普通高等教育“十三五”精品规划教材（计算机网络技术系列）

# Java 程序设计

主　编　甘　霞
副主编　王中婧　李　亮
主　审　何友鸣

中国水利水电出版社
www.waterpub.com.cn

普通高等教育“十三五”精品规划教材

（计算机网络技术系列）

# Java 程序设计

主　编　甘　霞

副主编　王中婧　李　亮

主　审　何友鸣

中国水利水电出版社
www.waterpub.com.cn

·北京·

## 内 容 提 要

本书全面系统地介绍了Java语言的特点及其应用技术，内容上以Java的基础程序设计、面向对象程序设计和事件处理为三大主线，利用浅显易懂的语言、简单丰富的实例，完整地介绍了Java面向对象程序设计的要点和难点。全书共14章，内容包括Java语言概述、Java语言基础、类与对象、键盘输入与流程控制、数组、类的方法、继承性和多态性、异常处理、I/O技术与文件处理、多线程、图形界面设计、小程序设计、数据库程序设计和网络编程。

本书在章节编排与内容上注重教材的体系，其特点是结构合理、概念清晰、语言简练；在结构上特别注重前后内容的连贯性，力求抓住关键、突出重点、分解难点，体现“理论性、实用性、技术性”三者相结合的编写特色。

本书可以作为高等院校计算机、信息管理与信息系统及相关专业的教学用书，也可以作为职业教育的培训用书和Java初学者的入门教材。

**图书在版编目（CIP）数据**

Java程序设计 / 甘霞主编. -- 北京 : 中国水利水电出版社, 2018.8
普通高等教育“十三五”精品规划教材. 计算机网络技术系列
ISBN 978-7-5170-6726-9

Ⅰ. ①J… Ⅱ. ①甘… Ⅲ. ①JAVA语言－程序设计－高等学校－教材 Ⅳ. ①TP312.8

中国版本图书馆CIP数据核字(2018)第185603号

**策划编辑：杜 威　责任编辑：张玉玲　加工编辑：张青月　封面设计：李 佳**

| | |
|---|---|
| 书　名 | 普通高等教育“十三五”精品规划教材（计算机网络技术系列）<br>Java 程序设计　Java CHENGXU SHEJI |
| 作　者 | 主　编　甘　霞<br>副主编　王中婧　李　亮<br>主　审　何友鸣 |
| 出版发行 | 中国水利水电出版社<br>（北京市海淀区玉渊潭南路1号D座　100038）<br>网址：www.waterpub.com.cn<br>E-mail：mchannel@263.net（万水）<br>sales@waterpub.com.cn<br>电话：（010）68367658（营销中心）、82562819（万水） |
| 经　售 | 全国各地新华书店和相关出版物销售网点 |
| 排　版 | 北京万水电子信息有限公司 |
| 印　刷 | 三河市祥宏印务有限公司 |
| 规　格 | 184mm×260mm　16开本　17印张　421千字 |
| 版　次 | 2018年8月第1版　2018年8月第1次印刷 |
| 印　数 | 0001—3000册 |
| 定　价 | 38.00元 |

# 前　言

Java 语言是完全面向对象的，具有容易学习、功能强大、程序可读性好等优点，是其他传统语言无可比拟的。

本书在内容编排上做了精心地设置与选取，注重基础知识的理解与基本技能的培养。本书内容思路清晰、结构严谨，在内容的叙述上由浅入深、循序渐进、用语规范，全面准确讲述基本语法和面向对象技术等理论内容；在结构上特别注重前后内容的连贯性，力求抓住关键、突出重点、分解难点，体现“理论性、实用性、技术性”三者相结合的编写特色。同时将实用性强的应用程序穿插在理论叙述中，以实例体现和巩固理论基础知识，并结合新技术的发展趋势介绍网络编程等。本书共分 14 章，其中第 1 章至第 5 章介绍了 Java 语言的基础；第 6 章和第 7 章介绍了面向对象的程序设计；第 8 章和第 9 章介绍了异常处理和文件处理；第 10 章介绍了多线程；第 11 章和第 12 章介绍了图形界面（UI）设计和小程序设计；第 13 章介绍了数据库程序设计；第 14 章介绍了网络编程。

