

Java基础之方法

原创 2017年07月29日 13:23:55 21 0 1

1.1 方法概述

在我们的日常生活中，方法可以理解要做某件事情，而采取的解决办法。

如：小明同学在路边准备坐车来学校学习。这就面临着一件事情（坐车到学校这件事情）需要解决，解决办法呢？可采用坐公交车或坐出租车的方式来学校，那么，这种解决某件事情的办法，我们就称为方法。

在java中，方法就是用来完成解决某件事情或实现某个功能的办法。

方法实现的过程中，会包含很多条语句用于完成某些有意义的功能——通常是处理文本，控制输入或计算数值。

我们可以通过在程序代码中引用方法名称和所需的参数，实现在该程序中执行（或称调用）该方法。方法，一般都有一个返回值，用来作为事情的处理结果。

1.2 方法的语法格式

在Java中，声明一个方法的具体语法格式如下：

```
修饰符 返回值类型 方法名(参数类型 参数名1, 参数类型 参数名2, . . . . . ) {  
    执行语句  
    .....  
    return 返回值;  
}
```

对于上面的语法格式中具体说明如下：

- **修饰符**：方法的修饰符比较多，有对访问权限进行限定的，有静态修饰符static，还有最终修饰符final等，这些修饰符在后面的学习过程中会逐步介绍
- **返回值类型**：用于限定方法返回值的类型
- **参数类型**：用于限定调用方法时传入参数的数据类型
- **参数名**：是一个变量，用于接收调用方法时传入的数据
- **return关键字**：用于结束方法以及返回方法指定类型的值
- **返回值**：被return语句返回的值，该值会返回给调用者

需要特别注意的是，方法中的“参数类型 参数名1，参数类型 参数名2”被称作参数列表，它用于描述方法在被调用时需要接收的参数，如果方法不需要接收任何参数，则参数列表为空，即()内不写任何内容。方法的返回值必须为方法声明的返回值类型，如果方法中没有返回值，返回值类型要声明为void，此时，方法中return语句可以省略。

接下来通过一个案例 来演示方法的定义与使用，如下图所示。MethodDemo01.java

克己的博客 (http://blog.csdn.net/qq_36596145)

原创	粉丝	喜欢
14	5	2

- > idea报错：BeanFactory not initialized or already closed - call 'refresh' before accessing beans via ... (/qq_36596145/article/details/76728787)
 - > Java基础之引用数据类型（类） (/qq_36596145/article/details/76310135)
 - > Java基础之方法 (/qq_36596145/article/details/76308385)
 - > Java基础之数组长 (/qq_36596145/article/details/76305576)
- 更多文章 (http://blog.csdn.net/qq_36596145)

在线课程



(http://edu.csdn.net/huiyiCourse/series_detail?utm_source=blog7)

【直播】机器学习&数据挖掘7周实训--韦玮
(http://edu.csdn.net/huiyiCourse/series_detail/54?utm_source=blog7)



(http://edu.csdn.net/combo/detail/471?utm_source=blog7)

【套餐】系统集成项目管理工程师顺利通关--徐朋
(http://edu.csdn.net/combo/detail/471?utm_source=blog7)

```

public class MethodDemo01 {
    public static void main(String[] args) {
        int area = getArea(3, 5); // 调用 getArea方法
        System.out.println(" The area is " + area);
    }

    // 下面定义了一个求矩形面积的方法，接收两个参数，其中x为高，y为宽
    public static int getArea(int x, int y) {
        int temp = x * y; // 使用变量temp记住运算结果
        return temp; // 将变量temp的值返回
    }
}

```

运行结果如下图所示。

```

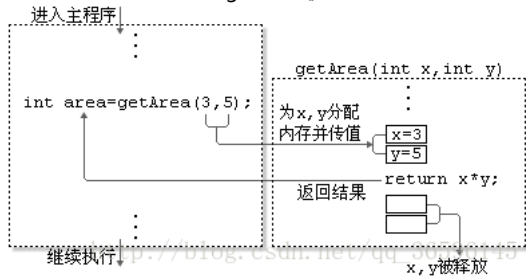
D:\java>java MethodDemo03
The area is 15

```

在上述代码中，定义了一个getArea()方法用于求矩形的面积，参数x和y分别用于接收调用方法时传入的高和宽，return语句用于返回计算所得的面积。在main()方法中通过调用getArea()方法，获得矩形的面积，并将结果打印。

