# 回顾:

1. 类(相同属性和方法的对象的集合,是一个模板),对象(现实世界中存在的可以被描述得事物)
2. 两者关系:类是对象的类型,对象是类的实例
3. 如何从一个类得到一个对象? 类名 对象名=new 类名()
4. 类的基本结构:

public class 类名{

//常见的属性和方法

}

1. 方法 有4种类型

//1.无参数无返回值

public void 方法名(){

//方法体-实现->每个方法都有自己特定功能->一个方法做一件事情

}

//2.无参数有返回值

public 返回值类型 方法名(){

//方法体-实现->每个方法都有自己特定功能->一个方法做一件事情

return 值;

}

//3.有参数无返回值

public void 方法名(类型 参数名,类型 参数名,类型 参数名){

//方法体-实现->每个方法都有自己特定功能->一个方法做一件事情

}

//4.有参数有返回值

public 返回值的类型 方法名(类型 变量名,类型 变量名,类型 变量名){

//方法体-实现->每个方法都有自己特定功能->一个方法做一件事情

return 值;

}

调用:

1. 无参数无返回值

对象名.方法名();

1. 无参数有返回值

有返回值的话需要定义一个变量去接收

类型 变量名=对象名.方法名();;;;(变量的类型根据方法返回值的类型而定)

1. 有参数无返回值

对象名.方法名(1,3,5);//参数个数和参数类型要对应

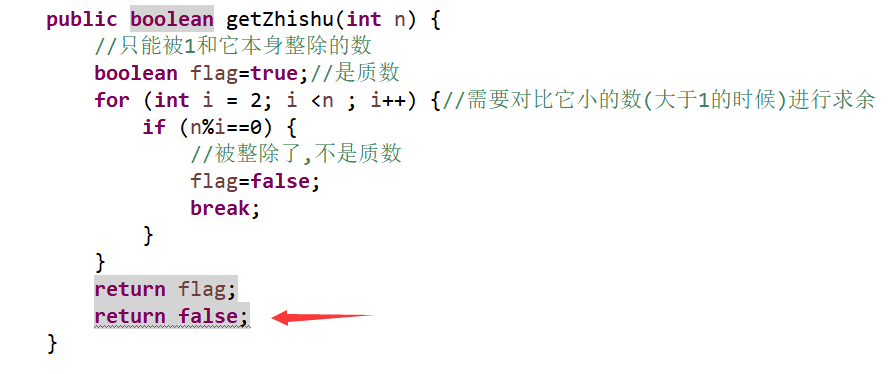
1. 有参数有返回值

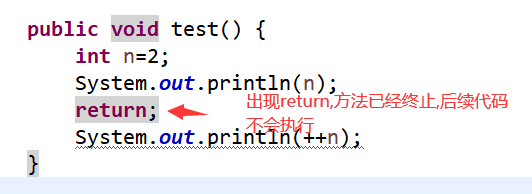
类型 变量名=对象名.方法名(1,3,5)

6.return

一个方法中的return最多只能执行一次,

return意味着这个方法的执行已经终止;这个方法的后续代码将不会执行





# 面向对象特征：

## 封装:

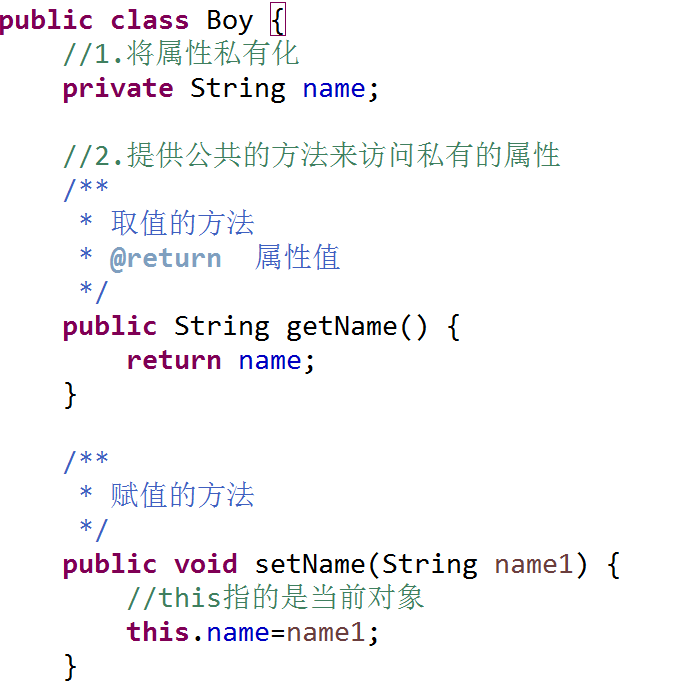
**封装**的特性能够让服务提供者把它服务的**细节隐藏掉**，你只需要**提交请求与传递它需要的参数**，它就会给你返回结果，而这个结果是如何产生的，经过了多少复杂运算，经过多少次数据读取，你都不用管，只要它给你结果就好了。