本书由甘霞任主编，王中婧、李亮任副主编，何友鸣任主审。非常感谢何友鸣教授以及两位参编宋洁和张永进老师在本书编写过程中所做出的贡献。

由于编者水平有限，书中疏漏和不妥之处在所难免，敬请广大读者和同行批评指正。

编　者

2018 年 5 月

# 目　　录

# 1 Java 语言概述

Java 语言是一种简单易用、完全面向对象、与平台无关、安全可靠、主要面向 Internet 的开发工具。

## 1.1 Java 语言的诞生与发展

Java 语言诞生于 20 世纪 90 年代初期。正式问世以来，它的快速发展已经让整个 Web 世界发生了翻天覆地的变化。

Java 语言的前身是 Sun 公司（Sun 公司于 2009 年 4 月被 Oracle 公司收购）开发的一种用于智能化家电的名为 Oak（橡树）的语言，Oak 语言的基础是当时最为流行的 C 和 C++语言。但是，由于一些非技术上的原因，Oak 语言并没有得到迅速的推广。直到 1993 年，WWW（万维网）迅速发展，Sun 公司发现可以利用 Oak 语言的技术来创造含有动态内容的 WWW 网页，于是已经受冷落的 Oak 语言又被重新开发和改造。改造后的 Oak 语言被改名为 Java 语言，Java 是太平洋上的一个盛产咖啡的岛屿的名字。终于，在 1995 年，Java 被定位于网络应用的程序设计语言。

Java 语言问世的时间虽然不长，但却被业界所接受，IBM、Apple、DEC、Adobe、HP、Oracle、Toshiba、Netscape 和 Microsoft 等大公司都已经购买了 Java 语言的许可证。Microsoft 还从其 Web 浏览器 Internet Explorer 3.0 版开始增加了对 Java 语言的支持。同时，众多的软件开发商也开发了许多支持 Java 的产品。在目前以网络为中心的计算机时代，不支持 HTML 和 Java 语言就意味着应用程序的应用范围只能局限于同质的环境。

随着 Java Servlet 的推出，Java 在电子商务方面开始崭露头角。最新的 Java Server Page(JSP) 技术的推出，更是让 Java 语言成为基于 Web 应用程序的首选开发工具。Internet 的普及和迅猛发展，以及 Web 技术的不断渗透，使 Java 语言在现代社会的经济发展和科学研究中占据了越来越重要的地位。

## 1.2 Java 语言的特点和技术

Java 语言是一种跨平台的、适合于分布式计算环境的面向对象的编程语言。它的特点很多，如简单易学、面向对象、分布式、解释型、可靠性、安全性、平台无关性、可移植性、高性能、多线程、动态性等。下面介绍 Java 语言的几个重要特性。

1. 简单易学

Java 语言虽然衍生自 C++，但与 C++相比，Java 是一个完全面向对象的编程语言。出于安全性和稳定性的考虑，Java 去掉了 C/C++支持的三个不易理解和掌握的数据类型：指针（pointer）、联合体（union）和结构体（struct）。这样做的目的是使用户不能通过 Java 程序直接访问内存地址，从而保证了程序更高的安全性。而 C/C++中联合体和结构体的功能完全可以在 Java 中用类及类的属性等面向对象的方法来实现，这不但更加合理规范，而且降低了学习难度。

2. 面向对象

Java 语言最吸引人之处，就在于它是一种以对象为中心、以消息为驱动的面向对象的编程语言。面向对象的语言都支持三个概念：封装、继承和多态。Java 语言也是如此。

（1）封装

所谓封装，就是指利用抽象数据类型将数据和基于数据的操作封装在一起，数据被保护在抽象数据类型的内部，系统的其他部分只有通过封装在数据外面的被授权的操作，才能够与这个抽象数据类型交互。

（2）继承

继承是指一个对象直接使用另一个对象的属性和方法。Java 语言给用户提供了一系列的类，并且 Java 语言的类很有层次结构，子类可以继承父类的属性和方法。Java 语言只支持单一继承，这样就大大降低了复杂度，但在 Java 语言中，可以通过接口来实现多重继承。

