：栈内存
* 生命周期不同
  + 成员变量：随着对象的创建而存在，随着对象的消失而消失
  + 局部变量：随着方法的调用而存在，随着方法的调用完毕而消失
* 初始化值的不同
  + 成员变量：有默认值
  + 局部变量：没有默认值。必须先定义，赋值，最后使用

**public class**  DemoVariable {

**int** x;

**public void** show() {

**int** y = 0;

System.***out***.println(x);

System.***out***.println(y);

}

}

## 封装

* private关键字由来及其使用
  + private关键字：
    - 是一个权限修饰符。
    - 可以修饰成员(成员变量和成员方法)
    - 被private修饰的成员只在本类中才能访问。

**public class** Student {

String name;

//int age;

**private int** age;

**public void** setAge(**int** a) {

**if**(a<0 || a>200) {

System.***out***.println("你给的年龄有误");

}**else** {

age = a;

}

}

**public int** getAge() {

**return** age;

}

**public void** show() {

System.***out***.println("姓名是："+name+",年龄是："+age);

}

}

**public class** DemoStudent {

**public static void** main(String[] args) {

//创建学生对象

Student s = **new** Student();

s.show();

s.name = "柳岩";

//s.age = 18;

//s.age = -18;

//s.setAge(-18);

s.setAge(28);

s.show();

}

}

* + private最常见的应用：
    - 把成员变量用private修饰
    - 提供对应的getXxx()/setXxx()方法
    - 一个标准的案例的使用
* 成员变量私有的标准代码

**Public class** Student {

**private** String name;

**private int** age;

**public void** setName(String n) {

name = n;

}

**public** String getName() {

**return** name;

}

**public void** setAge(**int** a) {

age = a;

}

**public int** getAge() {

**return** age;

}

}

**Public class** DemoStudent {

**Public static void** main(String[] args) {

//创建对象

Student s = **new** Student();

System.***out***.println(s.getName()+"---"+s.getAge());

s.setName("赵丽颖");

s.setAge(18);

System.***out***.println(s.getName()+"---"+s.getAge());

}

}

* 封装的概述和好处
  + 封装概述
    - 是面向对象三大特征之一
    - 是面向对象编程语言对客观世界的模拟，客观世界里成员变量都是隐藏在对象内部的，外界无法直接操作和修改。就像刚才说的年龄。
  + 封装原则：
    - 将不需要对外提供的内容都隐藏起来。
    - 把属性隐藏，提供公共方法对其访问。
    - 成员变量private，提供对应的getXxx()/setXxx()方法
  + 好处：
    - 通过方法来控制成员变量的操作，提高了代码的安全性
    - 把代码用方法进行封装，提高了代码的复用性

## this关键字

* this关键字的由来
  + this:代表所在类的对象引用
  + 记住：方法被哪个对象调用，this就代表那个对象
* this关键字的使用
  + 局部变量隐藏成员变量(局部变量和成员变量重名)

**public class**  Student {

**private** String name;

**private int** age;

**public void** setName(String name) {

//name = name;

**this**.name = name;

}

**public** String getName() {

**return** name;

}

**public void** setAge(**int** age) {

//age = age;

**this**.age = age;

}

**public int** getAge() {

**return** age;

}

}

**public class**  DemoStudent {

**public static void**  main(String[] args) {

//创建对象

Student s = **new** Student();

System.***out***.println(s.getName()+"---"+s.getAge());

s.setName("柳岩");

s.setAge(18);

System.***out***.println(s.getName()+"---"+s.getAge());

}

}

## 构造方法

* 作用
  + 给对象的数据进行初始化
* 格式
  + 方法名与类名相同
  + 没有返回值类型，连void都没有
  + 没有具体的返回值

修饰符 构造方法名(参数列表){

}

* 注意事项
  + 如果你不提供构造方法，系统会给出默认构造方法
  + 如果你提供了构造方法，系统将不再提供
  + 构造方法也是可以重载的

**Public class** Student {

**private** String name;

**private int** age;

/\*

public Student() {

System.out.println("构造方法");

}

\*/

**public** Student() {}

**public** Student(String name) {

**this**.name = name;

}

**public** Student(**int** age) {

**this**.age = age;

}

**public** Student(String name,**int** age) {

**this**.name = name;

**this**.age = age;

}

**public void** show() {

System.***out***.println(name+"---"+age);

}

}

**Public class** DemoStudent {

**Public static void** main(String[] args) {

//如何调用构造方法呢?

//通过new关键字调用

//格式：类名对象名 = new 构造方法(...);

Student s = **new** Student();

s.show();

//public Student(String name)

Student s2 = **new** Student("赵丽颖");

s2.show();

//public Student(int age)

Student s3 = **new** Student(18);

s3.show();

//public Student(String name,int age)

Student s4 = **new** Student("赵丽颖",18);

s4.show();

}

}

* 标准学生类代码编写和测试
  + 类
    - 成员变量
    - 构造方法
      * 无参构造方法
      * 带参构造方法
    - 成员方法
      * getXxx()
      * setXxx()
  + 给成员变量赋值的方式
    - 无参构造方法+setXxx()
    - 带参构造方法
  + 练习
    - 学生类(老师讲)

**public class**  Student {

//成员变量

**private** String name;

**private int** age;

//构造方法

**public** Student() {}

**public** Student(String name,**int**age) {

**this**.name = name;

**this**.age = age;

}

//成员方法

**public void** setName(String name) {

**this**.name = name;

}

**public** String getName() {

**return** name;

}

**public void** setAge(**int**age) {

**this**.age = age;

}

**public int** getAge() {

**return** age;

}

}

**public class** DemoStudent {

**public static void**  main(String[] args) {

//无参+setXxx()

Student s = **new** Student();

s.setName("柳岩");

s.setAge(18);

System.***out***.println(s.getName()+"---"+s.getAge());

//带参构造

Student s2 = **new** Student("赵丽颖",18);

System.***out***.println(s2.getName()+"---"+s2.getAge());

}

}

* + - 手机类(学生练)
    - 汽车类(学生练) Car 属性:颜色,品牌,价钱… 功能:行驶