## 封装的实现:

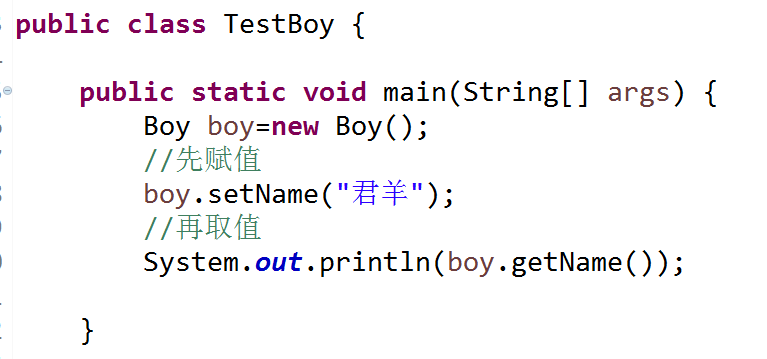
将属性私有化，属性的名字不对外公开，但是外部还要访问私有的属性，我们可以提供公共的方法来访问私有的属性;

**步骤:**

1. 将成员属性私有化 ②提供公共的setter/getter方法



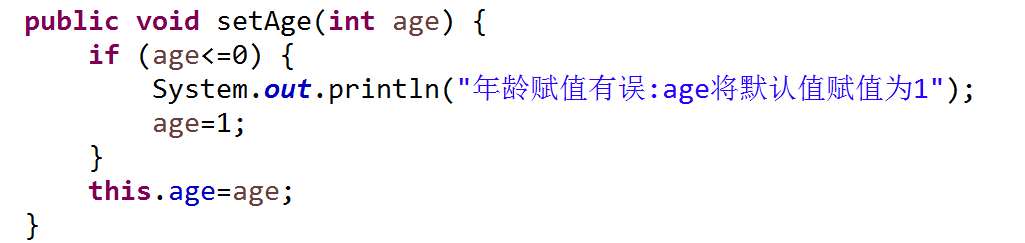
测试类:

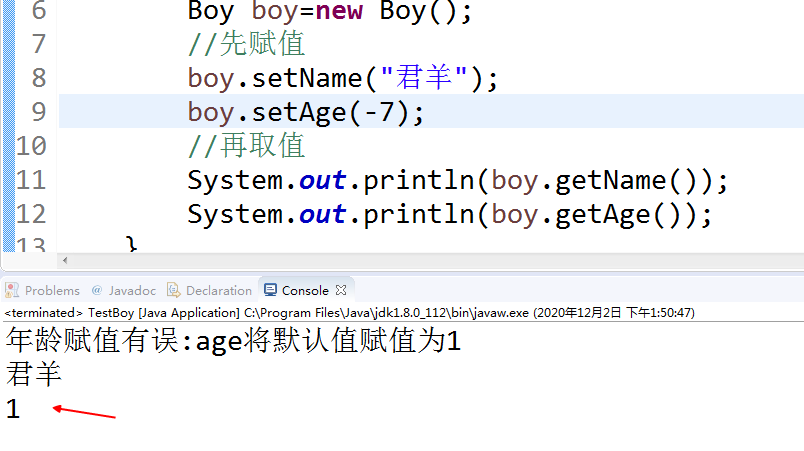


## 封装的作用:

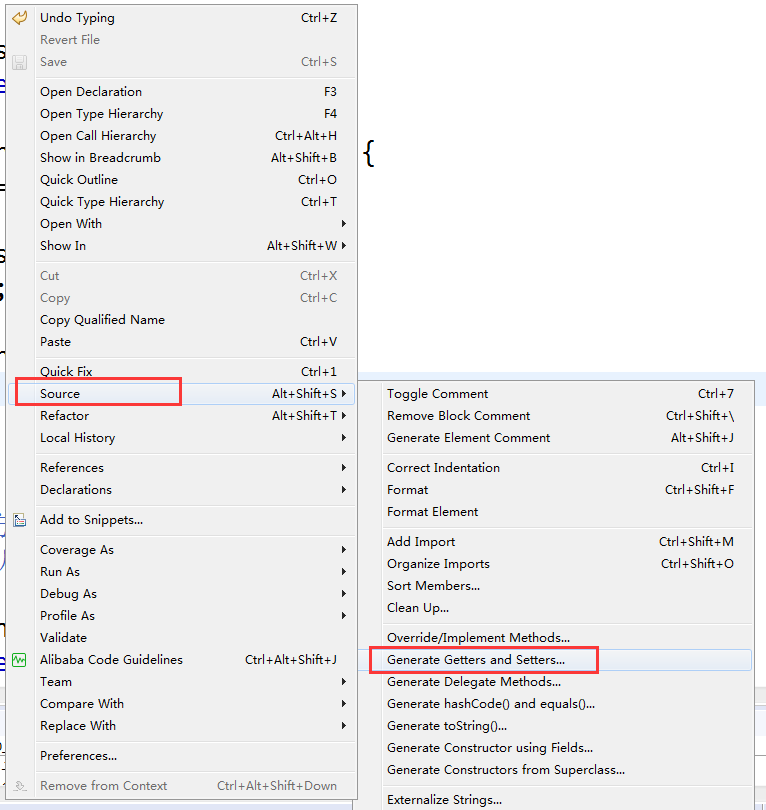
解决了数据的安全性问题,提供安全性,在setter中可以对外部传递进来的参数进行判断

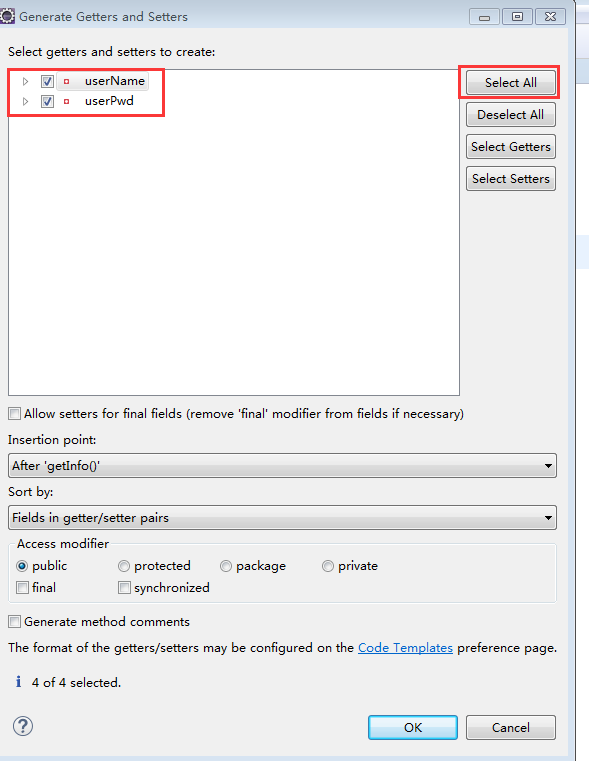
赋值(取值)过程中加一些判断





快捷键:右键->sources...