（3）多态

多态是指一个程序中同名的多个不同方法共存的情况，即一个对外接口、多个内在实现方法。面向对象的程序中多态的情况有多种，可以通过子类对父类方法的覆盖实现多态，也可以利用重载在同一个类中定义多个同名的不同方法来实现多态。多态的特点使得它们不需了解对方的具体细节就可以很好地共同工作。这个优点，对程序的设计、开发和维护都有很大的好处。

3. 平台无关性

Java 是与平台无关的语言，这是指 Java 语言编写的应用程序不用修改就可在不同的软硬件平台上运行。

平台无关有两种：源代码级和目标代码级。C 和 C++语言具有一定程度的源代码级平台无关，即用 C 和 C++语言编写的应用程序不用修改，只需重新编译就可以在不同平台上运行。Java 语言是靠 Java 虚拟机（Java Virtual Machine，JVM）在目标代码级实现平台无关性的，可以说，JVM 是 Java 平台无关的基础。

4. 分布式

分布式包括数据分布和操作分布。数据分布是指数据可以分散在网络的不同主机上；操作分布是指把一个计算分散在不同的主机上处理。Java 语言支持 WWW 客户机/服务器计算模

式，因此它支持这两种分布性。对于数据分布，Java 语言提供一个称作 URL 的对象，利用这个对象可以打开并访问 URL 地址上的对象，访问方式与访问本地文件系统相同。对于操作分布，Java 的小程序可以从服务器下载到客户端，将部分计算在客户端进行，从而提高系统执行效率。同时，Java 语言提供了一整套网络类库，开发人员可以利用类库进行网络程序设计，方便地实现 Java 语言的分布式特性。

5. 可靠性

Java 语言具有很高的可靠性。首先，Java 语言是强类型的语言，要求显示的方法说明，这就保证了编译器可以发现方法的调用错误，保证了程序更加可靠；其次，Java 语言不支持指针，这就避免了对内存的非法访问；再次，Java 语言的自动单元回收功能防止了内存丢失等动态内存分配导致的问题；然后，Java 解释器运行时实施检查，可以发现数组和字符串访问的越界；最后，Java 语言提供了异常处理机制，可以把一组错误的代码放在一个地方，这样可以简化错误处理任务，便于恢复。

6. 安全性

Java 是一种主要用于网络应用程序开发的语言，因此对安全性要有较高的要求。如果没有安全保证，用户从网络上下载程序执行就会非常危险。

Java 语言具有较高的安全性，它通过自己的安全机制防止了病毒程序的产生和下载程序对本地系统的威胁破坏。当 Java 字节码进入解释器时，首先必须经过字节码校验器的检查；其次，Java 解释器将决定程序中类的内存布局；再次，类装载器负责把来自网络的类装载到单独的内存区域，避免应用程序之间相互干扰破坏；最后，客户端用户还可以限制从网络上装载的类智能访问某些文件系统。综合了上述几种机制，使得 Java 成为了安全的编程语言。

7. 支持多线程

线程是比进程更小的可并发执行的单位。C++语言没有内置的多线程机制，因此必须调用操作系统的多线程功能来进行多线程程序设计。而 Java 语言却提供了多线程支持并在两个方面支持多线程：一方面，Java 环境本身就是多线程的，若干个系统线程运行，负责必要的无用单元回收、系统维护等系统级操作；另一方面，Java 语言内置多线程机制，可以大大简化多线程应用程序，提高程序执行效率。但需要注意的是，Java 语言的多线程在一定程度上受到运行时支持平台的限制。

8. 支持网络编程

Java 语言通过它所提供的类库可以处理 TCP/IP 协议，用户通过 URL 地址在网络上可以很方便地访问其他对象。Java 的小程序是动态、安全、跨平台的网络应用程序。Java 的小程序嵌入在 HTML 文档中，通过主页发布到 Internet。网络用户访问服务器的小程序时，这些小程序从网络上进行传输，然后在支持 Java 的浏览器中运行。

9. 编译与解释并存

用 Java 语言编写的程序称为源文件（扩展名为.java 的文件），源文件是不能被计算机执行的。要想使程序得以运行，必须利用编译器（不同的计算机语言有不同的编译器）对源文件进行编译，将源文件编译（即翻译）成计算机能懂的语言。C 语言是针对特定的 CPU 芯片对源文件进行编译，将源文件编译成二进制码（.exe 文件，也被称为机器码），这样计算机就可以读懂它，它就可以按照人们的意愿去实现相应的功能。但是 C 语言的这种编译方式生成的目标程序与特定的计算机有关，一旦运行环境有所变化就可能需要重新修改源程序并针对新的运

