高等职业技术教育 IT 类双证教材
Borland 软件认证课程推荐教材

Java 语言与 JBuilder 应用基础教程

（Java 初级程序员认证）

ATA 教育公司　总策划
张　健　编著

科 学 出 版 社
北　京

内 容 简 介

本书是高等职业技术教育中 IT 类专业学生获取学历证书和国际著名软件厂商 Borland 软件认证证书的学习用教材。本教材依据课程教学大纲编写而成。

本书全面讲述了 Java 语言的基础知识以及使用 Borland JBuilder 集成开发环境开发 Java 程序的方法。Java 语言部分包括 Java 语言概述，Java 的基础语法，Java 的面向对象特性，接口与包，无用对象回收，数组与字符串类，异常捕获，多线程，输入/输出流库，Java 小程序，用 Java Swing 编写图形界面程序，图形界面程序的事件驱动以及网络编程初步。JBuilder 使用部分包括如何用 JBuilder 的集成开发工具包生成图形界面的 Java 程序，如何使用 JBuilder 工程，如何用 UML 浏览类与包，如何用 JBuilder 制作、调试和发布 Java 程序。本书覆盖范围广泛、重点突出、结构清晰，并通过对大量简单而有针对性的应用例题进行讲解，实现了计算机语言基础知识与具体应用的充分结合。

本书可作为各大专院校、高等职业技术院校计算机软件开发专业课程和相关双证教学课程的教材，也可作为计算机语言的基础教材，并可供从事软件开发行业的技术人员学习参考。

图书在版编目（CIP）数据

Java 语言与 JBuilder 应用基础教程/张健编著. —北京：科学出版社，2005
（高等职业技术教育 IT 类双证教材.Borland 软件认证）
ISBN 7-03-015049-X

I. J… II. 张… III. Java 语言-程序设计-高等学校：技术学校-教材 IV. TP312

中国版本图书馆 CIP 数据核字（2005）第 012237 号

责任编辑：杨 凯 刘晓融 / 责任制作：魏 谨
责任印制：刘士平 / 封面设计：李 力

科学出版社出版
北京东黄城根北街 16 号
邮政编码：100717
http://www.sciencep.com

北京东方科龙图文有限公司制作
http://www.okbook.com.cn

源海印刷有限责任公司印刷

科学出版社发行 各地新华书店经销
*
2005 年 3 月第 一 版 开本：B5(720×1000)
2005 年 3 月第一次印刷 印张：24 1/4
印数：1～3 500 字数：465 000

定价：46.00 元

（如有印装质量问题，我社负责调换〈新欣〉）

高等职业技术教育IT类双证教材
Borland 软件认证课程推荐教材

编审委员会成员名单

前　言

Java 是由美国 Sun 公司开发的一种编程语言与平台，它是第一种具有硬件、操作系统无关性的程序语言。也就是说，用 Java 语言编写的程序可以在不同硬件、不同操作系统下运行，并且不需要重新编译。它是一种“一次编译，到处使用”的语言。使用 Java 语言编程时，不再需要考虑不同硬件、软件平台下的不同特性。Java 既是一种编程语言又是一种平台，在这里“平台”的含义是一套程序运行的软硬件环境。例如 WindowsXP,Linux,Unix,Solaris,Mac OS 等都是常用的平台，这些平台包括了硬件和操作系统。与它们不同，Java 平台是纯软件平台，它运行在上面这些平台之上，为 Java 程序提供一套统一的运行环境。

本书讲授了 Java 语言的基本语法，以及常用的 Java 应用程序界面，特别是用户图形界面的开发，对应的集成开发环境为 Borland 公司的 JBuilder 9。本书是进行 Java 学习的入门级教材，并不要求读者预先掌握一门编程语言。

本书包括以下几方面的内容:

第 1 章　本章是 Java 语言的入门章节，目的是让没有 Java 语言基础的初学者对 Java 语言和 JBuilder 集成开发系统有初步的了解，并能按照步骤指引创建 Java 程序。

第 2 章　本章介绍 Java 的基本语法，主要包括 Java 中对象和类的概念；简单变量和引用变量，简单变量的变量类型，变量的初始化、作用域，以及最终变量的概念；算术运算符、关系与条件运算符、移位与按位运算符、赋值运算符以及其他运算符的具体功能；分支控制语句、循环控制语句以及中断控制语句的功有。

