

Java 语言程序设计

PROGRAMMING WITH JAVA

郑莉 胡家威 编著

<http://hujiaweibujidao.github.io/>



清华大学逸夫图书馆·北京

2013·11-2014·5

内 容 简 介

本书是一本关于 Java 语言程序设计的教材，涵盖了 Java 语言的基本语法和编程技术，其中包含了作者对 Java 语言多年开发经验的总结，目的是让初学 Java 的读者感受到 Java 语言的魅力，并掌握 Java 语言程序设计的相关技术。

无论作者使用多少技巧去查找错误，有些错误还是悄悄地隐藏了起来，并且对新读者或多或少都会造成一定的困扰。如果你发现了任何你认为是错误的地方，请通过电子邮件联系我们，对于您的帮助我们不胜感激。

前言

随着互联网的发展，程序设计成为了众多计算机和相关专业本科生不可或缺的基本技能，而其中 Java 语言在高级编程语言中仍然是佼佼者。本书从 Java 语言的基础开始，深入浅出地讲解 Java 语言的方方面面，希望读者能够深刻感受到 Java 语言的魅力以及进行 Java 开发的快乐。

TO BE CONTINUED

**** 本书的特色 ****

本书有以下几大特色：

(1) 本书的内容深入浅出。考虑到读者可能是刚刚开始学习编程语言，所以本书的内容不会讲解很深奥的问题，但是为了保证内容的完整性，肯定会对难点有所涉及，对于这部分内容我们会给出相应的资料供读者参考。

(2) 本书的习题专业经典。为了让读者得到更好地编程锻炼，章节后面的习题我们都是精心挑选和设计的，目的是为了让读者对于职场笔试题有更深入的了解，并最终做到得心应手。

(3) 本书的案例一气呵成。目前大多数的程序设计教材中都没有太多的案例，即使有的话，也只是对应于某个知识点举几个相应的例子而已。本书不一样，本书除了在对对应知识点中举写例子让读者更明白如何使用之外，本书还紧紧围绕着一个精心设计的项目来进行的。

**** 本书写给谁看 ****

本书可作为计算机或者相关专业 Java 程序设计课程教材或参考书，也可作为科技人员使用 Java 语言进行开发的参考手册。

**** 源码和文档 ****

本书所有的源码和相关文档都会在 Github 项目 XJava 上共享，地址是：<https://github.com/hujiaweibujidao/XJava>，需要的读者请自行下载使用。

**** 衷心感谢 ****

首先最需要感谢的是导师郑莉，感谢您给了我这次宝贵的机会和您一起编写这本教材，我感到自己很幸运，庆幸自己能有这样的机会发挥自己所长，然后尽自己的能力完成这本教材，谢谢您！

其次要感谢我的父母，不管我走到哪里，不管我做什么，你们永远都支持我，放手让我去闯，放手让我去成长，谢谢你们给了我一辈子无私奉献的爱，儿子永远爱你们！

胡家威

2013 年 11 月

目 录

第一章 Java 语言概述	1
1.1 Java 语言发展简史	1
1.2 Java 语言的特点	3
1.3 Java 程序运行机制	4
1.4 Java 开发环境搭建	5
1.4.1 Windows 平台的开发环境搭建	5
1.4.2 Linux 平台的开发环境搭建	5
1.4.3 Mac 平台的开发环境搭建	6
1.5 第一个 Java 程序：Hello Java	7
参考文献	9

第一章 Java 语言概述

本章主要内容

1. 了解 Java 语言的发展过程和它的特点.
2. 了解 Java 程序的运行机制.
3. 搭建 Java 开发环境并编写简单程序.