行环境重新编译，生成新的目标程序。而 Java 语言不针对特定的 CPU 芯片进行编译，Java 提供的编译器并不是把源文件编译成二进制码，而是将其编译成一种独立于机器平台的中间代码，这种中间代码被称为字节码（扩展名为.class）。字节码可以被 Java 解释器所执行，由解释器将字节码再翻译成二进制码，使程序得以运行。字节码非常类似于机器指令，但字节码与具体机器是无关的，并不能在具体的平台上执行，而要通过 Java 运行系统中的解释器来解释执行。也就是说，Java 程序的运行要经过两个步骤来完成：首先是由编译器将 Java 源程序编译成字节码文件，然后再由 Java 运行系统解释执行字节码文件。这就是所谓的编译与解释并存。从本质上说，Java 语言属于解释型的高级程序设计语言，但 Java 语言通过字节码的方式，又在一定程度上克服了传统解释型语言的低执行效率，同时又保留了解释型语言可移植的特点。所以 Java 程序的运行效率比较高，而且，由于字节码并不专对一种特定的机器，因此 Java 程序无须重新编译便可在多种不同的计算机上运行。

目前 Java 技术主要包括三个方面的内容。

（1）Java SE（Java Platform Standard Edition）：以前的版本称为 J2SE，是 Java 平台的标准版，用于工作站、PC 机的 Java 标准平台。它体现了 Sun 公司的开放精神，被称为是“互联网上的世界语”。



（2）Java ME（Java Platform Micro Edition）：以前的版本称为 J2ME，是 Java 平台的精简版，致力于消费产品和嵌入设备的最佳解决方案。Java ME 是移动商务最佳的应用典范，不论是进行无线通信，还是在手机、PDA 等小型电子装置上的开发应用，均可采用 Java ME 作为开发工具及应用平台。它提供了 HTTP 等高级 Internet 协议，可以使移动电话能以 Client/Server 方式直接访问 Internet 的全部信息，不同的 Client 访问不同的文件，此外还能访问本地存储区，提供高效率的无线交流。

（3）Java EE（Java Platform Enterprise Edition）：以前的版本称为 J2EE，是 Java 平台的企业版。它是以企业为环境而开发应用程序的解决方案。它提供的企业 e-Business 架构及 Web Services 服务，以其优越的跨平台能力与开放的标准，深受广大企业用户的喜爱。目前它已成为开发商创建电子商务应用的事实标准。

## 1.3 Java 虚拟机

大部分的计算机语言程序都必须经过编译（Compile）或解释（Interpret）的操作后才能在计算机上运行。例如 C/C++等是属于编译型的语言，而 Basic 与 Lisp 等则是属于解释型的语言。然而，Java 程序却比较特殊，它必须先经过编译的过程，然后再利用解释的方式来运行。通过编译器（Compiler），Java 程序会被转成与平台无关（Platform-Independent）的机器码，Java 称之为“字节码”。字节码文件的扩展名为.class。通过 Java 的解释器便可解释并运行 Java 的字节码。

字节码是 Java 虚拟机 JVM 的指令组，和 CPU 上的微指令码很相像。它的形式为“<操作码><操作数>”，其中操作码就是指令码。Java 语言编译成字节码后文件较小，便于网络传输。

字节码最大的好处是可跨平台运行，即 Java 的字节码编写一次便可以到处运行。用户使用任何一种 Java 编译器将 Java 源程序（.java）编译成字节码文件（.class）后，无论使用哪种操作系统，都可以在含有 JVM 的平台上运行。这种跨平台的特性，也是使 Java 语言急速普及