1.3 方法调用图解

接下来通过一个图例演示getArea()方法的整个调用过程，如下图所示。



从上图中可以看出，在程序运行期间，参数x和y相当于在内存中定义的两个变量。当调用getArea()方法时，传入的参数3和5分别赋值给变量x和y，并将x*y的结果通过return语句返回，整个方法的调用过程结束，变量x和y被释放。

1.4 方法定义练习

分别定义如下方法：

定义无返回值无参数方法，如打印3行，每行3个*号的矩形

定义有返回值无参数方法，如键盘录入得到一个整数

定义无返回值有参数方法，如打印指定M行，每行N个*号的矩形

定义有返回值有参数方法，如求三个数的平均值

☐ 无返回值无参数方法，如打印3行，每行3个*号的矩形

```

public static void printRect() {
    //打印3行星
    for (int i=0; i<3; i++) {
        //System.out.println("***"); 相当于是打印3颗星，换行
        //每行打印3颗星
        for (int j=0; j<3; j++) {
            System.out.print("*"); // ***
        }
        System.out.println();
    }
}

```

☐ 有返回值无参数方法，如键盘录入得到一个整数

```
public static int getNumber() {
    Scanner sc = new Scanner(System.in);
    int number = sc.nextInt();
    return number;
}
```

□ **无返回值有参数方法，如打印指定M行，每行N个*号的矩形 怎么调用？**

```
public static void printRect2(int m, int n){
    //打印M行星
    for (int i=0; i<m; i++) {
        //每行中打印N颗星
        for (int j=0; j<n; j++) {
            System.out.print("*");
        }
        System.out.println();
    }
}
```

□ **有返回值有参数方法，如求三个数的平均值**

```
public static double getAvg(double a, double b, double c) {
    double result = (a+b+c)/3;
    return result;
}
```

1.5 方法的重载

我们假设要在程序中实现一个对数字求和的方法，由于参与求和数字的个数和类型都不确定，因此要针对不同的情况去设计不同的方法。接下来通过一个案例来实现对两个整数相加、对三个整数相加以及对两个小数相加的功能，具体实现如下所示。MethodDemo02.java

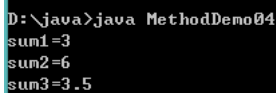
```
public class MethodDemo02 {
    public static void main(String[] args) {
        // 下面是针对求和方法的调用
        int sum1 = add01(1, 2);
        int sum2 = add02(1, 2, 3);
        double sum3 = add03(1.2, 2.3);
        // 下面的代码是打印求和的结果
        System.out.println("sum1=" + sum1);
        System.out.println("sum2=" + sum2);
        System.out.println("sum3=" + sum3);
    }

    // 下面的方法实现了两个整数相加
    public static int add01(int x, int y) {
        return x + y;
    }

    // 下面的方法实现了三个整数相加
    public static int add02(int x, int y, int z) {
        return x + y + z;
    }

    // 下面的方法实现了两个小数相加
    public static double add03(double x, double y) {
        return x + y;
    }
}
```

运行结果如下图所示。



```
D:\java>java MethodDemo04
sum1=3
sum2=6
sum3=3.5
```

从上述代码不难看出，程序需要针对每一种求和的情况都定义一个方法，如果每个方法的名称都不相同，在调用时就很难分清哪种情况该调用哪个方法。为了解决这个问题，Java允许在一个类中定义多个名称相同的方法，但是参数的类型或个数必须不同，这

就是方法的重载。

下面的三个方法互为重载关系

- `public static int add(int x,int y) {逻辑} //两个整数加法`
- `public static int add(int x,int y,int z) {逻辑} //三个整数加法`
- `public static int add(double x,double y) {逻辑} //两个小数加法`

接下来通过方法重载的方式进行修改，如下所示。MethodDemo03.java

```
public class MethodDemo03 {
    public static void main(String[] args) {
        // 下面是针对求和方法的调用
        int sum1 = add(1, 2);
        int sum2 = add(1, 2, 3);
        double sum3 = add(1.2, 2.3);
        // 下面的代码是打印求和的结果
        System.out.println("sum1=" + sum1);
        System.out.println("sum2=" + sum2);
        System.out.println("sum3=" + sum3);
    }

    // 下面的方法实现了两个整数相加
    public static int add(int x, int y) {
        return x + y;
    }
    // 下面的方法实现了三个整数相加
    public static int add(int x, int y, int z) {
        return x + y + z;
    }
    // 下面的方法实现了两个小数相加
    public static double add(double x, double y) {
        return x + y;
    }
}
```