1.1 Java 语言发展简史

Java 语言历时十多年的发展,现已成为一门应用最为广泛的编程语言。Java 语言是一门纯粹的面向对象的编程语言,它完美地实现了面向对象的编程理念,在摒弃了 C++ 中复杂的概念的同时,它为开发者提供了完整的开发环境和运行环境,这样极大地方便了 Java 语言的开发。时至今日,大部分的电子商务、银行、证券等系统都是使用 Java EE 平台架构搭建的,Java EE 规范是目前最为成熟的,也是应用最广泛的企业级应用开发规范。乍一看感觉 Java 语言是专门为了 Web 开发的而设计的编程语言,但实际上这只是个巧合。

1991 年 1 月, Bill Joy(Sun 公司首席科学家) 和 James Gosling(Java 语言之父) 等人聚集在一起讨论 Stealth 项目,也就是后来的 Green 项目,该项目是研究计算机在电子消费领域的应用,它的目标是开发一个智能的电子消费设备,因为 Sun 公司预料未来科技将在家用电器领域大显身手。经过讨论之后,团队

成员开始分工，James Gosling 的任务是从众多的编程语言中选择一门适合这个项目开发的编程语言。James Gosling 一开始选择使用 C++ 语言进行尝试，结果发现 C++ 对于这个特殊的项目来说还不够完善，于是他尝试对 C++ 进行了很多的扩展和修改，但是后来他放弃了这种方式。为了达到项目的目标，他自己开发了一门新的语言——Oak(在英语中是“橡树”的意思)，因为他取名字的时候看到办公室窗户外面正好就有一棵橡树。后来发现，Oak 这个名字已经被一家显卡制造商注册了，所以 James Gosling 要想一个新的名字。在某一次去当地咖啡店喝咖啡的时候，James Gosling 突然有了灵感，想出了 Java 这个名字，因为 JAVA 是印度尼西亚爪哇岛的名字，它因盛产咖啡而出名。

1995 年年初，Sun 公司发布了 Java 语言，它将 Java 源代码都直接放到互联网上，免费提供给所有人使用，这让 Java 语言瞬间成为了一门广为认知的编程语言。

1996 年，Sun 公司发布了 JDK1.0。这个版本包括两个部分：运行环境 (Java Runtime Environment，简称 JRE) 和开发环境 (Java Development Kit，简称 JDK)。运行环境包括虚拟机 (Java Virtual Machine，简称 JVM)、核心 API、用户界面 API、集成 API 和发布技术，开发环境包括编译器和一些其他实用的开发工具。

1998 年，Sun 公司发布了 Java 历史上的一个重要版本：JDK 1.2。该版本将 Java 分成了 J2SE、J2EE 和 J2ME 三个版本，其中 J2SE 是整个 Java 技术的核心和基础，同时也是其他两个版本的基础；J2EE 是 Java 进行企业级应用开发的解决方案；J2ME 是 Java 进行移动设备和信息家电等设备开发的解决方案。

2004 年，Sun 公司发布了万众期待的 JDK 1.5，同时将 JDK 1.5 改名为 Java SE5.0，J2EE 改名为 Java EE，J2ME 改名为 Java ME。JDK 1.5 增加了很多语言特性，例如泛型、自动拆箱和装箱、可变数目的形参等等。其次该版本还发布了新的企业级平台开发规范，并推出了自己的 MVC 框架规范 JSF。

2006 年，Sun 公司发布了 JDK 1.6，也就是 Java SE 6.0。一直以来，Sun 公司保持着每两年发布一次 JDK 新版本的习惯。但是，2009 年 4 月，Oracle 公司收购了 Sun 公司，从此“江湖”上没有了“Sun”的身影。

2011 年，Oracle 公司发布了 Java SE 7，这是 Oracle 发布的第一个 Java 版本，里面加入了不少新特性，本书后面会有详细的介绍。

拓展阅读

虽然 Sun 公司“倒下”了，但是 Google 公司在 2007 年推出的基于 Linux 的开源移动操作系统 Android 极大地推动了 Java 语言的发展。Android 平台使用类 JVM 的虚拟机 Dalvik，只是它没有遵守虚拟机规范，其应用开发使用 Java 语言，最终编译成 dex 格式的文件由 Dalvik 虚拟机执行。