的原因之一。

任何一种可以运行 Java 字节码的软件均可看成是 Java 的“虚拟机”，如浏览器与 Java 的开发工具等皆可视为一个 JVM。很自然地，可以把 Java 的字节码看成是 JVM 上所运行的机器码，即 JVM 中的解释器负责将字节码解释成本地的机器码。所以从底层上看，JVM 就是以 Java 字节码为指令组的“软 CPU”。也就是说，JVM 是可运行 Java 字节码的假想计算机。它的作用类似于 Windows 操作系统，只不过在 Windows 上运行的是.exe 文件， 而在 JVM 上运行的是 Java 字节码文件，也就是扩展名为.class 的文件。JVM 其实就是一种字节码解释器。

## 1.4 Java 程序种类和结构

使用 Java 语言可以编写两种类型的程序：Application（应用程序）和 Applet（小程序）。这两种程序的开发原理是相同的，但是在运行环境和计算结构上却有着显著的不同。

应用程序是从命令行运行的程序，它可以在 Java 平台上独立运行，通常称为 Java 应用程序。Java 应用程序是独立完整的程序，在命令行调用独立的解释器软件即可运行。另外，Java 应用程序的主类必须包含有一个定义为 public static void main(String[] args)的主方法，这个方法是 Java 应用程序的标志，同时也是 Java 应用程序执行的入口点。也就是说在应用程序中包含有 main()方法的类一定是主类，但主类并不一定要求是 public 类。

小程序是嵌入在 HTML 文档中的 Java 程序，需要搭配浏览器来运行，因此称为小程序。由此可见，当运行一个 Java 小程序时，同时还要为它编写一个 HTML 文件，然后在 WWW 浏览器中运行这个 HTML 文件，就可以激活浏览器中的 Java 解释器。另外，也可以调用一些能够模拟浏览器环境并执行 Java 小程序的软件来直接运行 Java 小程序。由于浏览器受安全控制的限制，所以 Java 小程序一般使用模拟浏览器环境的软件来执行。

Java 小程序与 Java 应用程序之间存在着很多不同之处，具体如下：

首先，小程序和应用程序之间的技术差别在于运行环境。Java 应用程序运行在最简单的环境中，它的唯一外部输入就是命令行参数；而小程序则需要来自 Web 浏览器的大量信息，它是内嵌在 HTML 文件里，在 WWW 浏览器这个特定环境下运行的，它需要知道何时启动，何时放入浏览器窗口，在何处、何时去激活、关闭等。

其次，由于小程序和应用程序的执行环境不同，它们的最低要求也不同。在应用方面，WWW 使小程序的发布十分便利，因此小程序更适合在 Internet 上的使用；相反，非网络系统和内存教学的系统更适合使用 Java 应用程序。

再次，Java 小程序可以直接利用浏览器或 appletviewer 提供的图形用户界面，而 Java 应用程序则必须另外书写专用代码来营建自己的图形界面。

最后，小程序的主类（程序执行的入口点）必须是一个继承自系统类 JApplet 或 Applet 的子类，且该类必须是 public 类；而 Java 应用程序的主类必须是包含有主方法 main()的类。

由此可见，Java 小程序与 Java 应用程序之间在编写组成、计算结构和运行方式上都有着较大的差别。为了便于读者理解和应用，我们在表 1.1 中列出了应用程序与小程序之间的主要差别。

表 1.1　应用程序与小程序的主要差别

| 功能要求 | 应用程序 | 小程序 |
| --- | --- | --- |
| 使用图形 | 可选 | 固定用图形 |
| 发布 | 主要从文件系统装入 | 通过 HTML 连接 |
| 内存要求 | 最低 Java 应用程序要求 | Java 程序加 Web 浏览器要求 |
| 环境输入 | 命令行参数 | 嵌入 HTML 文档的参数 |
| JVM 所要求的执行过程 | 主方法（main()）启动过程 | init()初始化过程<br>start()启动过程<br>stop()暂停/关闭过程<br>destroy()终止过程<br>paint()绘图过程 |

小程序的编写方式与应用程序类似，因此只要熟悉了 Java 应用程序的编写方式，很快就能学会编写小程序。

一个复杂的程序可以由一个或多个 Java 源文件构成，每个文件中可以有多个类定义。下面的程序是一个一般的 Java 应用程序文件。

```
package ch01;
import java.io.*;
public class app
{
    public static void main(String[] args)
    {
        char c=';
        System.out.print("请输入一个字符：");
        try{
            c=(char)System.in.read();
            }catch(IOException s){}
        System.out.println("您输入的字符是："+c);
    }
}
```