MethodDemo02.java的运行结果和MethodDemo03.java一样，如下图所示。

```
D:\java>java MethodDemo05
sum1=3
sum2=6
sum3=3.5
```

上述代码中定义了三个同名的add()方法，它们的参数个数或类型不同，从而形成了方法的重载。

在main()方法中调用add()方法时，通过传入不同的参数便可以确定调用哪个重载的方法，如add(1,2)调用的是两个整数求和的方法。值得注意的是，方法的重载与返回值类型无关，它只有两个条件，一是方法名相同，二是参数个数或参数类型不相同。

1.5.1 重载的注意事项

□ **重载方法参数必须不同：**

参数个数不同，如method(int x)与method(int x,int y)不同
参数类型不同，如method(int x)与method(double x)不同
参数顺序不同，如method(int x,double y)与method(double x,int y)不同

□ **重载只与方法名与参数类型相关与返回值无关**

如void method(int x)与int method(int y)不是方法重载，不能同时存在

□ **重载与具体的变量标识符无关**

如method(int x)与method(int y)不是方法重载，不能同时存在

1.5.2 参数传递

参数传递，可以理解当我们要调用一个方法时，我们会把指定的数值，传递给方法中的参数，这样方法中的参数就拥有了这个指定的值，可以使用该值，在方法中运算了。这种传递方式，我们称为参数传递。

- 在这里，定义方法时，参数列表中的变量，我们称为形式参数
- 调用方法时，传入给方法的数值，我们称为实际参数

我们看下面的两段代码，来明确下参数传递的过程：

```
public class ArgumentsDemo01 {
    public static void main(String[] args) {
        int a=5;
        int b=10;
        change(a, b); //调用方法时，传入的数值称为实际参数
        System.out.println("a=" + a);
        System.out.println("b=" + b);
    }

    public static void change(int a, int b) { //方法中指定的多个参数称为形式参数
        a=200;
        b=500;
    }
}
```

程序的运行结果如下：

```
D:\java>java ArgumentsDemo01
a=5
b=10
```

再看另一段代码

```
public class ArgumentsDemo02 {
    public static void main(String[] args) {
        int[] arr = { 1, 2, 3 };
        change(arr); // 调用方法时，传入的数值称为实际参数

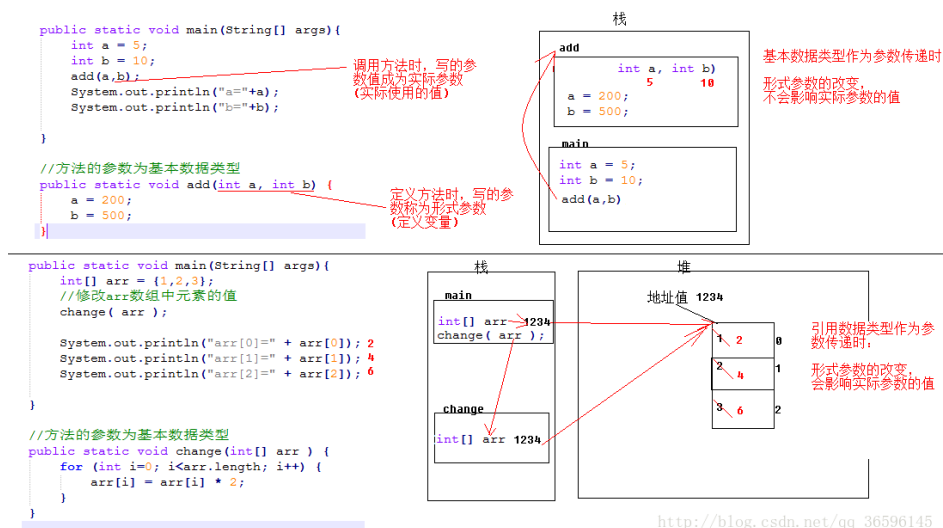
        for (int i = 0; i < arr.length; i++) {
            System.out.println(arr[i]);
        }

        public static void change(int[] arr) { // 方法中指定的多个参数称为形式参数
            for (int i = 0; i < arr.length; i++) {
                arr[i] *= 2;
            }
        }
    }
}
```

