



国家示范（骨干）高职院校
重点建设专业优质核心课程系列教材

主 编 彭德林 迟国栋
副主编 卢凤伟 何 鑫 石芳堂
主 审 金忠伟 薛永三

Java 程序设计

项目教程



中国水利水电出版社
www.waterpub.com.cn

国家示范（骨干）高职院校重点建设专业优质核心课程系列教材

Java 程序设计项目教程

主 编 彭德林 迟国栋

副主编 卢凤伟 何 鑫 石芳堂

主 审 金忠伟 薛永三



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

Java 语言具有面向对象、与平台无关、安全、稳定和多线程等特点,是目前软件设计中极为健壮的编程语言。Java 语言不仅可以用来开发大型的应用程序,而且特别适合于在 Internet 上应用开发,Java 已成为网络时代最重要的编程语言之一。对于计算机专业学生来说掌握 Java 是必需具备的技能之一。虽然现在关于 Java 的教材很多,但与社会实际应用相接轨的教材却不多,本书以社会实际需求为主线,以社会上应用最广的 Eclipse 为开发平台,采用项目实战教学模式进行内容编排,使学生真正能学有所用。

本教材在强调使学生全面掌握 Java 基本操作的基础上,把知识点与应用项目相结合,使学生学习起来有的放矢,操作时也更加得心应手。本书主要内容包括:Java 语言基础、Java 面向对象程序设计等。在项目实战教学思想指导下,每个项目都设置了若干个实战任务,内容以相关应用操作为主线进行叙述,通过案例进行应用剖析,并配有习题,方便学生检验学习成果。

本书深入浅出,操作性强,以实际应用为切入点,符合高职高专特点,适合高职高专学生使用,也可用作各类技能培训教材。

本书提供了配套电子教案和程序源代码,读者可以从中国水利水电出版社及万水书苑网站上下载,网址为: <http://www.waterpub.com.cn/softdown/> 和 <http://www.wsbookshow.com>。

图书在版编目(CIP)数据

Java 程序设计项目教程 / 彭德林, 迟国栋主编. —

北京: 中国水利水电出版社, 2014.8

国家示范(骨干)高职院校重点建设专业优质核心课程系列教材

ISBN 978-7-5170-2398-2

I. ①J… II. ①彭… ②迟… III. ①JAVA 语言—程序设计—高等职业教育—教材 IV. ①TP312

中国版本图书馆CIP数据核字(2014)第199892号

策划编辑: 石永峰

责任编辑: 李 炎

封面设计: 李 佳

书 名	国家示范(骨干)高职院校重点建设专业优质核心课程系列教材 Java 程序设计项目教程
作 者	主 编 彭德林 迟国栋 副主编 卢凤伟 何 鑫 石芳堂 主 审 金忠伟 薛永三
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路1号D座 100038) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn
经 售	电话: (010) 68367658 (发行部)、82562819 (万水) 北京科水图书销售中心(零售) 电话: (010) 88383994、63202643、68545874 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	北京上元柏昌印刷有限公司
规 格	184mm×260mm 16 开本 16.5 印张 428 千字
版 次	2014 年 8 月第 1 版 2014 年 8 月第 1 次印刷
印 数	0001—3000 册
定 价	32.00 元

凡购买我社图书,如有缺页、倒页、脱页的,本社发行部负责调换

版权所有·侵权必究

前 言

Java 语言经过最近十多年的快速发展,技术越来越成熟,应用也越来越广,在软件开发方面的霸主地位更加稳固。Java 语言已经是计算机类专业必修的课程之一,教材的优劣直接影响 Java 教学效果的好坏。本书在保证知识体系完整性的前提下,采用“任务驱动式教学”,以社会上应用最广的 Eclipse 为开发平台,突出实用性、针对性,引导学生“在做中学”,培养学生分析问题和解决问题的能力,把提高学生动手能力和综合素质放在首位。针对此种情况,我们与安博(大连)教育科技有限公司共同研发了《Java 程序设计项目教程》一书。

本书由安博(大连)教育科技有限公司提供案例样本和内容指导。由高职院校一线教师负责编写统稿定稿。内容以项目为组织单元,由任务驱动。全书共分 11 个项目,每个项目由若干个任务组成。项目 1“Java 软件开发相关技术简介”;项目 2“Eclipse 基本操作”;项目 3“Java 基础”;项目 4“基本控制结构”;项目 5“类和对象”;项目 6“类的继承与多态”;项目 7“异常处理和多线程”;项目 8“输入输出流”;项目 9“图形用户界面设计”;项目 10“键盘与鼠标事件的响应”;项目 11“数据库编程”;附录配有全书练习与提高的参考答案。每个项目配有实训,以提高和锻炼学生的动手能力,还有练习与提高,以检验学生对知识的掌握程度。

本书内容安排合理,逻辑性强,讲解循序渐进,通俗易懂,既适合高职高专计算机及相关专业学生使用,也适合各种培训使用。

本书由彭德林、迟国栋任主编,卢凤伟、何鑫、石芳堂任副主编,金忠伟、薛永三主审。全书由彭德林、迟国栋主持策划并审阅定稿。项目 1、3、5 由韩芝萍编写,项目 2 由何鑫编写,项目 4、6、7 由卢凤伟编写,项目 8 由孙希强编写,项目 9、10、11 由迟国栋编写,附录由彭德林、孙希强编写,部分章节案例素材由安博(大连)教育科技有限公司提供,总经理樊壮、总监丛敏楠参加了相关项目的编写和校对,金忠伟教授、薛永三教授对全书做了审核校对工作。