从这个程序可以看出，一般的 Java 源程序文件由以下三部分组成：

package 语句（0 句或 1 句）

import 语句（0 句或多句）

类定义（1 个或多个类定义）

其中，package 语句表示本程序所属的包，它只能有一个或者没有。如果有，必须放在最前面；如果没有，表示本程序属于默认包。

import 语句表示引入其他类库中的类以便使用。import 语句可以有 0 或多个，它必须放在类定义的前面。

类定义是 Java 源程序的主要部分，每个文件中可以定义若干个类。

Java 程序中使用关键字 class 定义类，每个类的定义由类头定义和类体定义两部分组成。类体部分用来定义属性和方法这两种类的成员，其中方法类似于其他高级语言中的函数，而属性则类似于变量。类头部分除了声明类名之外，还可以说明类的继承特性，当一个类被定义为另一个已经存在的类（称为父类）的子类时，它就可以从其父类中继承一些已定义好的类成员，

而不必自己重复编码。

在类体中通常有两种组成成分：一种是域，包括变量、常量、对象数组等独立的实体；另一种是方法，类似于函数的代码单元块，这两种组成成分通称为类的成员。在上面的例子中，类 app 中只有一个类成员，即第 5 行定义的方法 main()。用来标志方法头的是方法名后面的一对小括号，小括号中是该方法使用的形式参数，方法名前面的 public 是用来说明这个方法属性的修饰符。方法体部分由若干以分号“;”结尾的语句组成，并由一对大括号{}括起来，在方法体内部不能再定义其他的方法。

同其他高级语言一样，语句是构成 Java 程序的基本单位之一。每一条 Java 语句都由分号“;”结束，其构成应该符合 Java 语言的语法规则。类和方法中的所有语句应该用一对大括号{}括起来。除 package 及 import 语句之外，其他执行具体操作的语句都只能存在于类的大括号之中。

比语句更小的语言单位是表达式、变量、常量和关键字等，Java 的语句就是由它们构成的。其中，声明变量与常量的关键字是 Java 语言语法规定的保留字，用户程序定义的常量和变量的取名不能与保留字相同。

Java 源程序的书写格式比较自由，如语句之间可以换行，也可以不换行，但养成一种良好的书写习惯比较重要。

**注意**：Java 是严格区分字母大小写的语言。书写时，字母大小写不能混淆。

一个程序中可以有多个类，但只能有一个类是主类。在 Java 应用程序中，这个主类是指包含 main()方法的类。在 Java 小程序中，这个主类是一个继承自系统类 JApplet 或 Applet 的子类。应用程序的主类不一定要求是 public 类，但小程序的主类一定要求是 public 类。主类是 Java 程序执行的入口点。同一个 Java 程序中定义的若干类之间没有严格的逻辑关系要求，但它们通常是在一起协同工作的，每一个类都可以适用其他类中定义的静态属性或方法。

## 1.5 Java 开发环境

Java 开发工具（Java SE Development Kits，JDK）是许多 Java 程序员使用的开发环境。尽管许多编程人员已经在使用第三方的开发工具，但 JDK 仍被当作 Java 程序开发的重要工具。

JDK 由 Java API、Java 运行环境和一组建立、测试工具的实用程序等组成。其核心是 Java API，它是 Java 提供给编程人员使用的标准类库，开发人员需要用这些类来实现 Java 语言的功能。Java API 包括一些重要的语言结构以及基本图形、网络和文件 I/O 等。

作为 JDK 的实用程序，工具库中的主要程序都放在 JDK 安装文件夹下。其中 bin 子文件夹中包含了所有相关的可执行文件，下面是 bin 文件夹下的常用命令。

javac.exe：Java 编译器，将 Java 源代码文件转换成字节码文件。

java.exe：Java 解释器，执行 Java 程序的字节码文件。

appletviewer.exe：小程序浏览器，执行嵌入在 HTML 文件中的 Java 小程序的 Java 浏览器。

javadoc.exe：根据 Java 源代码及说明语句生成 Java 程序的 HTML 格式的帮助文档。

jdb.exe：Java 调试器，可以逐行执行程序、设置断点和检查变量。

jar.exe：创建扩展名为.jar（Java Archive，Java 归档）的压缩文件，与 zip 压缩文件格式相同。

### 1.5.1 JDK 的下载与安装

Oracle 公司提供了多种操作系统下的 JDK，随着时间的推移和技术进步，JDK 版本也在不断升级。各种操作系统下的 JDK 的各种版本在使用上基本相似，用户可以根据自己的使用环境，从 Oracle 公司的网站上下载相应的 JDK 版本。一般情况下是越新越好。本书以 JDK 8.0 为例。