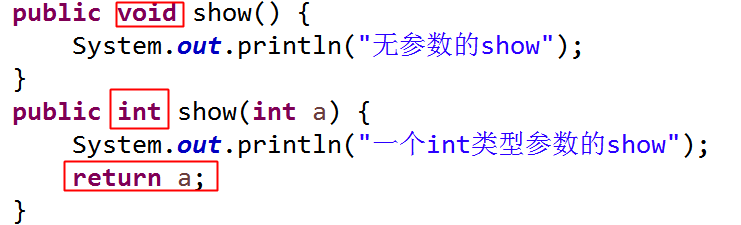
# [面试题]方法的重载:

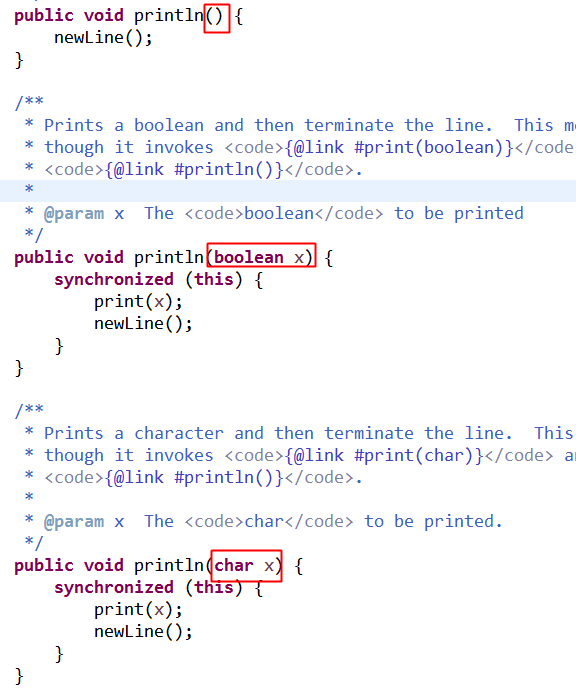
定义:在同一个类中方法名相同,参数项不同,这时候方法之间的关系称为重载.