本书在编写过程中得到了出版社有关领导和编辑的大力支持和帮助,在此一并表示感谢。由于编者水平有限,书中难免存在错误和不足,敬请广大读者和同仁给予批评指正。

编 者

2014 年 6 月

目 录

前言

项目 1 Java 软件开发相关技术简介..... 1	2.4.2 编译和运行程序..... 26
任务一 掌握 Java 语言基础知识..... 1	2.4.3 调试程序..... 27
1.1.1 Java 语言的发展..... 1	项目实战 Eclipse 的安装配置及 Java 程序
1.1.2 Java 程序开发与运行环境..... 2	的调试运行..... 28
1.1.3 Java 语言特点..... 2	项目小结..... 29
1.1.4 Java 程序分类..... 4	练习与提高二..... 29
1.1.5 JDK 的安装与环境变量配置..... 4	项目 3 Java 基础..... 31
1.1.6 Java Application 程序结构及运行过程..... 6	任务一 掌握 Java 关键字、标识符..... 31
任务二 掌握 Java 语言体系..... 8	3.1.1 Java 语言的关键字..... 31
1.2.1 J2SE 概述..... 8	3.1.2 Java 程序的注释..... 32
1.2.2 J2EE 概述..... 8	3.1.3 Java 语言标识符和命名约定..... 32
1.2.3 J2ME 概述..... 12	3.1.4 Java 程序中的分隔符..... 32
任务三 了解 Java 开发工具..... 13	任务二 掌握 Java 语言的基本数据类型、
1.3.1 JBuilder 简介..... 13	常量和变量..... 33
1.3.2 Eclipse 简介..... 13	3.2.1 基本数据类型..... 33
1.3.3 其他开发工具简介..... 14	3.2.2 常量..... 34
项目实战 JDK 安装配置及 Java 程序	3.2.3 变量的定义及作用范围..... 34
的编译运行..... 15	3.2.4 基本数据类型间的转换..... 36
项目小结..... 16	任务三 掌握 Java 运算符和表达式..... 37
练习与提高一..... 16	3.3.1 运算符..... 37
项目 2 Eclipse 基本操作..... 18	3.3.2 表达式..... 43
任务一 了解 Eclipse 概况..... 18	任务四 掌握 Java 数组及字符串的使用方法..... 43
2.1.1 Eclipse 的历史与发展..... 18	3.4.1 数组的声明与创建..... 43
2.1.2 Eclipse 的特点..... 18	3.4.2 数组的初始化及数组元素的使用..... 44
任务二 掌握 Eclipse 的安装与配置..... 19	3.4.3 字符数组与字符串 String 类..... 45
2.2.1 Eclipse 下载与安装..... 19	3.4.4 字符串数组..... 46
2.2.2 Eclipse 的多国语言包的安装..... 19	3.4.5 字符串相关操作..... 46
2.2.3 Eclipse 相关配置..... 20	项目实战一 数组的使用..... 51
任务三 熟悉 Eclipse 工作环境..... 21	项目实战二 字符串的使用..... 51
2.3.1 Eclipse 结构体系..... 21	项目小结..... 52
2.3.2 Eclipse 工作台..... 22	练习与提高三..... 52
2.3.3 Eclipse 资源管理..... 24	项目 4 基本控制结构..... 54
任务四 使用 Eclipse 进行 Java 编程..... 25	任务一 掌握 Java 分支结构..... 54
2.4.1 编辑 Java 程序..... 25	4.1.1 if 语句..... 54

4.1.2 switch 语句.....	58	6.3.3 接口的使用.....	112
任务二 掌握 Java 循环结构.....	60	任务四 掌握包的使用.....	113
4.2.1 for 语句.....	60	6.4.1 包的概念.....	113
4.2.2 do-while 语句.....	62	6.4.2 包的使用.....	114
4.2.3 while 语句.....	64	6.4.3 常用系统包.....	115
4.2.4 嵌套循环.....	65	项目实战一 继承编程.....	116
项目实战一 分支结构编程.....	67	项目实战二 多态编程.....	116
项目实战二 循环结构编程.....	68	项目小结.....	117
项目小结.....	69	练习与提高六.....	117
练习与提高四.....	69	项目 7 异常处理和多线程.....	120
项目 5 类和对象.....	71	任务一 掌握 Java 中对异常的处理.....	120
任务一 掌握类和对象的基本知识.....	71	7.1.1 异常类和异常处理机制.....	120
5.1.1 面向对象的基本概念.....	71	7.1.2 程序中异常处理方法.....	122
5.1.2 类的声明.....	73	任务二 掌握程序对多线程的处理.....	128
5.1.3 方法的声明.....	75	7.2.1 多线程的概念.....	128
5.1.4 方法调用及参数传递.....	76	7.2.2 实现多线程的两种方法.....	128
5.1.5 对象的创建和使用.....	80	7.2.3 线程的生命周期和控制.....	129
5.1.6 构造方法及特点.....	82	7.2.4 线程的同步.....	134
5.1.7 对象的初始化.....	85	项目实战一 异常处理编程.....	136
任务二 掌握类的封装.....	86	项目实战二 多线程编程.....	137
5.2.1 访问权限.....	86	项目小结.....	139
5.2.2 类变量和实例变量.....	88	练习与提高七.....	139
5.2.3 类成员方法和函数数学方法.....	90	项目 8 输入输出流.....	142
项目实战 类和对象.....	92	任务一 掌握流及标准输入输出.....	142
项目小结.....	93	8.1.1 流及分类.....	142
练习与提高五.....	93	8.1.2 标准输入.....	144
项目 6 类的继承与多态.....	96	8.1.3 标准输出和标准错误.....	146
任务一 掌握类的继承.....	96	任务二 掌握常用输入输出流的使用方法.....	148
6.1.1 继承的基本概念.....	96	8.2.1 字节流 InputStream 和 OutputStream 类简介.....	148
6.1.2 继承的实现.....	99	8.2.2 字符流 Reader 和 Writer 类简介.....	149
6.1.3 super 与 this 的使用.....	99	8.2.3 FileInputStream 类的使用.....	150
6.1.4 final 类和 final 方法.....	103	8.2.4 FileOutputStream 类的使用.....	152
任务二 掌握类的多态.....	104	8.2.5 FileReader 类和 FileWriter 类的使用.....	154
6.2.1 多态性的含义.....	104	8.2.6 使用 BufferedReader 类实现各种 数据类型的输入.....	157
6.2.2 方法的重载.....	104	任务三 掌握常用目录和文件操作.....	159
6.2.3 方法的覆盖.....	108	8.3.1 获取文件信息和更改文件属性.....	159
任务三 掌握接口的使用.....	112	8.3.2 列目录文件清单.....	162
6.3.1 抽象类和抽象方法.....	112		
6.3.2 接口的声明.....	112		