程序的运行结果如下：

```
D:\java>java ArgumentsDemo02
2
4
6
```

1.5.3 参数传递图解与结论



通过上面的两段程序可以得出如下结论：

- 当调用方法时，如果传入的数值为基本数据类型（包含String类型），形式参数的改变对实际参数不影响
- 当调用方法时，如果传入的数值为引用数据类型（String类型除外），形式参数的改变对实际参数有影响

版权声明：本文为博主原创文章，欢迎转载,但请注明出处



标签：java (<http://so.csdn.net/so/search/s.do?q=java&t=blog>) /

方法 (<http://so.csdn.net/so/search/s.do?q=方法&t=blog>) /

本文已收录于以下专栏：

0条评论

qq_36596145 (http://my.csdn.net/qq_36596145)

(http://my.csdn.net/qq_36596145)



发表评论

暂无评论

相关文章推荐

黑马程序员--Java基础加强--07.【反射创建对象 操作字段 调用方法的异同】【个人总结】(benjaminzhang666/article/details/9673743)

反射总结 反射创建对象 操作字段 调用方法的异同 ----- android培训、java培训、java学习型技术博客、期待与您交流！ -----...





u011406124 2013-07-31 22:22 964

JAVA基础--方法传参 (<http://yeluotiyang.iteye.com/blog/2064931>)



JAVA基础--方法传参 为啥拿这个当话题？在初学者阶段，许多童鞋都对方法传参比较迷茫，知其然不知其所以

然。一先说说参数传递的几个术语：值调用(call by vale):表示方法接收的是调用者传递的值。引用调用(call by reference):表示方法接收的

 叶落啼莺 2014-05-11 12:27  95

零基础学Java 10个方法 (/ww54gf8fv/article/details/74066548)

零基础学Java 10个方法 今天给大家总结一下零基础想转行做JAVA开发学习的10个小方法 小白新手学Java只要能够掌握一个合适的方法，循序渐进，正常来说...

 WW54GF8FV 2017-07-01 15:52  34



Java基础学习（01学习方法与学习心态）(http://xwk.iteye.com/blog/2134826)

Java作为现今流行的语言，很多朋友都在学习、应用，有很多论坛里的朋友都在问一个同样的问题：如何能学好Java。这是一个很大的问题，既可以长篇大论、洋洋洒洒，又可以一语概括。我在某软件培训机构教Java课程，时常需要和学员沟通一些学习方法和学习心态方面的问题，总结一下可归纳为以下5点：信念：必须抱有一定能学会，一定能学好的信念。Java和其它所有编程语言一样，无非是人与机器沟通的途

 阿尔萨斯 2008-07-03 20:21  42



Java虚拟机垃圾回收(一) 基础：回收哪些内存/对象 引用计数算法 可达性分析算法 finalize()方法 HotSpot实现分析 (/tjiyu/article/details/53982412)

下面先了解Java虚拟机垃圾回收的基础内容：如何判断对象是存活还是已经死亡？介绍相关的垃圾回收基础算法：引用计数算法、可达性分析算法，以及说明finalize()方法作用，最后再来说说HotSpo...

 tjiyu 2017-01-02 22:13  2492



黑马程序员-JAVA基础学习日志——通篇大总结及学习方法思想 (/guangruishenghui/article/details/48261861)

-----Java培训、Android培训、iOS培训、.Net培训、期待与您交流！----- Java基础知识总结 回首是为了更好向前，天空没有翅膀的痕迹，而我已...

 guangruishenghui 2015-09-07 10:04  682



java基础-参数数量可变的方法 (http://yovi.iteye.com/blog/2269388)

java基础-参数数量可变的方法 1，概述 <p style="margin-top: 5px; margin-bottom: 5px; fon

 yovi 2016-01-08 11:17  107

java部分基础类型及时间格式的数据格式相互转换方法整理 (/m0_38021722/article/details/70740021)

目录 目录 1LonglongIntegerint之间的恩怨情仇 2String与intLongbyteByteDoublefloatcharDate令人发指的纠缠 - 22 注意：本次整理涉及...

 m0_38021722 2017-04-25 15:23  115

开始学习Java RMI，远程方法调用一基础篇 (http://gyabooks.iteye.com/blog/1049824)