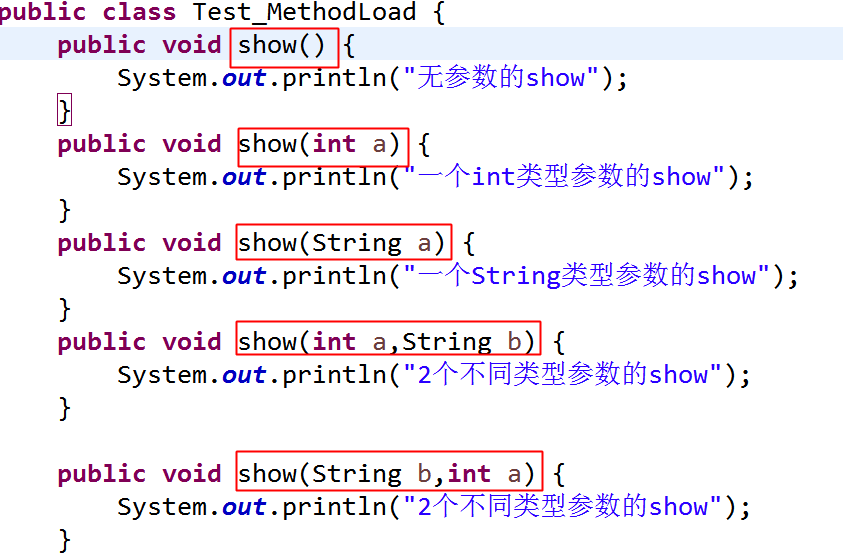
1. 参数个数不同
2. 参数个数相同,数据类型或者类型顺序不同

重载的好处:方法名的复用性.构造方法就是一个典型的方法重载

**注意**:方法的重载规则与返回值无关







## 课堂练习：

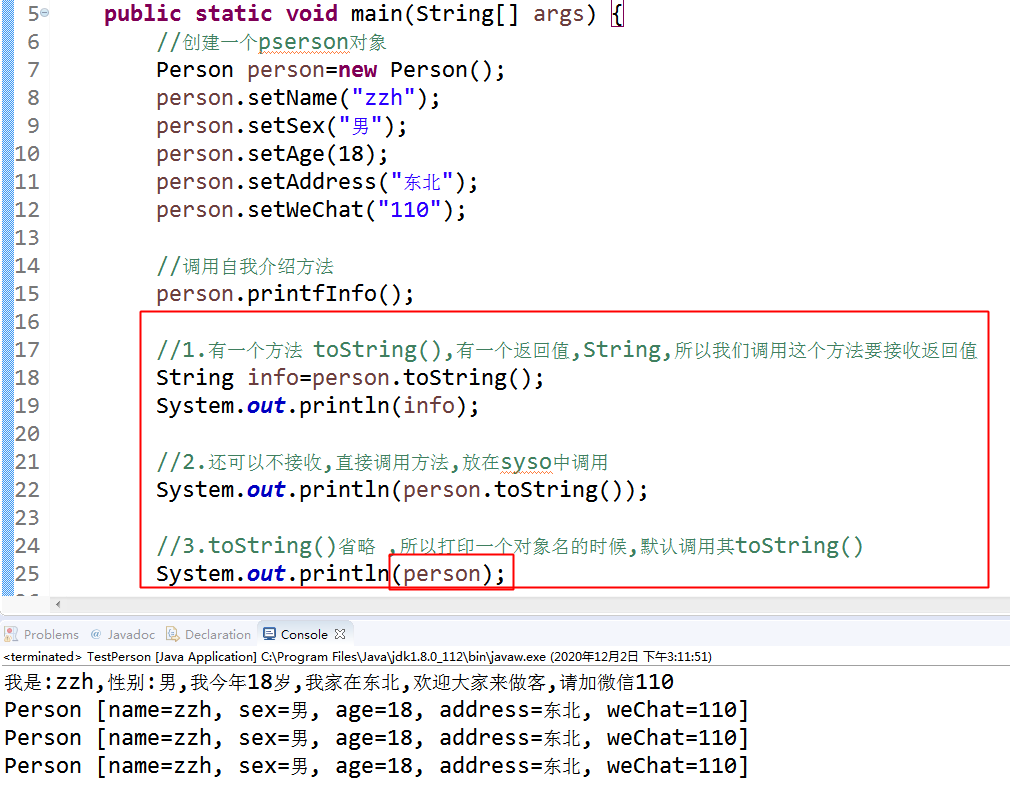
定义类，在类中定义自我介绍的方法，**输出**个人信息;包含姓名，性别，年龄，联系方式等.

分析: 将属性定义好，私有化，封装 ; 方法中将个人信息打印出来:



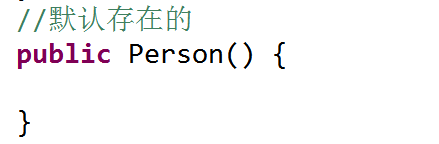
快捷键生成toString():右键:source->generate toString

测试类:



# 3.构造方法：

正常情况下，定义一个类，这个类内部会有一个构造方法(构造器);



## 规则：

1 构造方法的方法名和类名一样

2 无返回类型

判断以下方法是不是构造方法:



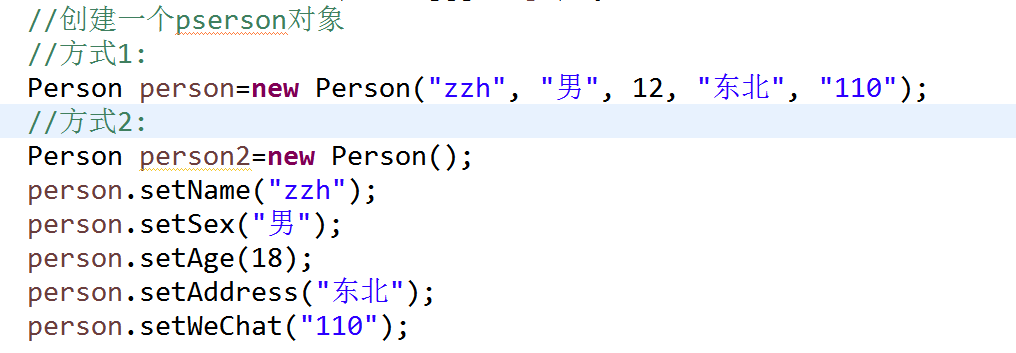
## 作用：

1. 用来实例化对象，调用无参也可以有参构造方法
2. 调用构造方法的时候同时希望对成员属性赋值，这个时候需要调用有参数的构造方法

## 语法:

类名 引用名= new 构造方法()