8.3.3 获取根目录和创建子目录	162	项目小结	207
8.3.4 文件的创建、更名和删除	163	练习与提高九	207
8.3.5 基本数据类型数据的读写	163	项目 10 键盘与鼠标事件的响应	210
8.3.6 使用增加了缓冲的流类	164	任务一 掌握事件处理机制	210
8.3.7 读取文件指定行	164	10.1.1 相关概念	210
项目实战 文件输入输出操作	164	10.1.2 事件处理机制	210
项目小结	166	任务二 掌握对键盘事件响应的方法	213
练习与提高八	166	10.2.1 键盘事件	213
项目 9 图形用户界面设计	169	10.2.2 键盘监听接口 KeyListener	214
任务一 了解与图形处理相关的包	169	10.2.3 键盘适配器类 KeyAdapter	215
9.1.1 java.awt 包	169	任务三 掌握对鼠标事件的响应方法	216
9.1.2 java.swing 包	170	10.3.1 鼠标事件	216
9.1.3 java.awt.event 包	171	10.3.2 使用 MouseListener 接口处理 鼠标事件	217
任务二 掌握 Java Applet 的程序结构及 运行过程	171	10.3.3 使用 MouseMotionListener 接口 处理鼠标事件	218
9.2.1 Applet 类及相关方法	172	项目实战一 键盘事件响应	219
9.2.2 Applet 程序建立及运行过程	174	项目实战二 鼠标事件响应	221
任务三 掌握使用 Swing 组件进行常用 图形界面设计的方法	175	项目小结	222
9.3.1 窗口容器类 JFrame 使用	176	练习与提高十	223
9.3.2 容器的布局	177	项目 11 数据库编程	225
9.3.3 字体与颜色的使用	183	任务一 使用 JDBC 连接数据库	225
9.3.4 按钮	185	11.1.1 JDBC 简介	225
9.3.5 文本框、文本区和标签	185	11.1.2 JDBC 驱动	227
9.3.6 复选框与单选按钮	188	任务二 建立数据源	228
9.3.7 列表框和组合框	191	11.2.1 建立 Access 数据库	228
9.3.8 对话框和文件选择对话框	194	11.2.2 建立数据源	230
9.3.9 菜单	199	任务三 数据库编程	231
任务四 掌握图形的绘制方法	203	11.3.1 数据库编程过程	231
9.4.1 坐标系统	203	11.3.2 数据库编程实例	237
9.4.2 显示字符串	203	项目实战 数据库编程	243
9.4.3 绘制直线与矩形	204	项目小结	243
9.4.4 绘制圆、椭圆与圆弧	204	练习与提高十一	244
9.4.5 绘制多边形与带颜色图形	204	练习与提高参考答案	245
项目实战 图形界面编程综合实训	205	参考文献	258

项目 1

Java 软件开发相关技术简介

项目内容

通过本项目的学习,读者可以掌握 Java 语言的基本情况,了解 Java 开发工具和 Java 语言体系,掌握 JDK 安装配置及 Java 程序的编译运行方法。

任务一 掌握 Java 语言基础知识

1.1.1 Java 语言的发展

Java 是由 Sun 公司开发的新一代编程语言。使用它可在不同计算机、不同操作平台的网络环境中开发软件。不论你使用的是哪一种 WWW 浏览器,哪一种计算机,哪一种操作系统,只要 WWW 浏览器支持 Java,就可以看到生动的页面。Java 正在逐步成为 Internet 应用的主要开发语言,它彻底改变了应用软件的开发模式,带来了自个人计算机出现以来又一次技术革命,为迅速发展的信息世界增添了新的活力。

Sun 的 Java 语言开发小组成立于 1991 年,其目的是开拓消费类电子产品市场,例如交互式电视、烤面包箱等。1994 年,WWW 已如火如荼地发展起来。开发小组意识到 WWW 需要一个中性的浏览器,它不依赖于任何硬件平台和软件平台,实时性较高、可靠安全、有交互功能。于是,开发小组决定用 Java 开发一个新的 Web 浏览器。Java 1.0 版终于在 1996 年年初正式发表。2009 年 4 月,Sun 公司被 Oracle 公司收购。

Java 虽出现的时间不长,但已被业界接受,IBM、Apple、DEC、Adobe、Silicon Graphics、HP、Oracle、Toshiba、Netscape 和 Microsoft 等大公司都购买了 Java 的许可证。Microsoft 还在其 Web 浏览器 Explorer 3.0 版中增加了对 Java 的支持。

另外,众多的软件开发商也开发了许多支持 Java 的软件产品。如 Borland 公司的基于 Java 的快速应用程序开发环境 Latte; Metrowerks 公司和 Natural Intelligence 公司分别开发的基于 Macintosh 的 Java 开发工具; Sun 公司的 Java 开发环境 Java Workshop; Microsoft 也开发出了系列 Java 产品。数据库厂商如 Sybase、Versant、Oracle 都在开发支持 HTML 和 Java 的 CGI (Common Gateway

Interface)。在以网络为中心的计算机时代，不支持 HTML 和 Java，就意味着应用程序的应用范围只能限于同质的环境。

Java 的出现是计算机信息交换的一个重要里程碑。在单机时代，程序进程之间靠共享存储进行变量交换；在网络时代，运行在不同宿主机上的程序按网络协议进行无格式的消息（二进制字符流）交换，消息的语义由交换程序双方维护；在 Java 时代，运行在网络上的程序进程交换的是小应用程序（Applet）。小应用程序是一个对象，由一组属性和方法构成，是一个可执行的实体；不仅有数据的状态，而且有定义在数据上的操作。未来可能进行代理（Agent）交换，代理交换有一定的智能性，是信息交换的更高级阶段。

