**JAVA基础**

# 面向对象

什么是对象?

万事万物皆对象，桌子，椅子，人

类

人类，鸟类，具有某种特征事物的集合-》具有某种特征对象的集合（表述一类，可以是一个人，也可以是千千万万个人） 泛指

对象具体的某一个，人类中的yasuo

人类-》public class People{}

对象/实例-》People p = new People();

动物类-》public class Animal{ }

狗实例-》Animal dog = new Animal();

猫对象->Animal cat = new Animal();

所有的类都是引用类型

所有的现实问题，我们都需要使用编程语言来描述，针对上述类与对象的定义，我们可以用java语言描述如下：

|  |
| --- |
| **public** **class** People {  // 特征-》人所具有的属性  // 名字 性别 年龄 身高 体重 种族 帅不帅 富不富 美不美....  // 8种基本数据类型 ，引用类型String 数组...  String name;// 表示人的名字 特征代表  String gender;// 性别 男 女  **int** age;// 年龄  **int** hight;// 身高 cm  **double** weight;// 体重 KG  String race;// 种族  String diamond;// 高富帅 或者 白富美  // 声明变量p1  **static** People *p1*;  **public** **static** **void** main(String[] args) {  // 从泛指的这个人类，new创造出来了一个人-》通过类创造对象  // People是类，p就是对象，我们也可以成p为实例  People p = **new** People();  // 输出这个人的名字,通过.连接符访问  System.***out***.println(p.name);  System.***out***.println(p.age);  p.name = "yasuo";  p.gender = "男";  p.age = 20;  p.hight = 180;  p.weight=70;  p.race="汉族";  p.diamond="高富帅";  System.***out***.println(p.name);  System.***out***.println(p.age);  // 声明时默认值是null  System.***out***.println(*p1*);  System.***out***.println("=====");  // nullpointexception  System.***out***.println(*p1*.name);  }  } |

## 方法

|  |
| --- |
| **public** **class** Student {  **public** String name;// 表示人的名字 特征代表  **public** String gender;// 性别 男 女  **public** **int** age;// 年龄  // 实例属性/普通属性  **public** **int** hight;// 身高 cm  // 类属性，不建议使用  **public** **static** String *race*;  // 方法代表学生具有的行为，具有一种共性  **public** **void** sleep(){  System.***out***.println("sleep");  }  **public** **int** walk(){  System.***out***.println("walk");  // 结束了方法，并返回一个 整数值  **return** 10000;  }  **public** String eat(){  System.***out***.println("eat");  **return** "吃的榴莲";  }  /\*\*  \* 实例方法  \* **@param** gameName(形參)  \* 遊戲名稱  \*/  **public** **void** playGame(String gameName){  System.***out***.println(gameName);  }  /\*\*  \* **@param** gameName(形參)  \* 遊戲名稱  \* **@param** level(形參)  \* 遊戲等級  \*/  **public** **void** playGame(String gameName, **int** level){  System.***out***.println(gameName + " " + level);  }  // 类方法  **public** **static** **void** test() {  System.***out***.println("类方法");  }  } |

### 方法的调用

|  |
| --- |
| // 將學生類导入到当前文件中，这样我们就可以操作这个学生类  **import** com.cognizant.model.Student;  /\*\*  \* 方法属性的调用都是通过实例去调用的，这里的实例是指我们创建出来的，不可能是空  \* 否则会报空指针异常  \* **@author** tzhang  \*/  **public** **class** Main {  **public** **static** **void** main(String[] args) {  // 創建一個學生實例  Student stu = **new** Student();  // 属性的访问  System.***out***.println(stu.name);  stu.name = "魏子安";  System.***out***.println(stu.name);  // 方法的访问,调用walk方法，执行walk方法体，执行结束后，  // 再回到此处，也就是调用的地方,  // 方法的执行结果就是返回了他return部分的内容  **int** returnVal = stu.walk();  // 方法的直接调用  stu.sleep();  stu.walk();  System.***out***.println(returnVal);  System.***out***.println("方法执行结束");  System.***out***.println(stu.eat());  // 按住ctrl ，移動鼠標到方法調用的名字上，單擊鼠標左鍵  // "王者榮耀"-》实参  stu.playGame("王者榮耀");  String game = "王者榮耀";  **int** lev = 1;  // game lev是实参  stu.playGame(game, lev);  // 类方法的调用，可以直接通过类.方法名调用，不需要通过实例调用  Student.*test*();  // 不建议使用，底层调用会转换为通过类的方式调用  stu.*test*();  System.***out***.println(Student.*race*);  }  } |