第 3 章　本章讲述了在 Java 中如何实现面向对象的特性，包括 Java 中类的定义，类的成员变量与成员函数的概念以及定义，成员函数的实现；如何利用构造函数为对象的成员变量赋初值；Java 中如何实现类的封装性，类成员的 4 级权限控制的作用；类的继承性概念，子类所能继承到的成员；成员的覆盖，成员函数的重载，带有子类的父类的多态性，以及三者的区别；抽象类与抽象函数的概念及应用；类的静态变量与静态函数的概念及应用。

第 4 章　本章讲述了接口与包的概念与使用。接口是一种抽象的形式，它只有函数定义，没有具体实现，也没有成员变量，接口的功能通过类来具体实现。包是 Java 语言中组织类的一种形式。本章还介绍了如何利用 JBuilder 提供的强大的图形化界面，让编程人员用图形的方式浏览 Java 程序中类与接口之间的继承、实现以及相互引用关系，以及使用 Javadoc 自动生成文档的过程。

第 5 章　本章主要介绍无用对象回收机制及其对编程的影响，深入剖析了变量的有效期与对象的生存期之间的差别，介绍了 Java 的无用对象回收机制，指出 Java 语言特有的“变量与对象分离”的编程机制。

第 6 章　本章介绍了几种常用的 Java 系统类的使用，包括字符串类，数值类以及数字与字符串之间的转换，数组类，集合类，以及 Object 类的功能。

第 7 章　本章主要介绍 Java 中异常处理的内容，包括异常处理的基本概念，使用 try...catch 块捕获各种异常的方法，从函数中抛出异常的方法，自定义异常，运行期异常的概念。

第 8 章　本章介绍了线程，多线程程序的概念，多线程程序的基本设计方法，线程的创建、启动、暂停、停止的方法，线程的优先级，线程间同步的基本知识。

第 9 章　本章介绍 Java 提供的文件输入/输出方法以及文件操作功能，包括文本文件的读出和写入，二进制文件的读出和写入方法，用 File 对象进行各种文件、文件夹操作，临时文件的产生与自动删除。

第 10 章　本章介绍使用 Jbuilder 制作、调试、发布程序的方法，包括用 Jbuilder 编译 Java 程序的方法；用 Jbuilder 调试 Java 程序的方法，包括设置断点，单步运行，实时监控变量的值等功能；制作 Java 压缩包的方法；用 Jbuilder 制作 Java 的发布程序，使得 Java 程序在 Windows，Linux，Solaris，Mac OS 等操作系统下都能够简单地通过一个可执行程序来运行。

第 11 章　本章介绍了 Java 小程序的基本概念及使用方法，编写 Java 小程序的基本步骤，小程序的生命周期及调用小程序的四个函数，如何在小程序中显示图片等资源，如何从网页中向小程序传输参数，以及小程序的安全限制等。

第 12 章　本章介绍了用 Swing 编写 Java 图形界面程序的入门知识，包括 Swing 的起源与概述，用 Swing 创建图形界面的基本步骤，图形界面的层次概念，布局管理器的概念，事件处理的基本原理，以及组件与模型的分离原理。

第 13 章　本章分类介绍各种常用的 Swing 组件，包括顶级容器组件，重点介绍了框架与对话框，特别是各种消息框；通用容器组件，重点介绍面板和工具栏，对各种特殊面板，如滚动条面板、分割条面板、标签页面板也有介绍；专用容器组件，简单介绍了内部框架、分层窗格以及根窗格的内容；基本控件，重点介绍了按钮、单选框、复选框、下拉框、列表框、菜单、文本框等常用的控件；不可编辑组件，重点介绍了文字标签，简单介绍了进度条以及工具提示的使用；交互的格式组件，介绍了调色板以及文件选择框的使用；文本组件，重点介绍文本框、文本区域、密码框的使用。

第 14 章　本章介绍了 Swing 图形界面的几个重要特征：布局、事件驱动，以及一些常用的其他特征。详细介绍了 7 种常用的布局管理器的作用、程序代码；编写事件驱动程序的要点，以及常用的事件监听器；Swing 常用的其他特征，如使用 HTML 控制文体格式、组件边框、工具提示等。

第 15 章　本章介绍了利用 Java 语言进行服务器/客户端网络程序设计的基本知识。介绍了基于 Socket 的网络程序设计步骤，之后综合前面各章内容，介绍了具有一定规模和实用性的 Java 网络应用程序的编写方法。