1.1.2 Java 程序开发与运行环境

Java 不仅提供了一个丰富的语言和运行环境，而且还提供了一个免费的 Java 开发工具集（Java Developers Kits, JDK）。编程人员和最终用户可以利用这些工具来开发 Java 程序或调用 Java 内容。

JDK 包括以下工具：

- javac: Java 语言编译器，输出结果为 Java 字节码。
- java: Java 字节码解释器。
- javap Disassembler: Java 字节码分解程序，本程序返回 Java 程序的成员变量及方法等信息。
- javaprof: 资源分析工具，用于分析 Java 程序在运行过程中调用了哪些资源，包括类和方法的调用次数和时间，以及各数据类型的内存使用情况等。
- javah: C 代码处理工具，用于从 Java 类调用 C++ 代码。
- java Applet Viewer: 小应用程序浏览工具，用于测试并运行 Java 小应用程序。
- java Debugger API: Java 调试工具。
- API Prototype Debugger: Java 调试工具原型。

(1) Java 编译器。

Java 程序的编译程序是 javac.exe。javac 命令将 Java 程序编译成字节码，然后用户可用 Java 解释器来解释执行这些 Java 字节码。Java 程序源码必须存放在后缀为.java 的文件里。对于 Java 程序里的每一个类，javac 都将生成与类相同名称但后缀为.class 的文件。编译器把.class 文件放在.java 文件的同一个目录里，除非用户使用了-d 选项。

(2) Java 调试器。

JDB (Java Debugger) 是 J2SDK 提供的一个调试工具，可以实现单步跟踪、断点设置、监视程序输出情况等功能。在早期 Beta 1 版的 Java 调试器中，JDB 是命令行形式的，如 Sun 公司的 DBX 调试器。用 JDB 来调试 Java 应用程序，在调试之前，要确定应用程序是带选项-g 编译的，例如 javac -g HelloWorld.java。

(3) Java 解释器。

Java 命令是由 Java 编译器 javac 输出的 Java 字节码。classname 参数是要执行的类名称。注意任意在类名称后的参数都将传递给要执行类的 main 函数。Java 执行完 main 函数后退出，除非 main 函数创建了一个或多个线程。如果 main 函数创建其他线程，Java 总是等到最后一个线程退出才退出。

1.1.3 Java 语言特点

Java 语言具有能独立于平台而运行、面向对象、可对动态画面进行设计与操作、坚固等特点，

又具有多线程、内置校验器以防止病毒入侵等功能,所以用于在 Internet 上研制与开发软件时,特别受到用户的欢迎。

1. 简单

由于 Java 的结构类似于 C 和 C++,所以一般熟悉 C 与 C++语言的编程人员稍加学习就不难掌握 Java 的编程技术了。Java 语言省略了 C++语言中所有的难以理解、容易混淆的特性,例如头文件、指针、结构、单元、运算符重载、虚拟基础类等。并且 Java 所具有的自动内存管理机制也大大简化了 Java 程序设计开发过程。

2. 面向对象

简单地讲,面向对象设计就是一种以数据(对象)以及其接口为中心的程序设计技术。面向对象的设计可以说是定义程序模块如何“即插即用”的机制。Java 的面向对象机制实际上可以看作是 C++面向对象机制的延伸。Java 提供了简单的类机制和动态的构架模型,对象中封装了它的状态变量和方法(函数、过程),实现了模块化和信息隐藏;而类则提供了一类对象的原型,通过继承和重载机制,子类可以使用或重新定义父类或者超类所提供的过程,从而实现代码的复用。

3. 自动内存管理

Java 的自动无用内存回收(auto garbage collection)实现了内存的自动管理,因此简化了 Java 程序开发的工作。早期的无用内存回收(garbage collection)对系统资源抢占太多而影响了整个系统的运行,Java2 对无用内存回收进行的改良使 Java 的效率有了很大提高。无用内存回收的工作机制是周期性地自动回收无用存储单元。Java 的自动内存回收机制在简化程序开发的同时,也提高了程序的稳定性和可靠性。

4. 分布计算

Java 为程序开发提供了 Java.net 包,该包提供了一组类,使程序开发者可以轻易实现基于 TCP/IP 的分布式应用系统。此外,Java 还提供了专门针对互联网应用的类库,如 URL、Java mail 等。

5. 稳定性

人们最常见的应用程序错误就是“非法访问 xxx 内存”,其实质是程序指针使用出错。Java 拥有一种指针(pointer)模型,能够排除内存被覆盖和毁损数据的可能性。Java 不采用指针算术法,而是提供真正的数组(array),运行程序下标检查;另外,它也不会发生将一个任意数转换成指针的情形。

6. 安全性

Java 的设计目的是提供一个用于网络/分布式的计算环境。因此,Java 强调安全性,如确保无病毒、小应用程序运行安全控制等。Java 的验证技术以公钥(public-key)加密算法为基础,而且从环境变量、类加载器、文件系统、网络资源和名字空间等方面实施安全策略。

7. 解释执行

Java 解释器(interpreter)可以直接在任何已移植了解释器的机器上解释、执行 Java 字节代码,不需重新编译。当然,其版本向上兼容,因此如果是高版本环境下编译的 Java 字节码,在低版本环境下运行也许会出现问题。

8. 跨异构环境

Java 是网络空间的“世界语”,编译后的 Java 字节码是一种结构中立性(architecture neutral)的目标文件格式,可以在所有提供 Java 虚拟机(JVM)的多种不同主机、不同处理器上运行。

9. 平滑移植

“write once,run every where!”也许是 Java 最诱人的特点。用 Java 开发而成的系统,其移植工作几乎为零,一般情况下只需对配置文件、批处理文件作相应修改即可实现平滑移植。

