

全国机械行业高等职业教育“十二五”规划教材
高等职业教育教学改革精品教材

开发基于Java语言的 银行卡模拟系统

KAIFA JIYU Java YUYAN DE
YINHANGKA MONI XITONG

孙华林 主编



 机械工业出版社
CHINA MACHINE PRESS



针对教材使用者

赠 电子教案

全国机械行业高等职业教育“十二五”规划教材

高等职业教育教学改革精品教材

开发基于 Java 语言的银行卡 模拟系统

主 编 孙华林

副主编 印 梅 吴红亚

参 编 赵建成 陈爱民

主 审 苏宝莉



机械工业出版社

本书为高等职业教育计算机类专业用教材。

本书采用一个完整的实例——银行卡模拟系统，全面讲述了 Java 语言面向对象的编程技术。从在 Eclipse 集成开发环境中开发一个简单的 Java 应用程序开始，循序渐进、由浅入深，结合生动的银行卡模拟系统实例，详尽地介绍了 Java 语言的基本要素、数组、字符串、类、对象、构造函数、方法的重载和覆盖、封装、继承、多态等。另外，还讲解了 Java 语言的异常处理机制和采用 JDBC 技术访问数据库的方法和步骤。读者可以通过对本书的学习，逐步掌握 Java 语言面向对象的编程技术。

本书可作为高等职业技术学院、各类计算机培训学校、高等专科学校、成人高校、本科院校承办的二级职业技术学院的计算机相关专业 Java 语言课程的教材，还可供各类计算机软件开发人员入门学习时使用。

图书在版编目 (CIP) 数据

开发基于 Java 语言的银行卡模拟系统/孙华林主编. —北京: 机械工业出版社, 2011.7
全国机械行业高等职业教育“十二五”规划教材 高等职业教育教学改革精品教材
ISBN 978-7-111-34208-3

I. ①开… II. ①孙… III. ①Java 语言—程序设计—高等职业教育—教材②信用卡—模拟系统—中国—高等职业教育—教材 IV. ①TP312 ②F832.2-29

中国版本图书馆 CIP 数据核字 (2011) 第 071601 号

机械工业出版社 (北京市百万庄大街 22 号 邮政编码 100037)

策划编辑: 崔占军 边 萌 责任编辑: 崔占军 边 萌 杨作良

封面设计: 鞠 杨 责任印制: 李 妍

北京外文印刷厂印刷

2011 年 5 月第 1 版第 1 次印刷

184mm×260mm·11.5 印张·281 千字

0 001—3 000 册

标准书号: ISBN 978-7-111-34208-3

定价: 23.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

电话服务

网络服务

社服务中心: (010) 88361066

门户网: <http://www.cmpbook.com>

销 售 一 部: (010) 68326294

教材网: <http://www.cmpedu.com>

销 售 二 部: (010) 88379649

读者购书热线: (010) 88379203

封面无防伪标均为盗版

前 言

Java 语言作为面向对象的编程语言，以其不依赖于硬件及操作系统环境并具有超强的安全性等特点受到了广大软件开发者的广泛关注，Java 已经成为网络时代最重要的语言之一。

然而当前，软件人才是困扰 IT 企业发展的第一大问题，企业之间的竞争正在变成人才的竞争。同时，很多高校开设的软件课程的教学内容和企业的真正需求脱节，学生在完成大学学业时只掌握一些简单的 Java 基础知识，没有从事 Java 项目开发的经验，碰到问题时不知道如何下手，这样培养出来的学生不能胜任 IT 企业要求的“掌握实用技术并且能解决问题”的软件工程师的职位。

这就需要一个好的引导者，引导学生与 IT 企业的应用紧密结合，在仿真的环境中学习和掌握实际实例，融理论、实践和技能为一体，真正做到学以致用。

本书正是本着上述目的而编写的，本课程遵循以下原则。

一、以实用技能为基础

软件开发人员都知道，软件开发领域所涉及的技术和知识点非常多，不同的行业、不同的项目都会使用不同的技术。而 Java 平台所涉及的技术及知识点更是让人眼花缭乱。现在市场上很多软件开发方面的书籍都是繁杂技术知识点的简单罗列，学生，尤其是初学者阅读上述书籍时往往不知道从何学起，甚至很多初学者研读完这样的书籍后，由于技术知识点繁多而不知道如何在项目中运用这些技术知识点。本书将把实际应用中一些常用的技术知识点全部运用到实例中，而不只是技术知识点的简单罗列。以用为本、学以致用，使学生在实例中感受并尝试新知识点的用法。

二、以实例为主线

本书尝试以实例驱动、以典型的实例开发过程为主线，以一个完整的银行卡模拟系统——Bank 系统为主线贯穿全书。将相关琐碎的技术知识点的讲解贯穿在完整的实例开发过程中，通过具体的实施步骤完成预定的实例开发流程和任务，从而掌握相关技术知识点及其用法。

