

Java对象和类疑问解答

一个类必须有构造器吗？

非必须
没有明确写构造器，Java会自动加一个无参构造器

构造器=构造方法

构造器 (Constructor) 一般用来进行对象的初始化工作

给成员变量赋初值 (比如传递参数进来设置name、age等) 进行对象所需的其他初始化逻辑 (如分配资源、执行检测等) 也可什么都不写 (空的构造器)

构造器里可以写复杂逻辑

最常见的写法 (赋初值)

```
public class Dog {  
    String name;  
    int age;  
  
    // 构造器：用参数初始化成员变量  
    public Dog(String name, int age) {  
        this.name = name; // this表示当前对象  
        this.age = age;  
    }  
  
    public Dog(String name, int age) {  
        if (age < 0) {  
            throw new IllegalArgumentException("年龄不能为负数");  
        }  
        this.name = name;  
        this.age = age;  
        // 可以加：如链接数据库、读取配置文件等初始化动作  
    }  
  
    public class Dog {  
        String name;  
        int age;  
  
        public Dog() {  
            this.name = "无名";  
            this.age = 0;  
        }  
    }  
}
```

无参构造器 (给默认值)

构造器 (Constructor)：用来创建对象并初始化对象的成员变量 (属性)，只在new时执行一次。

构造器名字和类名必须完全一致，没有返回值 (连void都不能写)

对比普通方法

不能和类名完全相同且没有返回类型 (就是说普通方法必须有返回类型)

```
public void name() {  
    // 可以实现自己的功能  
}
```

普通方法用来实现对象的具体动作 (比如set/get、打印、计算等)，可以反复调用。

普通方法是对象创建好后，调用来完成某些具体功能的

```
private String name;  
  
public void setName(String name) { // set方法  
    this.name = name;  
}  
  
public String getName() { // get方法  
    return this.name;  
}
```

Java约定 (Bean规范)：set/get后面的第一个字母大写 (即使变量名是小写)；驼峰命名法 (camelCase)，是规范，不是强制

```
public void setName(String name) { ... }  
public String getName() { ... }
```

set和get方法 (不是构造器)，用来赋值和取值

用于安全地设置/获取属性值，有命名规范 主要用于封装，外部不能直接访问private属性，只能通过set/get间接访问或修改

```
public class Student {  
    private String name; // 私有属性  
  
    public void setName(String name) { // set方法：设置属性  
        this.name = name;  
    }  
  
    public String getName() { // get方法：获取属性  
        return this.name;  
    }  
}
```

普通方法可以有任何名字，功能不限于属性读写 用来实现实际功能，比如计算、打印、操作等名称不需要像set/get那样有固定格式

```
public void study() {  
    System.out.println(name + "在学习Java! ");  
}
```

实际代码对比

```
public class Book {  
    private String title;  
    private int price;  
  
    // set/get方法  
    public void setTitle(String title) { this.title = title; }  
    public String getTitle() { return this.title; }  
    public void setPrice(int price) { this.price = price; }  
    public int getPrice() { return this.price; }  
  
    // 普通方法  
    public void showInfo() {  
        System.out.println("书名: " + title + ", 价格: " + price);  
    }  
}
```

理解构造器和方法？

与set/get方法和普通方法 (Method) 对比

```
public class Person {  
    // 这是构造器  
    public Person() {  
        // 构造的时候执行  
    }  
  
    // 这是普通方法  
    public void sayHello() {  
        System.out.println("Hello!");  
    }  
}
```

对象变量前面的那个 class 名 (比如：Person p = new Person(); 这句中的 Person，有以下几层含义：

类型声明 (数据类型) Person p 里的 Person 表示变量 p 的数据类型，也就是 p 只能指向 Person 类类型的对象 (类似c语言的int、double之类的)

限定变量能做什么 变量 p 能访问的方法和属性，仅限于 Person 类及其父类公开的(public)内容。编译器据此检查类型安全，防止你将不兼容的对象赋值给 p

```
// 定义一个类  
public class Person {  
    private String name;  
    private int age;  
  
    // 构造函数  
    public Person(String name, int age) {  
        this.name = name;  
        this.age = age;  
    }  
}
```

// 创建对象的标准写法 Person p = new Person("Tom", 18); // 使用 new 关键字和类的构造函数

创建对象

标准写法

使用new关键字分配内存空间并创建新对象。后面接构造方法 (即类名加上参数)，用于初始化对象属性

创建对象的标准写法