本书的编写过程得到了 Borland 公司的授权和大力支持，尤克滨、李晓辉、刘兴国等技术专家就本书的技术审核提供了热情帮助，在此一并表示感谢。

本书提供了教师用 PPT 教学软件和多媒体演示光盘供教师和学生使用。教学 PPT 和多媒体光盘内容可从 ATA 教育公司网站下载获得：http://www.atalearning.com/download.asp。

藉此特别说明，本书是科学出版社出版的高等职业技术双证教育 IT 类专业 Borland 软件认证课程教材系列之一，针对 Borland 软件认证《Java 初级程序员认证》证书的考试。本系列还包括针对 Borland 软件认证《J2ME 应用开发程序员认证》《Java Web 应用开发程序员认证》《Delphi 初级程序员认证》《Delphi 程序员认证》《Delphi 高级程序员认证》《Delphi 快速网络开发工程师认证》等证书考试的课程教材。读者可以通过 http://www.sciencep.com/了解其他教材的出版情况，并通过 http://www.atalearning. com/网站获得相关考试的信息。

目　录

第 1 章
Java 与 JBuilder 概述

本章教学目标

- 了解创建 Java 程序的方法，学会一步步创建基础 Java 程序。
- 了解用 JBuilder 创建 Java 程序的方法，学会一步步创建基础的 Java 图形界面程序。

重点与难点

本章是学习 Java 与 JBuilder 的入门章节，目的是让读者对 Java 有初步的了解，并且在还不了解 Java 语言的情况下，就能够按照书中的步骤一步步地创建并运行 Java 程序。这部分的重点在于手工键入整个源代码，而不是直接编译源代码，这样可使学员对输入代码时容易出现的错误有一个直观印象，有利于以后的编程。

JBuilder 是强大的集成化 Java 开发平台，利用它的向导功能，可以让程序员在不用写一行程序的情况下就产生非常漂亮的用户界面。本章的后半部分将利用 JBuilder 的向导创建一个图形界面程序，同时介绍 JBuilder 各部分向导的功能。

推荐课时安排

序号	章节	辅助材料	推荐学时数
1	1.1～1.2	教学 PPT 第 1 章	2
2	实验 1	多媒体演示光盘	2
3	1.3～1.4	教学 PPT 第 1 章	2
4	实验 2	多媒体演示光盘	2

注：教学 PPT 和多媒体演示光盘可从 http://www. atalearning.com/download.asp 上获得。

1.1 Java 历史与概述

Java 是由美国 Sun 公司开发的一种编程语言，它是第一种具有硬件、操作系统无关性的程序语言。也就是说，用 Java 语言编写的程序可以在不同硬件、不同操作系统下运行，并且不需要重新编译。它是一种“一次编译，到处使用”的语言。使用 Java 语言编程时，不再需要考虑不同硬件、软件平台下的不同特性。

最早的 Java 语言开发计划是从 1991 年开始的。1991 年 4 月，Sun 的绿色计

划(Green Project)开始着手于发展消费性电子产品(Consumer Electronics)，所使用的语言是 C、C++及 Oak（为 Java 语言的前身）。随后因语言本身和市场的问题，消费性电子产品的发展已无法达到当初预期的目标，再加上网络的兴起，绿色计划因此而改变了发展方向，这时已是 1994 年了。在绿色计划中，最早考虑采用 C++语言，但从一开始的编译问题到最后的一连串其他问题都迫使放弃 C++语言，转而开发一种新的语言，于是有了 Java 语言的诞生。Sun 公司的目标是要 Java 成为一个简单的(Simple)、面向对象的(Object Oriented)、分布式的(Distributed)、解释的(Interpreted)、鲁棒的(Robust)、安全的(Secure)、结构中立的(Architecture Neutral)、可移植的(Portable)、高效能的(High Performance)、多线程的(Multithreaded)、动态的(Dynamic)程序语言。