三、采用全新的“1221”模式下的理论教学和实践教学两个体系系统的编写模式

无论是理论教学还是上机（实践）教学，均从项目入手，每章内容围绕实例/项目展开，将各知识点的介绍融入到实例/项目的解决方案中。教师好讲，学生好学，学生动手能力能得到很好的训练，学完本书将完成一个完整的项目实例。当然，程序员们都知道，现在的软件开发其实是对现实世界的生产、生活的模拟过程，因此在实例的选择上，本书尽量在考虑实用性的同时，尽可能地增强实例的趣味性，并加强与日常生活中遇到的问题和现象的联系，从而帮助学生加深理解。

四、以提高动手能力为核心

本课程以提高学生的动手能力、在学习 Java 基础知识点的同时增加学生项目开发经验为目标。鼓励学生勇于编码、乐于编码，同时大量反复地动手实践，让他们在学习的过程中逐步具备熟练、规范的编码以及调试的能力，成为一名企业真正需要的人才。

本书内容丰富全面，全书共分为 8 章和一个 Java 常用集合类、接口专题以及 String、

String Buffer 类专题。其中第 1 章首先阐述如何开发一个简单的 Java 小程序；第 2 章为 Java 语言基础知识，包括 Java 语言的基本要素和语句控制；第 3 章结合 Bank 系统介绍了数组和字符串的用法；第 4、5、6 章全面介绍了面向对象的基本常识，并结合 Bank 系统详细地讲解了类、对象、继承、多态、接口等用法；第 7 章引入了 Java 语言的异常处理机制；第 8 章详细讲解了使用 JDBC 访问数据库的方法。

本书配有电子教案和所有实例的源代码等相关资源，以方便教师教学。

本书由孙华林任主编，印梅、吴红亚任副主编。全书由孙华林统稿，苏宝莉任主审。其中第 1、2 章由吴红亚编写，第 3、4 章和第 8 章由印梅编写，第 5、6、7 章以及 Java 常用的集合类及接口专题以及 **String**、**StringBuffer** 类专题由孙华林编写。另外，赵建成和陈爱民参与编写了第 8 章和 Java 常用的集合类及接口专题。

本书可作为高等职业技术学院、各类计算机培训学校、高等专科学校、成人高校、本科院校承办的二级职业技术学院的计算机相关专业 Java 语言课程的教材，还可供各类计算机软件开发人员学习时使用。

由于作者水平有限，在内容及结构上难免存在错误和不足之处，恳请各位同行和广大读者给予批评指正。

编 者

目 录

前言

第 1 章 开发简单的 Java 应用程序 1

1.1 Java 语言简介.....	1
1.1.1 Java 语言的产生.....	2
1.1.2 Java 语言的特色.....	2
1.2 开发第一个 Java 程序.....	3
1.2.1 开发 Java 程序的准备.....	3
1.2.2 第一个 Java 程序的说明.....	5
1.3 Eclipse 的使用.....	6
1.4 包.....	8
1.4.1 包的创建.....	8
1.4.2 包的引用.....	8
1.4.3 输入.....	9
1.4.4 为程序加上注释.....	11
1.5 上机实践.....	11
1.5.1 上机目标与训练技能点.....	11
1.5.2 上机任务.....	12
小结.....	12
习题.....	13

第 2 章 Java 语言基础..... 14

2.1 Java 语言基本要素.....	14
2.1.1 标识符.....	14
2.1.2 关键字.....	14
2.1.3 数据类型.....	15
2.1.4 变量.....	15
2.1.5 数据类型转换.....	16
2.1.6 运算符与表达式.....	16
2.2 程序控制语句.....	19
2.2.1 顺序结构.....	19
2.2.2 选择结构.....	19
2.2.3 循环结构.....	24
2.3 在 Eclipse 中调试程序.....	29
2.4 上机实践.....	30
2.4.1 上机目标与训练技能点.....	30
2.4.2 上机任务.....	30

小结.....	32
习题.....	32

第 3 章 在 Bank 系统中使用数组和字符串 35

3.1 数组.....	35
3.1.1 一维数组.....	35
3.1.2 多维数组.....	38
3.2 字符串.....	40
3.2.1 字符串的创建.....	40
3.2.2 字符串的连接.....	41
3.3 上机实践.....	43
3.3.1 上机目标与训练技能点.....	43
3.3.2 上机任务.....	43
小结.....	45
习题.....	45

第 4 章 在 Bank 系统中引入类和对象 47

4.1 类和对象.....	47
4.1.1 类的创建.....	48
4.1.2 类的变量和方法.....	49
4.1.3 对象的创建.....	51
4.2 构造函数.....	53
4.3 方法重载.....	56
4.4 上机实践.....	57
4.4.1 上机目标与训练技能点.....	57
4.4.2 上机任务.....	58
小结.....	64
习题.....	64

第 5 章 在 Bank 系统中引入继承和多态 66

5.1 何谓继承.....	66
5.1.1 自然界中的继承现象.....	66
5.1.2 Java 语言中类的继承的实现.....	67
5.2 成员变量的隐藏.....	69