10. 多线程

Java 的多线程(multithreading)机制使程序可以并行运行。Java 还有一组同步化基本单元,它们是以广泛使用的 C.A.R.Hoare 监视器与条件变量图为基础的。同步机制保证了对共享数据的正确操作。多线程使程序设计者可以用不同的线程分别实现各种不同的行为,而不需要采用全局的事件循环机制,因此使用 Java 语言可以非常轻松地实现网络上的实时交互行为。

11. 异常处理

C 语言程序员大都使用 goto 语句来进行条件跳转,但 Java 编程不支持 goto 语句。Java 采用异常模型使程序的主流逻辑变得更加清晰明了,并且能够简化错误处理工作。

12. 可扩充

Java 目前发布的 J2EE 标准主要为采用 Java 技术为企业提供全面解决方案提供了一个技术规范框架,规划了一个利用现有和未来各种 Java 技术整合解决企业应用的远景蓝图。

1.1.4 Java 程序分类

Java 程序分为两种类型:Java 应用程序(Java Application)和 Java 小应用程序(Java Applet)。Java Application 可以独立运行;Java Applet 不能独立运行,但可以使用 AppletViewer 或其他支持 Java 的浏览器运行。无论哪种 Java 程序,都用扩展名为.java 的文件保存。

Application 和 Applet 是两个概念,前者指的是一个完整程序,后者指的是一个小程序。作为一个完整程序具有一定的独立性,玩过手机游戏的人都知道一个手机游戏对应一个.jar 文件,这个.jar 文件就相当于一个 Application,只需要选中它运行即可游戏,不需要先启动另外一个程序后才能启动它。而 Applet 则需要先启动浏览器后才能运行它。由于较小,Applet 可以被看作一个软件组件,它可以作为一个动态网站的一个组成部分,Applet 的主要用途是完成复杂的逻辑,它在 JavaScript 等脚本语言和 JSP、ASP 等动态网页技术还没出现时负责动态地显示数据,类似微软的 ActiveX 组件。

1.1.5 JDK 的安装与环境变量配置

JDK (Java Developer's Kit)即 Java 开发者工具包,也称为 J2SDK (Java 2 Software Development Kit),是 Sun 公司提供的基础 Java 语言开发工具,该工具软件包含 Java 语言的编译工具、运行工具以及执行程序的环境(即 JRE)。JDK 现在是一个开源、免费的工具,也是其他 Java 开发工具的基础,也就是说,在安装其他开发工具以前,必须首先安装 JDK。对于初学者来说,使用该开发工具进行学习,可以在学习的初期把精力放在对 Java 语言语法的学习上,体会更多底层的知识,对于以后的程序开发会很有帮助。

但是 JDK 未提供 Java 源代码的编写环境,这个是 Sun 提供的很多基础开发工具的通病,所以实际的代码编写还需要在其他的文本编辑器中进行。其实大部分程序设计语言的源代码都是一个文本文件,只是存储的后缀名不同罢了。

常见的适合 Java 的文本编辑器有很多,例如 JCreator、EditPlus、UltraEdit 等。下面依次介绍 JDK 的下载、安装和环境配置。

1. JDK 的下载

如果需要获得最新版本的 JDK，可以到 Oracle 公司的官方网站上进行下载。下载最新版本的 JDK，选择对应的操作系统，以及使用的语言即可。

2. JDK 的安装

Windows 操作系统上的 JDK 安装程序是一个可执行程序，直接安装即可，在安装过程中可以选择安装路径以及安装的组件等，如果没有特殊要求，可选择默认设置。程序默认的安装路径在 C:\Program Files\Java 目录下。

3. JDK 环境变量的配置

Java 需要设置的环境变量有 3 个。

(1) 变量名 JAVA_HOME，变量值 “C:\Program Files\Java\jdk1.7”。

变量值为安装的 JDK 路径，在该路径下你应该能够找到 bin、lib 等目录，请根据自己的实际情况填写。本例中的设置就应为 “JAVA_HOME=C:\ProgramFiles\Java\jdk1.7”。

注意：值后面不要加分号。

(2) 变量名 PATH，变量值 “%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;”。

设置 PATH 的目的是为了指向 JDK 的 bin 目录，在 bin 目录下存放的是各种编译执行命令。本例中的设置就应该为 “PATH=%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;”。需要说明的是，系统本身就有 PATH 环境变量，只要把 “%JAVA_HOME%\bin;%JAVA_HOME%\jre\bin;” 直接放到后面即可，中间使用 “;” 隔开。

(3) 变量名 CLASSPATH，变量值 “.;%JAVA_HOME%\bin;%JAVA_HOME%\lib\tools.jar;”。

CLASSPATH 用来设置类的路径，十分重要，所有环境变量配置好后，在命令窗口中直接执行 java 或者 javac 命令，出现相应的信息，说明配置成功。配置环境变量的步骤如下：

(1) 右击“我的电脑”图标，单击“属性”命令，选择“高级”选项卡，单击“环境变量”按钮，如图 1-1 所示。

(2) 在“系统变量”区域中，设置 3 项属性：JAVA_HOME、path、Classpath（大小写不影响）。若已存在则单击“编辑”按钮，不存在则单击“新建”按钮，如图 1-2 所示。