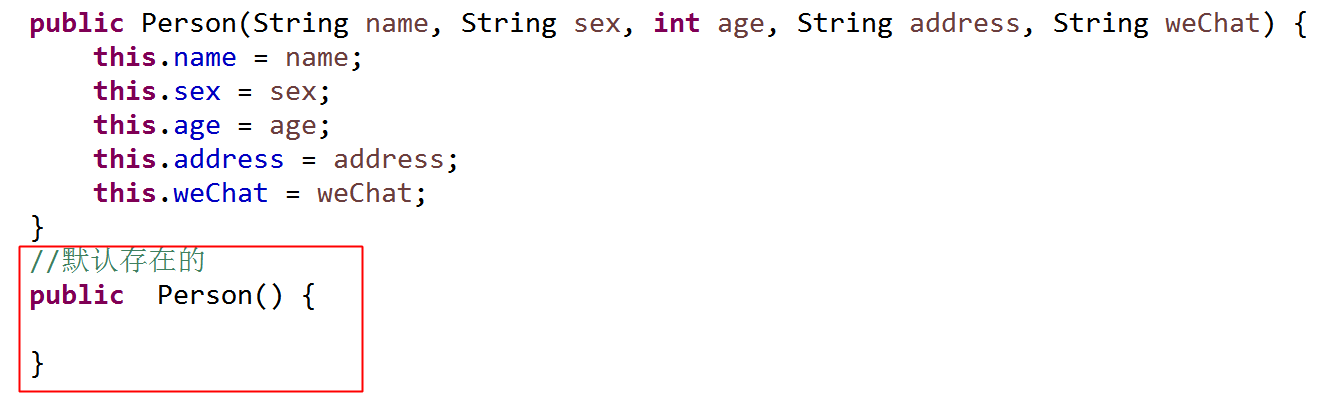
类名 引用名= new 构造方法(参数列表)



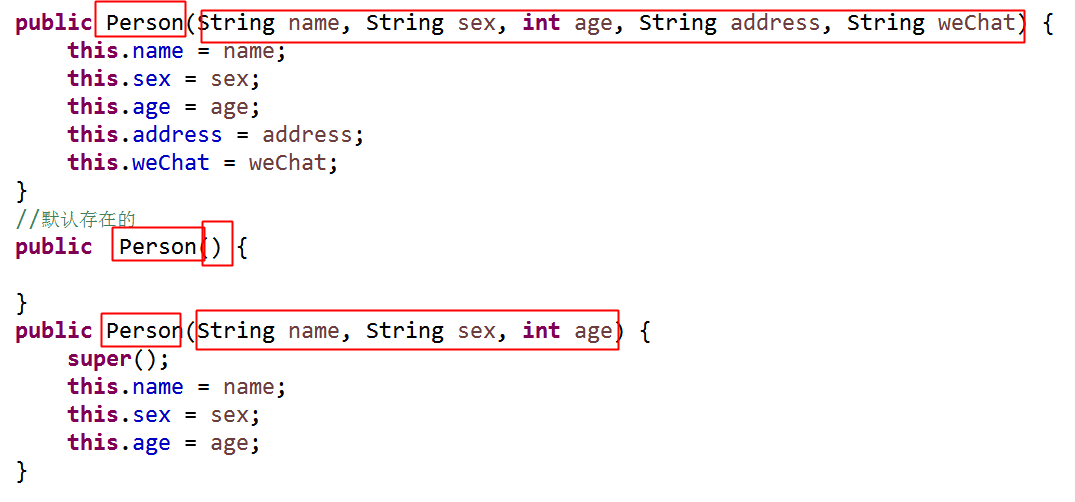
当我们在一个类中定义一个带参数构造方法的时候，那么这个类中，就有两个构造方法了?? 不对

**注意**:一个类中有一个默认的无参数构造方法,如果显式声明一个有参数构造方法,这个时候无参构造方法就被覆盖掉了,

所以,如果既要无参又要有参,那么也要将无参构造方法显式出来



## 构造方法的重载：



## 应用场景:

当我们给对象中的属性初始化操作的时候，通常会用到构造方法;

# 作业

day08