5.3 成员方法的覆盖	71
5.4 super 和 this 的使用	73
5.5 多态性	74
5.5.1 何谓多态	74
5.5.2 Java 语言中如何实现多态	74
5.6 上机实践	76
5.6.1 上机目标与训练技能点	76
5.6.2 上机任务	76
小结	83
习题	84
第 6 章 访问权限控制及接口	86
6.1 类及类中成员的访问权限控制	86
6.2 修饰符	91
6.2.1 final 修饰符	91
6.2.2 static 修饰符	93
6.2.3 abstract 修饰符	94
6.3 Java 语言的接口	95
6.3.1 接口的定义和实现	95
6.3.2 如何定义接口中的常量	97
6.3.3 使用接口的好处	97
6.3.4 面向接口编程	100
6.4 上机实践	103
6.4.1 上机目标与训练技能点	103
6.4.2 上机任务	104
小结	109
习题	110

第 7 章 Java 语言的异常处理机制 在 Bank 系统中的使用	112
7.1 何谓异常	112
7.1.1 现实生活中的异常	112

7.1.2 程序运行中的异常	113
7.1.3 异常的定义	114
7.2 异常处理	114
7.2.1 如何进行异常处理	114
7.2.2 编写代码处理异常情况	115
7.2.3 使用 Java 语言的异常处理机制处理 异常情况	116
7.2.4 用户自定义异常	122
7.3 上机实践	124
7.3.1 上机目标与训练技能点	124
7.3.2 上机任务	125
小结	127
习题	128

第 8 章 基于 SQL Server 2005 数据库 继续升级 Bank 系统

8.1 JDBC 概述	130
8.2 JDBC 驱动	131
8.3 使用 JDBC 进行数据库编程的模板	131
8.4 综合示例	135
8.5 上机实践	140
8.5.1 上机目标与训练技能点	140
8.5.2 上机任务	141
小结	157
习题	157

附录

附录 A Java 常用的集合类和集合 接口专题	158
附录 B String 类和 StringBuffer 类专题	163

参考文献

第1章 开发简单的 Java 应用程序

Java 语言越来越流行了，现在被广泛地用在 Internet 环境上开发应用系统。本章将介绍 Java 语言的基本特点以及如何开发一个简单的 Java 应用程序。现在有很多 Java 集成开发环境 (IDE)，本书选用的是 Eclipse，通过使用 IDE 可以提高程序开发的效率，减轻程序员的编程工作。当然，对于初学者，初期最好不要依赖 IDE，基础知识还是要通过文本编辑器慢慢地敲打出来才能更好地扎实掌握。



技能目标

- ☆ 了解 Java 语言的特点。
- ☆ 熟悉 Java 语言开发环境及 Java 程序开发过程。
- ☆ 掌握 Java 中包 (Package) 的使用。
- ☆ 掌握 Java 的标准输入输出。



上机任务

开发简单的 Java 程序：

- ☆ 在实践中掌握开发 Java 程序的步骤。
- ☆ 使用输入输出功能提高程序的互动性。
- ☆ 使用包提高程序的清晰性。

1.1 Java 语言简介

Java 语言于 1995 年 6 月由美国 Sun 微系统公司 (Sun Microsystems) 发明。Java 语言集中体现和充分利用了若干现代软件技术新成果，从而成为一种跨平台、分布式、适于网络应用的面向对象的现代高级编程语言。

(1) Java 语言包含了一种计算机编程语言和一个软件运行平台。

1) Java 编程语言是一种高级语言，由 Sun 微系统公司发布，并作为一种开放的标准提供给用户。

2) Java 平台包括了 Java 虚拟机和 Java 应用程序接口 (API)。

(2) Java 语言的主要应用有如下几方面。

1) 利用浏览器中的 Java 虚拟机运行客户端的小程序 (Applet)。

2) 利用本地虚拟机运行 Java 应用程序，通常利用图形用户界面 (GUI) 框架 Swing 或 SWT 等来编写跨平台的 GUI 系统。

3) 通过编写服务器端应用程序 Servlet/JSP 来提供网络服务, 运行于网络应用程序服务器中。

4) 利用嵌入式 Java 虚拟机运行手机等移动设备中的 Java 应用程序。

1.1.1 Java 语言的产生

Java 语言来自于 Sun 公司的一个叫 Green 的项目, 其原先的目的是为家用消费电子产品开发一个分布式代码系统, 以便实现诸如把 E-mail 发给电冰箱、电视机等家用电器, 对它们进行控制, 和它们进行信息交流等功能。开始, 准备采用 C++, 但 C++ 太复杂, 安全性差, 于是决定基于 C++ 开发一种新的语言 Oak (Java 的前身)。Oak 是一种用于网络的精巧而安全的语言, Sun 公司曾以此投标一个交互式电视项目, 但结果是被 SGI 公司打败。可怜的 Oak 几乎无家可归, 恰巧这时 Marc Andreessen 开发的 Mosaic 和 Netscape 启发了 Oak 项目组成员, 他们用 Java 编制了 HotJava 浏览器, 得到了 Sun 公司首席执行官 Scott McNealy 的支持, 触发了 Java 语言进军 Internet。