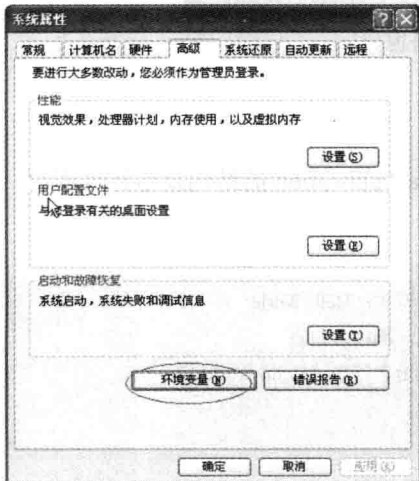


图 1-1 “系统属性”对话框

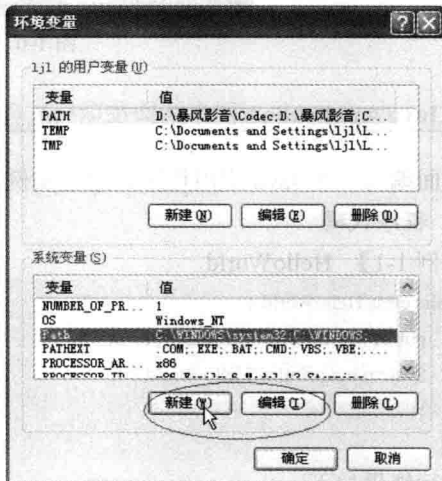


图 1-2 “环境变量”对话框

(3) JAVA_HOME 变量指明 JDK 安装路径: C:\Program Files\Java\jdk1.7, 就是刚才安装时所选择的路径, 此路径下包括 lib、bin、jre 等文件夹, 如图 1-3 所示。

(4) path 变量使系统可以在任何路径下识别 Java 命令, 其值设置如图 1-4 所示。

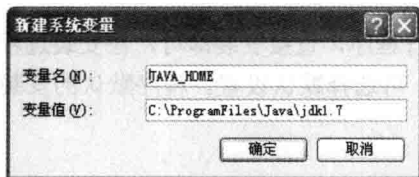


图 1-3 设置 JAVA_HOME

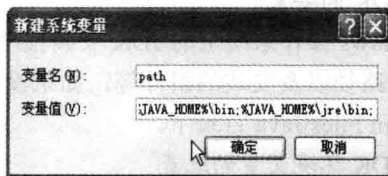


图 1-4 设置 path

(5) Classpath 为 Java 加载类 (class 或 lib) 路径, 只有类在 Classpath 中, Java 命令才能识别, 设为 “.;%JAVA_HOME%\lib;%JAVA_HOME%\lib\tools.jar” (要加 “.” 表示为当前路径)。“%JAVA_HOME%” 就是引用前面指定的 JAVA_HOME, 如图 1-5 所示。

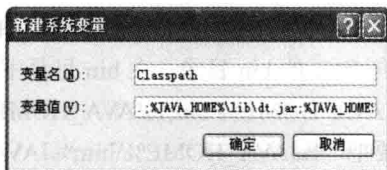


图 1-5 设置 Classpath

(6) 单击“开始”按钮, 选择“运行”项, 键入 cmd, 在命令提示符窗口中键入命令 “java -version”, 出现如图 1-6 所示画面, 说明环境变量配置成功。

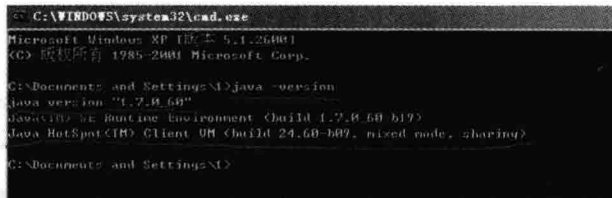


图 1-6 命令提示符窗口

1.1.6 Java Application 程序结构及运行过程

下面编写一个 Java 应用程序, 通过实例来掌握 Java Application 的结构及运行过程。

1. 程序代码

【例 1-1】 HelloWorld

```
public class HelloWorld {           // 声明一个公有类: HelloWorld
    public static void main(String[] args) { // 类中主方法, 程序入口点
        System.out.println("Hello World!"); // 在屏幕上输出字符串 "Hello World !"
    }
}
```

运行结果如下:

Hello World !

2. 程序解析

(1) 每个 Java 程序至少包含一个类的声明。一个 Java 类由类的声明和类体两部分组成。

例 1-1 程序代码中的第 1 行“`public class HelloWorld {`”开始了一个公有类 `HelloWorld` 的声明。

- `class` 关键字引出 Java 的类声明，其后面直接跟上类名 `HelloWorld`。
- `public` 关键字用来修饰类，表示类的访问权限是公有的，对于公有的类则可以被其他任意类使用。
- 类体部分则由大括号“`{`”括起来，此处只定义一个 `main()` 成员方法。

(2) 成员方法由方法声明和方法体两部分构成。

例 1-1 程序代码中的第 2 行“`public static void main(String args[ ])`”为主方法 `main()` 的声明部分。

- `public` 关键字用来修饰方法，表示方法的访问权限是公有的，对于公有的方法则可以被其他任意类调用。
- `static` 关键字指明该方法是一个类方法，它可以通过类名直接调用，由 JVM 要求通过类直接调用 `main()` 方法，所以 `main()` 方法必须声明为类方法（`static` 方法）。
- `void` 则指明 `main()` 方法的返回值类型为空值（不返回值）。
- `main()` 方法首部圆括号中的 `String args[ ]` 是传递给 `main()` 方法的参数，参数名为 `args[ ]`，但其名字允许改变，`[ ]` 亦可置于 `args` 与 `String` 之间，它是一个 `String` 类型的一维数组引用，可以接受命令行中的 0 个或多个字符串类型的实际参数。
- `main()` 方法是运行 Java 应用程序的入口点，含有 `main()` 方法的类通常称为主类。

(3) 标准输出流。

`main()` 方法体部分由大括号“`{`”括起来，此处 `main()` 方法体中只调用了“`System.out.println("Hello World !");`”这一条语句，用来在显示器屏幕上输出一行字符串文本信息；`System` 是一个预定义的类，它提供对系统的访问。

- `out` 是 `System` 类中的一个 `PrintStream` 类型的类成员变量，它代表标准输出流对象，将输出流传送到屏幕上。
- `out` 对象的 `println()` 方法的功能是输出字符串后将光标跳至下一行行首，它能实现与 C 语言中的 `printf` 语句和 C++ 中的 `cout<<` 语句类似的一些功能。