1.2 Java 语言的特点

Java 语言的特点很多，正是因为这些特点使得 Java 语言在众多的编程语言中脱颖而出，并在编程语言排行榜中一直处于遥遥领先的地位。下面简单介绍 Java 语言的几个重要的特点：

(1) 简单高效：Java 语言的语法和 C 或者 C++ 语言很接近，另外，Java 丢弃了 C++ 中指针、多继承等难以理解的内容，所以比较容易掌握。除此之外，Java 语言还提供了垃圾回收机制，一方面不需要程序员担心内存管理，另一方面使得内存得到高效地利用。

(2) 面向对象：Java 语言是一门纯粹的面向对象的编程语言，提供了封装、继承和多态的三大特性，支持类的单继承和接口间的多继承，并支持类和接口之间的实现机制，可以说，Java 是一门优秀的面向对象的编程语言。

(3) 安全健壮：Java 语言通常用在网络环境中，因此 Java 提供了很强大的安全机制以防止恶意代码的进攻。此外，Java 的异常处理机制、强类型机制和自动垃圾回收机制等等为 Java 程序的健壮性提供了重要的保障。

(4) 分布式和可移植：Java 语言提供了网络应用编程接口以支持网络应用的开发，同时提供了 RMI(远程方法调用) 机制以支持分布式应用开发。Java 程序是可移植的，因为 Java 的体系结构是中立的，Java 程序在 Java 平台上被编译成平台无关的字节码格式，然后可以在实现这个 Java 平台的任何系统中运行。Java 系统本身 also 具有很强的可移植性，Java 编译器是 Java 实现的，而 JRE 是用 ANSI C 实现的。

(5) 高性能和多线程：JDK 1.1 新增加了 JIT(Just-In-Time) 编译器，传统的编译器是编译一条指令，等运行完了之后就扔掉，而 JIT 会将经常用到的指令保存在内存中，当下次调用的时候就不用重新编译了。随着 JIT 编译技术的发展，Java 程序的性能越来越高了。另外，Java 语言支持多个线程同时执行，并提供多个线程之间的同步和通信机制。

1.3 Java 程序运行机制

如果按照程序的执行方式划分，高级编程语言可以分为编译型语言和解释型语言。编译型语言是使用专门的编译器，针对特定的平台将源代码编译成该平台可以执行的机器码。它的运行效率高，但是跨平台性差。解释型语言是使用专门的解释器，逐行将源代码解释成特定平台的机器码，并立即执行的语句。它运行效率较低，而且不能脱离解释器独立执行，优点是跨平台性好，只需要提供特定平台的解释器即可。

严格来讲，Java 语言既不属于编译型语言，也不属于解释型语言。Java 语言比较特殊，Java 程序首先是要经过编译器编译，但是编译出来的结果并不是生成特定平台的机器码，而是生成一种与平台无关的字节码，并且这个字节码需要解释器来解释执行。

编译器编译 Java 源程序的时候，生成的是与平台无关的字节码，这些字节码不是针对特定平台的，而是针对 Java 虚拟机的。为了实现 Java 程序的平台无关性，Sun 公司制定了 Java 虚拟机的统一标准，内容包括指令集、寄存器、类文件格式、栈、垃圾回收堆和存储区等。不同平台上的 JVM 是不同的，但是它们提供了相同的接口。JVM 负责解释执行字节码，在某些 JVM 实现中，虚拟机代码能够转换成特定系统的机器码执行，从而提高执行效率。

提问环节

你现在知道为什么“Java 语言既不属于编译型语言，也不属于解释型语言”吗？

1.4 Java 开发环境搭建

在进行 Java 程序开发之前需要在计算机上安装和配置 Java 开发环境的。不同系统平台环境配置方式略有差异，下面分别介绍三个主流系统下的环境配置方式。

1.4.1 Windows 平台的开发环境搭建

Windows 系统下搭建 Java 开发环境较为复杂，请参考下面的步骤进行：

(1) 登录 Oracle 官方 Java SE 下载网址下载对应于自己计算机平台的最新版本的 JDK，笔者写作时 JDK 的最新版本是 Java SE 7u45。