Java 的取名也有一个趣闻。有一天, 几位 Java 成员组的会员正在讨论给这个新的语言取什么名字, 当时他们正在咖啡馆喝着 Java (爪哇) 咖啡, 有一个人灵机一动说就叫 Java 怎样, 得到了其他人的赞赏, 于是, Java 这个名字就这样传开了。

根据 API 和使用领域, Java 平台主要分为三种技术:

(1) Java SE (旧称 J2SE) Java Platform, Standard Edition, 定位在客户端, 主要用于桌面应用程序的编程。

(2) Java EE (旧称 J2EE) Java Platform, Enterprise Edition, 定义在服务器端 Java 的企业版, 主要用于分布式网络程序的开发, 如电子商务网站和 ERP (Enterprise Resource Planning) 系统。

(3) Java ME (旧称 J2ME) Java Platform, Micro Edition, 主要应用于嵌入式系统的开发, 如手机和 PDA (Personal Digital Assistant) 的编程。

1.1.2 Java 语言的特色

Java 语言从 1995 年开始在计算机业界就受到了高度注意, 特别是在 Internet 和多媒体 (Multimedia) 相关产品类方面。Java 语言是:

(1) 一种为 Internet 应用而发展的计算机语言。

(2) 一种使网页 (Web Page) 由静态 (Static) 转变为动态 (Dynamic) 的编程语言。

(3) 一种编程语言, 用以产生小程序 (Applets)。

(4) 一种将安全性 (Security) 列为第一优先考虑的编程语言。

(5) 一种突破用户端机器环境和 CPU 结构的编程语言。

(6) 一种“写一次, 即可在任何机器上执行 (Write Once, Run Anywhere)”的语言, 是有史以来第一种允许使用者将应用程序 (Applications) 通过 Internet 从远端的服务器 (Remote Server) 传输到本地端的机器上 (Local Machine) 并执行的编程语言。

(7) 一种应用程序提供者不需要知道使用者的计算机硬件 (如: Sun, Intel 或 MAC 等) 与软件 (如: UNIX, MAC O/S 或 Windows 等) 环境的编程语言。

1.2 开发第一个 Java 程序

在了解 Java 后, 先来试着开发第一个 Java 程序。为了能成功完成一个 Java 程序的开发, 希望大家千万不要急躁, 必须按照如下步骤一步一步地进行。

1.2.1 开发 Java 程序的准备

步骤 1 下载 JDK

JDK (Java Development Kit) 称为 Java 开发包, 是整个 Java 的核心, 包括了 Java 运行环境 (Java Runtime Environment), 一堆 Java 工具和 Java 的基础类库 (rt.jar)。本书选用 Sun 公司的 Windows 版本的 JDK, 可以从 Sun 公司网站 www.sun.com 下载, 下载之后安装 .exe 文件, 比如这就是安装后 JDK 的路径地址  E:\Program Files\Java\jdk1.6.0_06。

补充

JDK 的组成

- javac——编译器, 将源程序转成字节码。
- jar——打包工具, 将相关的类文件打包成一个文件。
- javadoc——文档生成器, 从源码注释中提取文档。
- jdb——debugger, 查错工具。
- java——运行编译后的 java 程序 (.class 扩展名的)。
- appletviewer——小程序浏览器, 一种执行 HTML 文件上的 Java 小程序的 Java 浏览器。

步骤 2 配置环境变量

(1) JAVA_HOME (要大写) 指明 JDK 安装路径, 此路径下包括 lib, bin, jre 等文件夹。此变量最好设置, 因为以后若运行 tomcat 或 MyEclipse 等都需要依靠此变量。

(2) Path 使得系统可以在任何路径下识别 java 命令。

(3) CLASSPATH (要大写) 为 Java 加载类 (class or lib) 路径, 只有类在 CLASSPATH 中, java 命令才能识别, 要加 “.” 表示当前路径。

了解如上一些环境变量的作用后, 就可以设置了。在 Windows 操作系统中, 操作步骤为: 用鼠标右键单击我的电脑→属性→高级→环境变量 (见图 1-1)。

“系统变量→新建→变量名: JAVA_HOME 变量值: E:\Program Files\Java\jdk1.6.0_06;”

“系统变量→新建→变量名: CLASSPATH 变量名: .;%JAVA_HOME%\lib;”