<!-- google_ad_client = "pub-4615277071069293"; /* 文字连接广告横行A */ google_ad_slot = "7077201342"; google_ad_width = 728; google_ad_height = 15; //--> 摘要：本译文翻译Sun的在线文档，为了说明RMI的基础知识，如果你刚开始接触RMI，那么本文对你 很有帮助。本文属于基础教程，学



gyabooks 2008-08-28 11:21 626

JAVA基础再回首（十六）——泛型的概述、使用、泛型类、泛型方法、泛型接口、泛型高级(通配符) (/m366917/article/details/52264728)

JAVA基础再回首（十六）——泛型的概述、使用、泛型类、泛型方法、泛型接口、泛型高级(通配符) 版权声明：转载必须注明本文转自程序员杜鹏程的博客:http://blog.csdn.net/m366...



m366917 2016-08-21 01:04 1238

老紫竹JAVA基础培训(11),方法的Override (http://laravel.iteye.com/blog/268006)

原文地址：http://www.java2000.net/p11697Override,主要是因为父类的功能无法满足需求，我们又不能直接修改父类的情况下，我们通过子类重写，来实现新的功能。package Lesson11;<li c



eimhee 2008-11-13 13:11 823

黑马程序员——JAVA基础----语法（三）--方法和数组 (/du_jing/article/details/49665503)

-----android培训、java培训、java学习型技术博客、期待与您交流！----- 语法（三）01_Java语言基础(方法概述和格式说明) ...



Du_jing 2015-11-05 17:22 197

Java基础 write方法应用 (http://hechuanzhen.iteye.com/blog/1565479)

```
[code="java"] package IO; import java.io.*; public class TextTa { public static void main(String[] args) { try { OutputStreamWriter os = new OutputStreamWriter( new FileOutputStream("d:\\java\\IO\\IO.txt")); //OutputStreamWriter 是字符流通向字节流的桥梁：使用指定的 charset 将要向其写入的字符编码为字节。 os.writ
```



中国凉茶 2012-06-21 11:53 2344

JAVA基础再回首（二十二）——转换流概述及用法、简化写法FileWriter和FileReader、字符缓冲流及特殊用法、字节流字符流复制文件方法总结 (/m366917/article/details/52435993)

JAVA基础再回首（二十二）——转换流概述及用法、简化写法FileWriter和FileReader、字符缓冲流及特殊用法、字节流字符流复制文件方法总结 版权声明：转载必须注明本文转自程序员杜鹏...



m366917 2016-09-04 23:47 977

[黑马程序员][java基础学习]05——方法和数组 (http://lqq5522.iteye.com/blog/2206602)

-----Java培训、Android培训、iOS培训、.Net培训、期待与您交流！----- 1.方法 1)概念： 就是完成特定功能的代码块。 2)格式 <



不上课也要睡觉 2015-04-27 22:07 178

【Doing1】Java 学习之路——过程篇、工具篇、方法篇、资源篇、书籍篇（有一定基础之后再来看） (/skiffloveblue/article/details/6984413)

Java Learning Path (一) 过程篇 每个人的学习方法是不同的，一个人的方法不见得适合另一个人，我只能谈谈自己的学习方法。因为我学习Java是完全自学的，从来没有问过别人，所以学习...



skiffloveblue 2011-11-17 21:09 693



老紫竹 **JAVA基础培训(11),方法的Override**
(<http://wangxiaojs.iteye.com/blog/292167>)

原文地址：http://www.java2000.net/p11697Override, 主要是因为父类的功能无法满足需求，我们又不能直接修改父类的情况下，我们通过子类重写，来实现新的功能。package Lesson11;<li c

 逆风的香1314 2008-11-13 13:11  281



Java基础学习（01学习方法与学习心态） (/georgebai/article/details/2611477)

信念：必须抱有一定能学会，一定能学好的信念。Java和其它所有编程语言一样，无非是人与机器沟通的途径，通俗的讲就是你说：“天王盖地虎”，机器就知道要说：“宝塔镇河妖”。我们要学习的无非就是众多的向机器...

 georgebai 2008-07-04 14:25  152

黑马程序员--java基础复习--抽象类、抽象方法和面向父类编程
(<http://liujkh123.iteye.com/blog/1895052>)

----- ASP.Net+Android+IO开发S、<a style="color: #108ac6

 liujkh123 2013-06-27 16:42  270