### 构造器

* 构造器最大的用处就是在创建对象时执行初始化，初始化类中的属性
* 如果程序员没有为Java 类提供任何构造器，则系统会为这个类提供一个无参的构造器。
* 一旦程序员提供了自定义的构造器，则系统不再提供默认的构造器。

|  |
| --- |
| // 无参构造器，如果有参数，有参构造器，参数称为形参  // 参数的来源，一般为实例属性  // 构造器的名字，就是类名  **public** Student(){    }  // 构造器重载  **public** Student(String name,String gender){  **this**.name = name;  **this**.gender = gender;    }    **public** Student(String name,String gender,**int** age,**int** height){  // 调用的是含有2个参数的构造器  **this**(name,gender);  **this**.age = age;  **this**.height = height;  } |

### 对象this的引用

this 关键字总是指向调用该方法的对象.

1.构造器中引用该构造器执行初始化的对象

2.在方法中引用调用该方法的对象

3.this用于区别属性和局部变量

注: this 不能用在有static 修饰的方法中（原因）

### 方法的重载

|  |
| --- |
| /\*\*  \* 实例方法  \* **@param** gameName(形參)  \* 遊戲名稱  \*/  **public** **void** playGame(String gameName){  System.***out***.println(gameName);  }  /\*\*  \* 与上面下面相邻的方法都是方法重载，方法重载只关注  \* 1、同一个类中  \* 2、方法名相同  \* 3、形参列表不同  \* 其他都不管（如返回值类型）  \* **@param** level  \* **@return**  \*/  **public** **int** playGame(**int** level){  **return** level;  }    /\*\*  \* **@param** gameName(形參)  \* 遊戲名稱  \* **@param** level(形參)  \* 遊戲等級  \*/  **public** **void** playGame(String gameName, **int** level){  System.***out***.println(gameName + " " + level);  } |

### 递归

|  |
| --- |
| /\*\*  \* 求阶乘的递归  \* **@param** i  \* **@return**  \*/  **private** **static** **int** factorial(**int** i){  // 考虑i<0或者i=1的情况    **if**(i==1|| i==0) {  **return** 1;  }  **return** i\**factorial*(i-1);  }  // 存在着方法调方法，一个return只能结束一个方法，所以一开始的那个return最后结束  // return return  // 3 \* factorial(2)-》3 \* 2 \* factorial(1) -> 3 \* 2 \* 1  // 4 \* factorial(3)-》4 \* 3 \* factorial(2)->4 \* 3 \* 2 \* factorial(1) // 4 \* 3 \* 2 \* 1 |

### 长度可变的参数

|  |
| --- |
| /\*\*  \*  \* **@param** subject  \* 科目  \* **@param** grade  \* 长度可变的形参（0-无穷），必须在参数的最后一个，数据类型后面加上三个点，  \* 后面跟着参数名就行,grade接收参数后是一个数组  \*/  **public** **void** score(String subject,**int**... grade) {  System.***out***.println(subject);  System.***out***.println(grade.length);  **for** (**int** i:grade){  System.***out***.print(i + " ");  }  System.***out***.println();  } |

局部变量

局部变量可分为三种：

* 1. 形参
  2. 方法局部变量

与成员变量不同的是除了形参外,其他局部变量都必须显式地初始化

|  |
| --- |
| **public** **class** Main1 {  **static** String *str* = "hello";  **public** **static** **void** main(String[] args) {  String str = "world";  System.***out***.println(str);//world  }  } |

* 局部变量仅在方法内有效，当方法执行完成时，局部变量便会自动销毁

成员变量

方法外，类内部定义

局部变量与成员变量他们定义的位置？

局部变量与成员变量他们的作用域？

引用传递

值传递（基本数据类型）

|  |
| --- |
| **public** **class** ValTran {  **public** **static** **void** main(String[] args) {  // 值传递-传递的是变量值-基本数据类型  **int** y = 20;  *baseTran*(y);  System.***out***.println(y);// 20  // 值传递（引用传递）-传递的是地址值-引用数据类型（String除外）  Student stu = **new** Student("李白","男");  *referenceTran*(stu);  System.***out***.println(stu.name);// 杜甫  }  **static** **void** baseTran(**int** y ){  y = y + 10;  }  **static** **void** referenceTran(Student stu ){  stu.name="杜甫";  }  } |