“系统变量→编辑→变量名: Path。”在变量值的最前面加上: “;%JAVA_HOME%\bin;” 或者加在最后面, “;” 写在前面与其他环境变量分隔开。

配置完成后, 在 DOS 下输入 java -version 可以查看 Java 的版本信息。

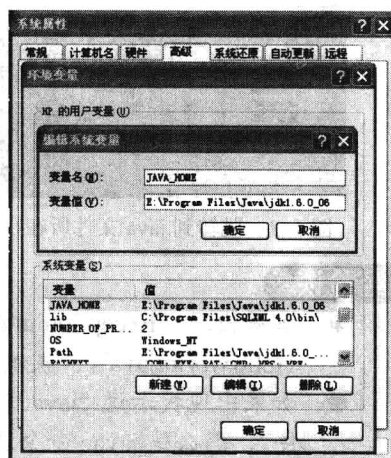


图 1-1 配置环境变量

步骤 3 编写程序

使用一种文本编辑器（比如：记事本、UltraEdit 或 EditPlus）编写如下代码，并保存为文件：HelloWorld.java（必须大小写正确！假设保存路径为 E:\workspace\01\HelloWorld.java）。

例 1-1

```

1  public class HelloWorld
2  {
3      public static void main (String[] args)
4      {
5          System.out.println ("Hello World!");
6      }
7  }

```

步骤 4 编译运行

Java 的源代码是以扩展名为 .java 的文件给出的。源代码送给编译器，比如 Sun 公司的 javac (Java Compiler)，产生字节码 (.class) 文件。字节码是一种在 Java 虚拟机 JVM (Java Virtual Machine) 上执行的代码，执行方式是“解释执行”或“即时编译”。

在 Windows 系统中单击开始→运行，用 cmd 命令开启一个 DOS 窗口，用 cd 命令 (Change Directory) 跳转到 HelloWorld.java 所在的目录中（见图 1-2）。

编译 HelloWorld.java 为 HelloWorld.class:

输入“javac HelloWorld.java”，如果没有错误，将会在 java 文件同一目录下找到 .class 文件；如果检测到错误，请按照相应提示修改源程序，保存后再次编译。

运行 HelloWorld.class:

输入“java HelloWorld”，命令和文件名之间要加空格，不需要加 .class。这样就会看到第一个程序的运行结果，在 DOS 的屏幕上会看到如图 1-3 所示的信息。

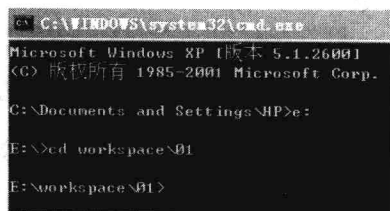


图 1-2 跳转到 .java 文件所在目录

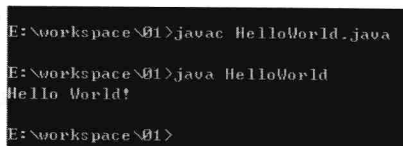


图 1-3 编译和运行程序

经验谈

- 如果出现“Exception in thread ‘main’ java.lang.NoClassDefFoundError: HelloWorld/java”一类的错误，那么是错把 javac 命令写成了 java 命令。
- 如果出现提示说“javac 不是内部或外部命令，也不是可运行的程序或批处理文件”，那么是 javac 命令没起作用，需要重新设置。
- Java 程序可以分为 Java Application（应用程序）和 Java Applet（小程序）两类，Java Application 是由独立的解释器程序来运行的，需要调用 Java 的解释

器软件 java.exe。Java Applet 是一类非常重要的 Java 程序，虽然它的源代码编辑与字节码的编译生成过程与 Java Application 相同，但它却不是一类可以独立运行的程序，其字节码文件必须嵌入到另一种语言——HTML 的文件中并由负责解释 HTML 文件的浏览器充当其解释器，来解释执行 Java Applet 的字节码程序。

1.2.2 第一个 Java 程序的说明

即使是最简单的 Java 程序都会涉及文件管理、类 (class)、主程序、命令行参数等概念，一下子要接受这么多概念并不容易，对于 Java 初学者来说先记住一个基本的 Java 程序按下述步骤来写就行了。

1. 定义类

编写 Java 程序都是由定义“类”(class)开始的。class 是 Java 语言用来定义类的关键字，如例 1-1 中的 HelloWorld，这与所编写的文件(HelloWorld.java)的文件名必须相同。HelloWorld 类的定义中使用了关键字 public，在编写 Java 程序时，一个文件中可编写数个类，但是只能有一个公共 (public) 类，而且文件名必须与这个公共类的名称相同。在定义类名称时，建议将类名首字母大写，并在类名上表明类的作用。

2. 定义程序块

Java 程序使用大括号 {} 来定义程序块，大括号两两成对，目的在于区别定义的作用范围。例如在程序中，HelloWorld 类的程序块包括了 main()，而 main() 的程序块包括了一句显示信息的程序代码。

3. 定义 main() 方法 (Methods)

main() 方法是 Java 程序的“入口”，程序的执行是由入口开始的。类中的方法是类的成员 (Member)，main() 方法一定是 public 成员，这样它才可以在执行环境中被调用。main() 方法不需要产生对象 (Object) 就能被执行，所以它必须是个 static 成员。public 与 static 的概念都是 Java 面向对象 (Object-Oriented) 编程技术中的概念，后面专门讨论类与对象时会再介绍，这里先不用理解 public 与 static 的真正含义。

main() 之前的 void 表示 main() 执行结束后不返回任何值，Java 程序的 main() 方法不需返回任何值，所以一律定义为 void。main() 括号间的 String[] args 可以在执行程序时取得用户指定的命令行参数 (Command Line Argument)。