(2) 运行下载的 EXE 文件，选择安装全部的组件，路径可以使用默认的路径“C:\Program Files\Java”，但是建议修改为不包含空格的路径，例如“C:\Java\jdk7”。安装完 JDK 之后，会提示继续安装 JRE，建议将 JRE 和 JDK 安装在同一个目录下，例如可以设置 JRE 安装路径为“C:\Java\jre7”。

(3) 配置环境变量。右击“我的电脑”图标，选择“属性”，在“系统属性”窗口中选择“高级”选项卡，然后选择“环境变量”，在“环境变量”窗口中选中名称为“Path”的系统变量，点击“编辑”，在该环境变量值末尾添加“;C:\Java\jdk7\bin”，注意前面的“;”，同时读者需要将路径修改为自己设置的 JDK 安装路径下的 bin 目录。

(4) 打开 Windows 系统的命令提示符，执行“java -version”命令，如果输出了刚刚安装的 JDK 版本号的话，证明 Java 开发环境没有问题了。

1.4.2 Linux 平台的开发环境搭建

Linux 系统下搭建 Java 开发环境与 Windows 系统下的操作类似，具体步骤如下：

(1) 登录 Oracle 官方 Java SE 下载网址下载对应于自己计算机平台的最新版本的 JDK，注意选择扩展名为“.tar.gz”格式的压缩文件，笔者写作时 JDK 的最新版本是 Java SE 7u45。

(2) 将下载得到的压缩文件解压到某个目录下，为了以后的方便，建议放置在用户主目录下，例如解压到“/home/[yourname]/jdk1.7”目录。

(3) 打开终端，以管理员权限在文件“/etc/profile”末尾添加下面两行，保存之后注销当前账户重新进入系统。

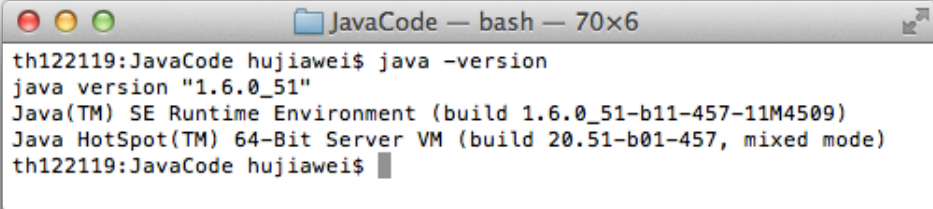
code/linux_java_configuration.sh

```
export JAVA_HOME=/home/hujiawei/jdk1.7
export PATH=$JAVA_HOME/bin:$PATH
```

(4) 执行“java -version”命令，如果输出了刚刚安装的 JDK 版本号的话，证明 Java 开发环境没有问题了。

1.4.3 Mac 平台的开发环境搭建

Mac 系统已经内置了 Java 开发环境，所以可以不用配置，但是版本可能不是最新的。打开终端，执行“java -version”可以看到当前内置的 JDK 的版本，一般是“JDK 1.6.x_yz”。对于一般的 Java 程序开发，使用 JDK 1.6 就已经足够了，如果想体验 JDK 1.7 以上的新特性，可以考虑自行安装最新版本的 JDK，本书不再介绍。

A screenshot of a macOS terminal window titled "JavaCode — bash — 70x6". The window shows the command "java -version" being executed. The output is: "java version "1.6.0_51\"", "Java(TM) SE Runtime Environment (build 1.6.0_51-b11-457-11M4509)", and "Java HotSpot(TM) 64-Bit Server VM (build 20.51-b01-457, mixed mode)". The prompt "th122119:JavaCode hujiawei\$" is visible at the bottom.

```
th122119:JavaCode hujiawei$ java -version
java version "1.6.0_51"
Java(TM) SE Runtime Environment (build 1.6.0_51-b11-457-11M4509)
Java HotSpot(TM) 64-Bit Server VM (build 20.51-b01-457, mixed mode)
th122119:JavaCode hujiawei$
```