类的get，set 方法会使用Split**方法使用**

|  |
| --- |
| **public** **class** Student {  // private只能在当前类中可以访问，访问方式 .  **private** String name;// 表示人的名字 特征代表  **private** String gender;// 性别 男 女  **private** **int** age;// 年龄  // 实例属性/普通属性  **private** **int** height;// 身高 cm    **public** Student(){    }    **public** Student(String name, String gender, **int** age, **int** height) {  **super**();  **this**.name = name;  **this**.gender = gender;  **this**.age = age;  **this**.height = height;  }  // alt + shift + s  // 获取属性值  **public** String getName() {  **return** name;  }  // 修改或设置属性  **public** **void** setName(String name) {  **this**.name = name;  }  **public** String getGender() {  **return** gender;  }  **public** **void** setGender(String gender) {  **this**.gender = gender;  }  **public** **int** getAge() {  **return** age;  }  **public** **void** setAge(**int** age) {  **this**.age = age;  }  **public** **int** getHeight() {  **return** height;  }  **public** **void** setHeight(**int** height) {  **this**.height = height;  }  // toString方法输出学生基本信息  @Override  **public** String toString() {  **return** "name=" + name + ", gender=" + gender + ", age=" + age + ", height=" + height;  }  } |

|  |
| --- |
| **public** **static** **void** main(String[] args) {  Student stu = **new** Student();  stu.setName("李白");  stu.setGender("男");  stu.setAge(20);  stu.setHeight(180);  System.***out***.println(stu.toString());  // 修改学生信息  stu.setName("杜甫");  System.***out***.println(stu.toString());  // 获取年龄  System.***out***.println(stu.getAge());    // 构造器初始化数据  Student stu1 = **new** Student("李清照","女",20,180);  System.***out***.println(stu1.toString());    // String的split分割字符串，得到一个字符串数组  Scanner scanner = **new** Scanner(System.***in***);  System.***out***.println("请输入信息：姓名 性别 年龄 身高");  String str = scanner.nextLine();  String[] info = str.split(" ");  **if** (info.length == 4) {  // 字符串怎样转换为整数  String s = "123";  **int** x = Integer.*parseInt*(s);  Student stu2 = **new** Student(info[0],info[1],  Integer.*parseInt*(info[2]),Integer.*parseInt*(info[3]));  System.***out***.println(stu2.toString());  }    } |

作业：

编写一个应用程序绘制一个如下的操作菜单并实现功能(定义一个类学生表示学生)

nextLine(),默认一共有100个学生，一维数组-》一维对象（student）数组

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

| 1. 增加学生 |

| 2. 显示学生 |

| 3. 修改学生 |

| 4. 删除学生 |

| 5. 查找学生 |

| 6. 退出程序 |

| 请输入选择（1-6）： |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1、Scanner 学号 姓名 班级 年龄

2、数组-》对象数组-》Student[] 10 2=5

当前有多少位学生，一共多少页，默认显示显示第一页数据，

提供功能，输入页数时，显示指定页数的数据

3、构造器实现学生信息的初始化

4、添加学生就是赋值，删除学生就是赋值（null）

5、修改学生，遍历学生是否存在-》yes-》set方法修改学生信息

6、显示学生，遍历数组，toString方法输出学生信息

7、退出，循环次数不固定，所以使用while循环，控制while循环的准入条件实现退出

提示：

|  |
| --- |
|  |

|  |
| --- |
| /\*\*  \* 主要完成功能  \* **@author** tzhang  \*  \*/  **public** **class** Tool {  **public** **void** outputMenu() {  }  **public** **void** add(Student stu){  }  **public** **void** update() {  }  **public** **void** delete(String sno) {  }  **public** **void** selectBySno(String sno){  }  **public** **void** selectAll() {  }  } |

|  |
| --- |
| **public** **class** Demo {  // 控制主逻辑，方法流程  **public** **static** **void** main(String[] args) {    }    } |