4. 编写语句 (Statement)

来看 main() 中的一行语句：System.out.println("Hello World!");

语句是程序语言中的一行指令，简单讲就是程序语言的“一句话”。注意每一条语句的结束要用分号“;”。在上面的语句中，使用了 java.lang 包 (Package) 下的 System 类的公共 (public) 成员 out 对象 (out 是一个 PrintStream 对象)，并使用了 PrintStream 所提供的 println() 方法，将其中指定的字符串 (String) “Hello World!” 输出至命令行模式上。

注意在 Java 中字符串要使用英文格式的双引号 “” 包括，println() 表示输出字符串后自动换行，如果使用 print()，则输出字符串后程序并不会自动换行。

1.3 Eclipse 的使用

如果完成自己的第一个 Java 程序之后，发现在开发一个程序的时候需要在不同的软件之间来回切换，那么接下来就为大家介绍如何使用 IDE 来进行程序开发。

IDE (Integrated Development Environment) 即集成开发环境，它将编辑、编译、调试等功能集成在一个桌面环境中，大大方便了开发者。如微软的 Visual Studio 系列，Borland 的 C++ Builder、Delphi 系列等。不同的技术体系有不同的 IDE，比如 Visual Studio.Net 可以称为 C++、VB、C#等语言的集成开发环境，那么 Java 语言的 IDE 有哪些呢？

(1) Eclipse MyEclipse 是在 Eclipse 上集成了插件，一般都用于开发 Java EE。Eclipse 免费，MyEclipse 要收费。

(2) NetBeans 这是 Sun 公司自己的 IDE。

(3) JCreator 这个 IDE 功能简单，适合初学者。

(4) JBuilder 目前是最好用的，也是最贵的 IDE。

本书采用 Eclipse 来开发程序，但建议初学者还是多练习使用文本编辑器编写程序。下面把第一个 Java 程序在 Eclipse 中再实现一遍。

步骤 1 建立 Workspace

Workspace 是用来存放 Java 程序的，当启动 Eclipse 时，会出现 Workspace 的路径设置。假设将 Workspace 设置在 e:\workspace。

步骤 2 建立 Java 工程

此处与在文本编辑器中不同的是，需要先建立 Java 工程才能编写相应的 Java 程序。选择“File→New→Java Project”，在“Project name”这项中输入工程名称，例如“JavaOOP”，不要进行其他设置，直接单击“Finish”按钮，如图 1-4 所示。随后会在左边的视图中看到刚才建立的工程已经存在，如图 1-5 所示，同时存在的还有一个 JRE System Library。

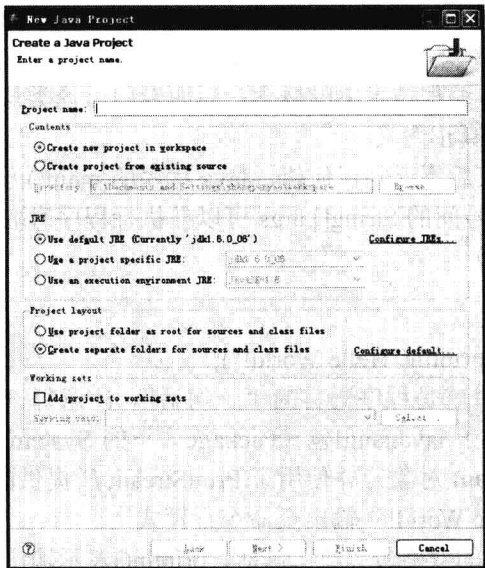


图 1-4 建立 Java Project

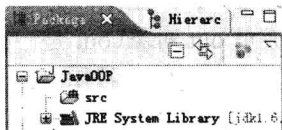


图 1-5 建立好的 JavaOOP 工程

步骤 3 新建 HelloWorld 类

在 src 文件夹下建立的 Java 程序，也就是类（Class），选择“File→New→Class”，在向导对话框中输入类名，此处为“HelloWorld”，并且在“public static void main(String[] args)”选项前打勾，完成操作，如图 1-6 所示。