3. Java Application 程序的结构特点

(1) 一个 Java Application 程序由一个或多个文件组成，每个文件中可以定义一个或多个类，每个类由若干个方法和变量组成。

(2) 一个文件中定义多个类时，允许其中声明 0 个或 1 个 `public` 类，若有 `public` 类则程序文件名必须与 `public` 类的类名相同，并区分大小写，扩展名为 `.java`。

(3) 一个 Java Application 程序仅有一个主方法 `main()`，是整个程序的入口。

4. Java 程序编写的注意事项

- 程序名必须与 `public` 类同名；Java 程序区分大小写字母；Java 程序中所有的方法都是属于某个类的，没有不属于某个类的方法。
- 根据 Java 命名规范的约定，Java 所有的类名都以一个大写字母开头，由多个词构成类名时每个词首字母大写。
- 程序中以“`//`”开头的语句注释，称之为行注释，它不影响程序的编译与运行。
- Java 程序的书写格式很自由，一般采用紧缩对齐格式进行书写，Java 语句用分号“`;`”作

为语句的分隔标记，一般一行写一条语句，也允许一行写多条语句，或一条语句分成多行书写。

5. Java 程序的编辑

编辑 Java 程序可以使用任何一种文本编辑器，例如 UltraEdit、Notepad、Wordpad，然后只要把编辑好的文件保存为 .java 文件。当然也可以用一些集成开发环境完成编辑，例如 Borland 公司的 JBuilder。

6. Java Application 的执行

当编译结束以后，Java 源文件中的每一个类都会生成相应的 .class 文件，例如上例中就会生成一个 HelloWorld.class 文件，Java 程序在执行时调用的就是 .class 文件。Java Application 的执行是在命令行下进行的，如果是在 Windows 系统中，就可以在命令提示符下敲入“java HelloWorld”进行执行，该 Java 命令会启动 Java 虚拟机，并读入 HelloWorld.class 文件执行。

任务二 掌握 Java 语言体系

1.2.1 J2SE 概述

Java 2 平台包括标准版 (J2SE)、企业版 (J2EE) 和微缩版 (J2ME) 三个版本，也就是 Sun ONE (Open Net Environment) 体系。J2SE 就是 Java 2 的标准版，主要用于桌面应用程序的编程；J2ME 主要应用于嵌入式系统开发，如手机和 PDA 的编程；J2EE 是 Java 2 的企业版，主要用于分布式的网络程序的开发，如电子商务网站和 ERP 系统。

Standard Edition (标准版)——J2SE 包含那些构成 Java 语言核心的类，如数据库连接、接口定义、输入/输出、网络编程。

Enterprise Edition (企业版)——J2EE 包含 J2SE 中的类，并且还包含用于开发企业级应用的类，如 EJB、Servlet、JSP、XML、事务控制。

Micro Edition (微缩版)——J2ME 包含 J2SE 中的一部分类，用于消费类电子产品的软件开发，如呼机、智能卡、手机、PDA、机顶盒。

J2SE 包含于 J2EE 中，而 J2ME 包含了 J2SE 的核心类，并添加了一些专有类。

J2SE、J2EE、J2ME 的应用场合，API 的覆盖范围各不相同。笼统地讲，可以这样理解：J2SE 是基础；压缩一点，再增加一些 CLDC 等方面的特性就是 J2ME；扩充一点，再增加一些 EJB 等企业应用方面的特性就是 J2EE。

J2EE 更恰当地说，应该是 Java 2 企业开发的技术规范，不仅仅是比标准版多了一些类。J2EE 还包括许多组件，如 JSP、Servlet、JavaBean、EJB、JDBC、JavaMail 等。

J2SE 是 J2EE 的基础，它大量的 JDK 代码库是每个要学习 J2EE 的编程人员必须掌握的。

从 JDK 5.0 开始，J2SE 更名为 Java SE，因为那个“2”已经失去了意义。

1.2.2 J2EE 概述

J2EE 即 Java 2 平台企业版 (Java 2 Platform Enterprise Edition)。J2EE 是一套全然不同于传统应用开发的技术架构，包含许多组件，可简化且规范应用系统的开发与部署，进而提高可移植性、安全性与重用价值。

J2EE 的核心是一组技术规范与指南,其中所包含的各类组件、服务架构及技术层次,均有共通的标准及规格,让各种依循 J2EE 架构的不同平台之间存在良好的兼容性,解决了过去企业后端使用的信息产品彼此之间无法兼容,导致企业内部或外部难以互通的窘境。

1. J2EE 的概念

J2EE 是一种利用 Java 2 平台来简化企业解决方案的开发、部署和管理相关的复杂问题的体系结构。J2EE 技术的基础就是核心 Java 平台或 Java 2 平台的标准版, J2EE 不仅巩固了标准版中的许多优点,例如“编写一次、随处运行”的特性,方便存取数据库的 JDBC API、CORBA 技术以及能够在 Internet 应用中保护数据的安全模式等,同时还提供了对 EJB (Enterprise JavaBeans)、Java Servlets API、JSP (Java Server Pages) 以及 XML 技术的全面支持。其最终目的就是成为一个能够使企业开发者大幅缩短投放市场时间的体系结构。

J2EE 体系结构提供中间层集成框架,用于满足无需太多费用而又需要高可用性、高可靠性以及可扩展性的应用的需求。通过提供统一的开发平台, J2EE 降低了开发多层应用的费用和复杂性,同时提供对现有应用程序集成的强有力支持,完全支持企业应用。