1. 下载 JDK

进入 Java SE 的下载网页下载 JDK，用户可以根据自己所用的操作系统（Windows 或 Linux）、位数（32 位或 64 位）选择不同的链接下载。

2. 安装 JDK

下载得到 JDK 文件之后，即可进行安装，安装 JDK 的步骤如下：

（1）双击 JDK 安装文件 jdk_8u73_windows_i586_8.0.730.2.exe，弹出如图 1.1 所示的安装向导界面。



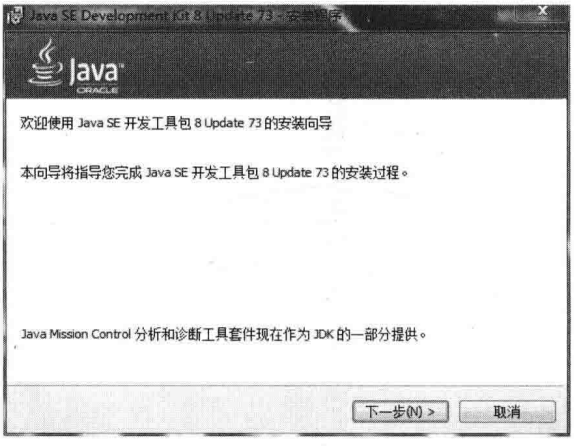


图 1.1 安装 JDK 向导界面

（2）单击“下一步”按钮，进入图 1.2 所示的“定制安装”界面。

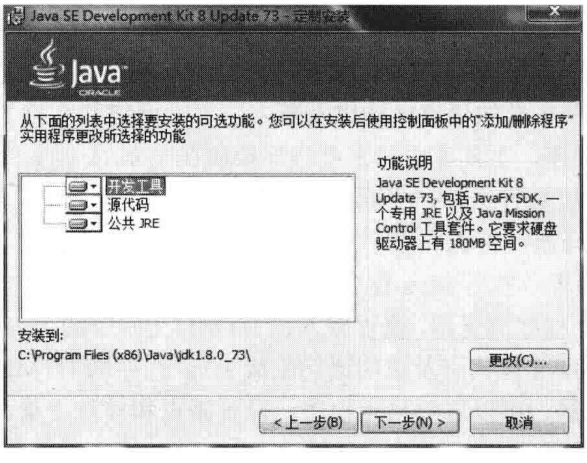


图 1.2 JDK“定制安装”界面

（3）在图 1.2 所示的界面中，用户可以选择欲安装的项目，建议使用默认设置。然后由用户确定安装位置，用户可以单击“更改”按钮选择欲安装的路径或使用默认值，建议直接使用默认的安装路径，也就是 C:\Program Files\Java\jdk1.8.0_73\，然后单击“下一步”按钮继续。

（4）开始进行文件复制与安装。

（5）然后出现“目标文件夹”界面，要求设定 Java 运行环境（Java Runtime Environment，JRE）安装文件夹，也可以单击“更改”按钮进行修改，但建议直接使用默认的安装路径。之后单击“下一步”按钮，出现如图 1.3 所示的“Java 安装-进度”界面。