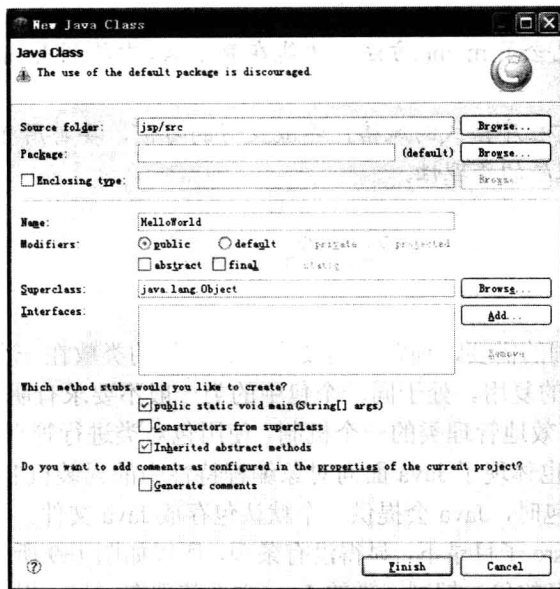


图 1-6 新建类 HelloWorld

步骤 4 编写程序

完成步骤 3 之后，Eclipse 界面将会出现一个编辑区，并且自动生成了 Java Application 的代码框架，只需要在 main()方法体中添加一些 Java 语句，比如若想输出 Hello World，可以这么写：“System.out.println(“Hello World!”);”，如图 1-7 所示。

```
public class HelloWorld {

    /**
     * @param args
     */
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Hello World!");
    }

}
```

图 1-7 HelloWorld 类程序段

步骤 5 程序运行

选择“Run→Run As→Java Application”，或者可以直接用鼠标右键进行相应的选择，这时可以在整个 Eclipse 工作窗口的下方“Console”（控制台）标签内看到运行结果，如图 1-8 所示。

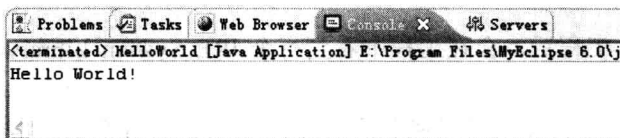


图 1-8 程序运行结果

经验谈

- 在编写程序的过程中，Eclipse 会随时查错，并给出相应的提示。当发现自己的程序出现红色波浪线提示的错误时，把鼠标放在错误处会有相应提示信息，而该错误行的左边也会有个红叉，单击这个红叉，Eclipse 会提供几种更改错误的方式以供参考。
- 只有在类中包含了 `main()` 方法，才能在 Run As 中找到 Java Application 进行程序的运行。
- 写程序要注意程序格式的风格，养成良好的习惯，要善用 Tab 键对程序行进行缩进，以增加程序的条理性。

1.4 包

基于 Java 语言编程的需要，通常把需要在一起工作的类放在一个包（Package）里，以便于实现不同程序间类的复用。处于同一个包中的类一般不要求有明确的相互关系。

包是 Java 语言中有效地管理类的一个机制。使用包对类进行管理，还可以使得程序功能清楚、结构分明，同时也体现了 Java 面向对象编程特性中的封装机制。

当没有自己创建的包时，Java 会提供一个默认包存放 Java 文件。但是所有的文件都放在 `src` 子目录下，显得没有条理，所以如图 1-9 所示，用户可以建立了很多的包，把同一类的 Java 文件整理在一起，以后开发大型程序就会省掉很多麻烦。

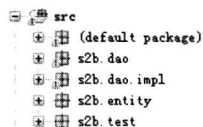


图 1-9 文件结构

1.4.1 包的创建

1. 在 Eclipse 中创建包

在需要存放包的目录下选择“New→Package”即可打开创建包的向导，在 Name 这一栏输入包名。

注意：包名可以是一个合法的标识符，一般全是小写，如 `entity`；也可以使用若干个标识符，用“.”分隔，如：`sun.com`，这种创建方法就相当于在程序目录中建立了两级文件夹，`sun` 文件夹包含了 `com` 文件夹，就是路径 `sun/com`。比如编写第 1 章的程序，建立的包名为 `czmec.java.ch01`。

2. 使用语句创建包

使用 `package` 语句创建包，格式为：“`package 包名;`”。它必须是 Java 文件的第一条语句，而且前面不能有注释和空格，其作用是指明该源文件定义的类都属于这个包。

1.4.2 包的引用

当建立一个 Java 工程后，会在工程中看到有个 JRE System Library，这个类库提供了很多可供用户引用的类，根据功能不同存放在不同的包中，常用的包有以下几个：

(1) `java.lang` 这个是系统的基础类库，比如 `String` 类等都在这里面的。这个包是唯一一个可以不用 `import` 语句就可以使用的包。

- (2) `java.io` 这里面是所有输入输出有关的类, 比如文件操作的类等。
 - (3) `java.net` 这里面是与网络有关的类, 比如 `URL`, `URLConnection` 等。
 - (4) `java.util` 这个是系统辅助类库, 特别是集合类 `Collection`, `List`, `Map` 等。
 - (5) `java.sql` 这个是数据库操作的类库, 包括 `Connection`, `Statement`, `ResultSet` 类等。
 - (6) `javax.servlet` 这个是 `JSP`, `Servlet` 等使用到的类。
- 不管是系统类库还是用户自定义的包, 都要用 `import` 语句来引用。