JavaBeans 有良好的向导支持打包和部署应用,添加目录支持,增强了安全机制,提高了性能。

2. J2EE 的优势

J2EE 为搭建具有可伸缩性、灵活性、易维护性的商务系统提供了良好的机制。

(1) 保留现存的 IT 资产: 由于企业必须适应新的商业需求,因此利用已有的企业信息系统方面的投资,而不是重新制定全盘方案就变得很重要。这样,一个以渐进的(而不是激进的、全盘否定的)方式建立在已有系统之上的服务器端平台机制是公司所需求的。J2EE 架构可以充分利用用户原有的投资,如一些公司使用的 BEA Tuxedo、IBM CICS、IBM Encina、Inprise VisiBroker 以及 Netscape Application Server。其之所以成功的原因之一是因为 J2EE 拥有广泛的业界支持和一些重要的“企业计算”领域供应商的参与。每一个供应商都对现有的客户提供了不用废弃已有投资进入可移植的 J2EE 领域的升级途径。由于基于 J2EE 平台的产品几乎能够在任何操作系统和硬件配置上运行,现有的操作系统和硬件也能被保留使用。

(2) 高效的开发: J2EE 允许公司把一些通用的、很烦琐的服务端任务交给中间件供应商去完成,这样开发人员可以集中精力在如何创建商业逻辑上,相应地缩短了开发时间。

(3) 支持异构环境: J2EE 能够开发部署在异构环境中的可移植程序。基于 J2EE 的应用程序不依赖任何特定操作系统、中间件和硬件。因此设计合理的基于 J2EE 的程序只需开发一次就可部署到各种平台。这在典型的异构企业计算环境中是十分关键的。J2EE 标准也允许客户订购与 J2EE 兼容的第三方的现成的组件,把它们部署到异构环境中,节省由自己制订整个方案所需的费用。

(4) 可伸缩性: 企业必须要选择一种服务器端平台,这种平台应能提供极佳的可伸缩性去满足那些在它们系统上进行商业运作的大批新客户。基于 J2EE 平台的应用程序可被部署到各种操作系统上。例如可被部署到高端 UNIX 与大型机系统,这种系统单机可支持 64~256 个处理器(这是 NT 服务器所望尘莫及的)。J2EE 领域的供应商提供了更为广泛的负载平衡策略,它能消除系统中的瓶颈,允许多台服务器集成部署。这种部署可多达数千个处理器,实现可高度伸缩的系统,满足未来商业应用的需要。

(5) 稳定的可用性: 一个服务器端平台必须能全天候运转以满足公司客户、合作伙伴的需要。因为 Internet 是全球化的、无处不在的,即使在夜间按计划停机也可能造成严重损失。若是意外停机,那将会有灾难性后果。将 J2EE 部署到可靠的操作环境中,它们支持长期的可用性。一些 J2EE

部署在 Windows 环境中, 客户也可选择健壮性能更好的操作系统, 如 Sun Solaris、IBM OS/390。最健壮的操作系统可达到 99.999% 的可用性或每年只需 5 分钟停机时间。这是实时性很强的商业系统的理想选择。

3. J2EE 的四层模型

J2EE 使用多层的分布式应用模型, 应用逻辑按功能划分为组件, 各个应用组件根据它们所在的层分布在不同的机器上。事实上, Sun 设计 J2EE 的初衷正是为了解决两层模式 (client/server) 的弊端。在传统模式中, 客户端担当了过多的角色而显得臃肿, 在这种模式中, 第一次部署的时候比较容易, 但难于升级或改进, 可伸展性也不理想, 而且经常基于某种专有的协议, 通常是某种数据库协议。它使得重用业务逻辑和界面逻辑非常困难。现在 J2EE 的多层企业级应用模型将两层化模型中的不同层面切分成许多层。一个多层化应用能够为不同的每种服务提供一个独立的层, 以下是 J2EE 典型的四层结构:

- 运行在客户端机器上的客户层组件;
- 运行在 J2EE 服务器上的 Web 层组件;
- 运行在 J2EE 服务器上的业务逻辑层组件;
- 运行在 EIS 服务器上的企业信息系统 (Enterprise Information System) 层软件。

4. J2EE 的结构

这种基于组件、具有平台无关性的 J2EE 结构使得 J2EE 程序的编写十分简单, 因为业务逻辑被封装成可复用的组件, 并且 J2EE 服务器以容器的形式为所有的组件类型提供后台服务。你不用自己开发这种服务, 所以可以集中精力解决手头的业务问题。

容器设置定制了 J2EE 服务器所提供的内在支持, 包括安全、事务管理、JNDI (Java Naming and Directory Interface) 寻址、远程连接等服务, 以下列出最重要的几种服务:

J2EE 安全 (Security) 模型可以让你配置 Web 组件或 Enterprise Bean, 这样只有被授权的用户才能访问系统资源。每一客户属于一个特别的角色, 而每个角色只允许激活特定的方法。你应在 Enterprise Bean 的布置描述中声明角色和可被激活的方法。依靠这种声明性的方法, 你不必编写加强安全性的规则。

J2EE 事务管理 (Transaction Management) 模型让你指定组成一个事务所有方法间的关系, 这样一个事务中的所有方法被当成一个单一的单元。当客户端激活一个 Enterprise Bean 中的方法时, 容器介入管理事务。因为有容器管理事务, Enterprise Bean 中不必对事务的边界进行编码。要求控制分布式事务的代码会非常复杂。你只需在布置描述文件中声明 Enterprise Bean 的事务属性, 而不用编写并调试复杂的代码。容器将读此文件并为你处理此 Enterprise Bean 的事务。

JNDI 寻址 (JNDI Lookup) 服务为企业内的多重名字和目录服务提供了一个统一的接口, 这样应用程序组件可以访问名字和目录服务。

J2EE 远程连接 (Remote Client Connectivity) 模型管理客户端和 Enterprise Bean 间的低层交互。当一个 Enterprise Bean 创建后, 一个客户端可以调用它的方法, 就像它和客户端位于同一虚拟机上一样。

生存周期管理 (Life Cycle Management) 模型管理 Enterprise Bean 的创建和移除, 一个 Enterprise Bean 在其生存周期中将会历经几种状态。容器创建 Enterprise Bean, 并在可用实例池与活动状态中移动它, 最终将其从容器中移除。即使可以调用 Enterprise Bean 的 create 及 remove 方法, 容器也将会在后台执行这些任务。