图 1.3 “Java 安装-进度”界面

（6）继续进行文件复制与安装，安装完成后弹出如图 1.4 所示的界面。

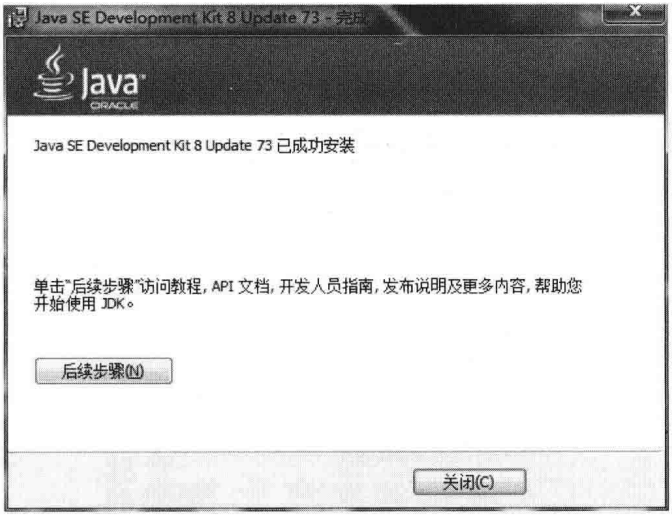


图 1.4 JDK 安装完成

单击“关闭”按钮完成安装，JDK 被安装到 C:\Program Files\Java\jdk1.8.0_73 文件夹下，此文件夹称为 JDK 安装文件夹或安装路径。在该文件夹下有几个子文件夹：

bin：该文件夹存放 javac、java、appletviewer 等命令程序。

db：给文件夹包含 Apache Derby 数据库等开放资源，支持 JDBC4.0 的规范。

include：该文件夹存放与 C 程序相关的头文件。

jre：该文件夹存放 Java 运行环境相关的文件。

lib：该文件夹存放 Java 类库。

另外，在安装文件夹下还有名为 src.zip 的压缩文件，该文件中含有 Java API 所有类的源代码，有兴趣的读者可以解压缩此文件，阅读并学习其中的源程序。

### 1.5.2 设置 JDK 的操作环境

在使用 Java 来编译与运行程序之前，必须先设置系统环境变量。所谓系统环境变量，就是在操作系统中定义的变量，可供操作系统上的所有应用程序使用。为此，需要设置两个环境变量：一个是系统路径 Path，另一个是类路径 ClassPath。

Path 环境变量的作用是设置供操作系统去寻找可执行文件（如.exe、.com、.bat 等）的路径，对 Java 而言即 Java 的安装路径。也就是说，如果操作系统在当前文件夹下没有找到想要执行的程序或命令时，操作系统就会按照 Path 环境变量指定的路径依次去查找，以最先找到的为准。Path 环境变量可以存放多个路径，路径与路径之间用分号“;”隔开。

ClassPath 环境变量的作用与 Path 的作用相似，ClassPath 是 JVM 执行 Java 程序时搜索类的路径的顺序，以最先找到的为准。JVM 查找类的过程，同 Windows 查找可执行文件的过程稍有不同，它不会在当前文件夹下查找，只找 ClassPath 指定的文件夹。也就是说，JVM 除了在 ClassPath 的环境变量指定的文件夹中查找要运行的类之外，是不会在当前文件夹下查找相应类的，即 ClassPath 环境变量的作用是告诉 Java 解释器在哪里找到.class 文件及相关的库程序。

下面介绍在 Windows 7 操作系统中设置系统环境变量 Path 和 ClassPath 的方法。

（1）选择“控制面板”→“系统和安全”→“系统”命令，在弹出窗口的左侧窗格中选择“高级系统设置”命令，弹出“系统属性”对话框，在该对话框中选择“高级”选项卡，如图 1.5 所示。在图 1.5 中单击“环境变量”按钮后，弹出如图 1.6 所示的“环境变量”对话框。

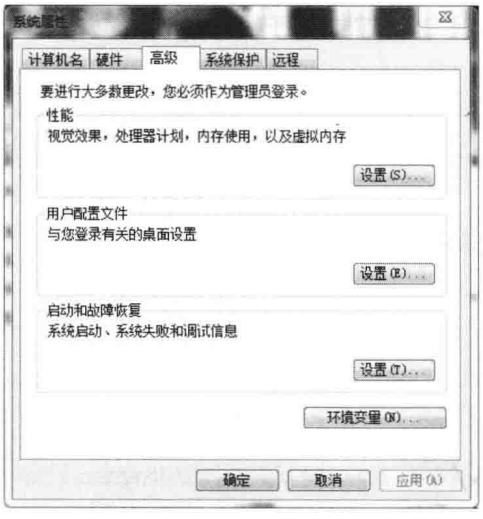


图 1.5 “系统属性”对话框中的“高级”选项卡