1.4.3 输入

1. 使用 `Scanner` 类取得输入

学习程序设计的一个好办法是使程序有良好的互动性, 可以对程序输入一些数据, 以便看到程序的执行结果。在命令行模式下要输入数据至程序中时, 可以使用标准输入串对象 `System.in`。在 Java SE6 中, 可以使用 `java.util.Scanner` 类取得用户的输入。`java.util` 指的是包 (Package) 层级, `java.util.Scanner` 表示 `Scanner` 这个类位于 `java.util` 这样的层级之下, 类似于文件管理的目录。

下面直接通过实例来看看如何使用 `Scanner` 类取得用户的输入字符串。

例 1-2

```

1 package czmec.java.ch01;
2 import java.util.Scanner;
3 public class ScannerDemo {
4     public static void main(String[] args) {
5         Scanner input=new Scanner(System.in);
6         System.out.println("请输入您的名字: ");
7         System.out.println("哈罗: "+input.next());
8     }
9 }

```

`java.util` 包是 Java SE 的标准包, 使用 `import` 语句是在告诉编译器, 将使用 `java.util` 下的 `Scanner` 类。

关键字 `new` 表示要新增一个 `Scanner` 对象, 在新增一个 `Scanner` 对象时需要一个 `System.in` 对象, 因为实际上还是 `System.in` 在取得用户的输入。可以将 `Scanner` 看做是 `System.in` 对象的支持者, `System.in` 取得用户输入的内容后, 交给 `Scanner` 作一些处理。

简单来说, 就是可以使用 `Scanner` 这个工具, 用它的 `next()` 功能, 来取得用户输入的字符串, 那么如何取得数字呢? 可以使用 `Scanner` 工具的 `nextInt()` 功能。

现场编程

- 从键盘输入名字和年龄, 并在控制台输出, 使用 `Scanner` 的 `next()` 和 `nextInt()`。

使用同样的方式, 还可以使用 `Scanner` 的 `nextFloat()` 等方法来取得用户的输入, 并转换为正确的数据类型。

2. 使用 `BufferedReader` 类取得输入

`Scanner` 类取得输入的依据是空格, 按下空格键、Tab 键或是 Enter 键, `Scanner` 就会返回

下一个输入，所以有时并不适用。当用户输入一个字符串时，中间可能会包括空格，而用户希望取得的是完整的字符串。如果想要取得包括空格的完整输入，比较简单的方法是使用 `java.io.BufferedReader` 类。

必须要先说明的是，关于使用 `BufferedReader` 类来取得输入，理解起来要复杂得多，如果现阶段无法理解，只要暂时如法炮制就可以了。

`BufferedReader` 类是 `java.io` 包中所提供的一个类，所以使用这个类时必须使用 `import` 语句告诉编译器这个类位于 `java.io` 包下。

使用 `BufferedReader` 对象的 `readLine()` 方法时必须处理 `java.io.IOException` 异常。异常处理机制是 Java 语言为程序设计人员捕捉程序中可能发生的错误所提供的机制，现阶段处理 `IOException` 的方法是在 `main()` 函数中加上 `throws IOException`，在后面介绍异常处理时，大家就理解为何要这么做了。

例 1-3

```
1 package czmcc.java.ch01;
2 import java.io.*;
3 public class BufferedReaderTest {
4     public static void main(String[] args) throws IOException{
5         // TODO Auto-generated method stub
6         BufferedReader reader=new BufferedReader(new InputStreamReader(System.in));
7         System.out.print("请输入一系列文字，可包括空白：");
8         String text=reader.readLine();
9         System.out.println("您输入的文字为："+text);
10    }
11 }
```

“`BufferedReader reader`”表示定义一个类型为 `BufferedReader` 的引用名称，而“`new BufferedReader()`”表示以 `BufferedReader` 类构建一个对象；“`new InputStreamReader(System.in)`”表示接受一个 `System.in` 对象来构建一个 `InputStreamReader` 对象。

简单来说：就是创建一个 `BufferedReader` 工具，这个工具中还要加上一个 `InputStreamReader` 工具，而 `InputStreamReader` 工具的核心是 `System.in` 工具。这三个工具组合在一起，就可以进行文字输入的读取。

由于这次所用到的 `BufferedReader`、`InputStreamReader` 与 `IOException` 等类，都位于 `java.io` 包下，所以在程序的开头使用 `import` 与 `*` 号，告诉编译器到 `java.io` 包下找这些类。

`readLine()` 方法会返回用户在按下 `Enter` 键之前的所有字符输入，不包括最后按下的 `Enter` 字符。程序的执行结果如图 1-10 所示。

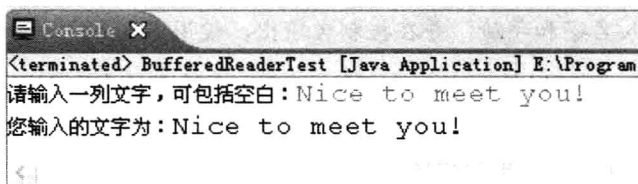


图 1-10 `BufferedReaderTest.java` 运行结果