1.5 第一个 Java 程序: Hello Java

下面我们来编写第一个 Java 程序，它的目标很简单，就是在控制台输出“Hello Java!”，以表达我们对 Java 语言世界的问候。读者可以在自己计算机中任意打开一个文本编辑器，例如 Windows 下的记事本 (Mac 下的文本编辑器，或是 Ubuntu 下的 gedit)，然后在文件中输入下面的代码，最后将文件保存为“HelloWorld.java”，保存路径最好是一个既不包含中文也不包含空格的路径下，为了方便以后的代码管理，建议读者建立一个工作目录专门用来保存编写的 Java 代码，Windows 系统可以保存在 C 盘根目录的某个文件夹下，例如“C:\JavaCode”，而 Linux 或者 Mac 系统可以保存在用户主目录的某个文件夹下，例如“/home/[yourname]/JavaCode”。

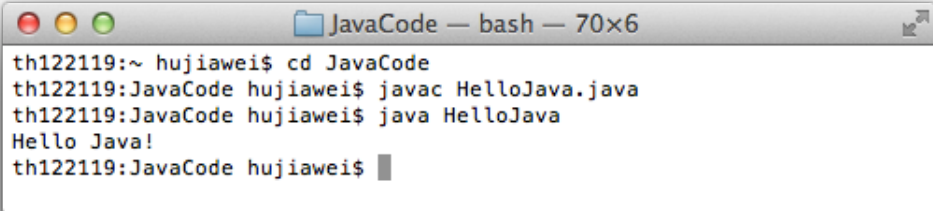
code/HelloJava.java

```
public class HelloJava {  
  
    public static void main(String[] args) {  
        System.out.println("Hello Java!");  
    }  
  
}
```

然后打开终端，切换到“HelloJava.java”文件保存的目录下，输入命令“javac HelloWorld.java”，javac 就是 Java 编译器，参数“HelloJava.java”就是要进行编译的文件。成功执行后，当前目录下会出现一个“HelloJava.class”的新文件，它就是由 Java 编译器生成的 Java 字节码文件。接下来我们使用 Java 解释器进行解释执行，输入命令“java HelloWorld”，注意末尾不需要加上“.class”即可看到控制台输出了字符串“Hello Java!”。

下面简单解释下源程序：第一行“public class HelloJava”声明了一个类，类的名称是“HelloJava”，“public”是类的访问修饰符，以后会详细介绍。在 Java 语法中，Java 源文件的扩展名一定是“.java”，其次文件名必须和文件中定义为“public”的类的名称相同，这里都是“HelloJava”。接下来，第

三行“`public static void main(String[] args)`”声明了一个方法，方法名称为“`main`”，只接收一个字符串数组作为参数，这里的“`public`”是对方法的访问修饰符，而“`static`”表明这是一个静态方法，以后都会详细介绍。在 Java 语法中，变量和方法是不能够独立存在的，它们只能存在于某个类中。一般而言，一个程序只有一个主入口，这里主入口就是 `main` 方法，它的方法声明是固定不变的，其中的参数通常是由控制台传递过来的。在 `main` 方法体中，使用“`System.out.println`”方法在系统默认的控制台输出中打印“`Hello Java!`”字符串，看到的效果如下：

A terminal window titled "JavaCode — bash — 70x6" with standard macOS window controls (red, yellow, green buttons). The terminal shows the following commands and output:

```
th122119:~ hujiawei$ cd JavaCode
th122119:JavaCode hujiawei$ javac HelloJava.java
th122119:JavaCode hujiawei$ java HelloJava
Hello Java!
th122119:JavaCode hujiawei$
```

参考文献

- [1] Eckel, Bruce, and Chuck Allison. Thinking in JAVA. Vol. 4. Englewood Cliffs: Prentice Hall, 2003.
- [2] Bloch, Joshua. Effective java. Addison-Wesley Professional, 2008.
- [3] 李刚. 疯狂 Java 讲义. 电子工业出版社, 2008.