- 简单的（Simple）：容易编写程序，不需要长时间的训练，即能满足实际需求。程序小型化也是简单的一种特性，这使得程序能够在小型机器，甚至家电、机顶盒、手机上执行。基本的 Java 解释器只有 25KB，如果再加上程序运行所需的基本程序库，也只有 215KB。这在动辄数十兆、数百兆的 Windows 程序中，的确算是小巧玲珑了。
- 面向对象的（Object-Oriented）：Java 是一种纯粹的面向对象语言，Java 语言中没有全局变量和全局函数，只有对象和对象的成员变量、成员函数。
- 分布式的（Distributed）：Java 有一个很全面的程序库，且很容易地与 HTTP 和 FTP 等 TCP/IP 通讯协定相配合，方便地实现 B/S 和 C/S 以及点对点网络程序结构。
- 鲁棒的（Robust）：由 Java 语言编写出的程序能在多种情况下执行而具有相当高的稳定性。Java 具有完善的变量类型检查、变量初始化检查、数组下标越界检查、无用变量回收机制，因此能够最大限度地提高程序的鲁棒性。
- 安全的（Secure）：Java 是被设计用于网络及分布式的环境中，安全性自然是一个很重要的考虑因素。Java 拥有多层的互锁(Interlocking)保护措施，能有效地防止病毒的侵入和破坏行为的发生。
- 结构中立的（Architecture Neutral）：一般而言，网络是由很多不同机型的机器所组合而成的，这些机器的硬件和操作系统体系结构均有所不同；因此，如何使一个应用程序可以在每一种机器上执行，是一个难题。Java 的编译器产生一种结构中立的目标文件格式(Object File Format)；这使得编译码得以在很多种处理器中执行，而不用考虑不同机器的差异。
- 可移植的（Portable）：Java 的简单数据类型的大小和规格是指定的，例如"float"一直是表示一个 32 位符合 IEEE 754 规范的浮点数，因绝大多数的 CPU 都具有此共同特征。程序库属于系统的一部分，它定义了一些可移植的程序接口，Java 本身具备有很好的可移植性。
- 解释的（Interpreted）：Java 解释器能直接在任何机器上执行 Java 二进制码

(Bytecodes)，这样就省去了在不同机器上进行编译、连接的时间。这对于缩短程序的开发过程，有极大的帮助。

- 高效能的（High Performance）：Java 二进制码能迅速地被转换成机器码(Machine Code)。随着 Java 解释器的改进，Java 二进制码的执行效率正在逐渐逼近其他编译语言的执行效率。
- 多线程的（Multithreaded）：Java 语言具有多线程的功能，这对于交互式程序以及实时响应程序是很有帮助的。
- 动态的（Dynamic）：Java 比 C 或 C++语言更具有动态性，更能适应时刻变化的环境。Java 不会因程序库的更新，而必须重新编译程序。

大部分编程语言都是编译型或解释型的语言，而 Java 语言有所不同，它既是编译语言又是解释语言。编译部分体现在它将 Java 源程序编译为一种中间语言，称为 Java 二进制码，这种代码是与平台无关的。解释部分体现在 Java 平台逐句分析解释并运行 Java 二进制码。因此，Java 程序只需要编译一次，就可以在任何能运行 Java 平台的机器上运行，而不需要考虑计算机的具体硬件特征和操作系统。例如：你在 Linux 下编写的 Java 程序，可以直接拿到运行 Windows 操作系统的 PC 机、Solaris 工作站或 iMac 苹果机上运行。图 1-1 显示了这种“一次编译，到处运行”的工作过程。

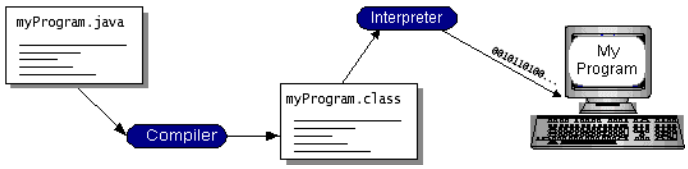

图 1-1

Java 既是一种编程语言又是一种平台，在这里“平台”的含义是一套程序运行的软硬件环境。例如：WindowsXP、Linux、Unix、Solaris、MacOS 等都是常用的平台，这些平台包括了硬件和操作系统。与它们不同，Java 平台是纯软件平台，它运行在上面这些平台之上，为 Java 程序提供一套统一的运行环境。

Java 平台由两个部分组成：Java 虚拟机(JVM)和 Java 应用程序界面(Java API)。Java 虚拟机的作用是解释并运行 Java 二进制码。Java API 是建立在 Java 虚拟机之上的一个包含很多现成的软件组件的软件包，其中包括用户图形界面(GUI)组件。Java API 由许多软件包组成，每个软件包中包含若干类和接口。图 1-2 给出了 Java 平台的组成以及它的定位。

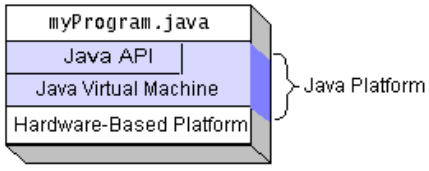

图 1-2

图 1-3 给出了 Java 软件包所包含的主要内容，Java 平台所运行的操作系统，以及 Java 开发系统与运行系统的结构。

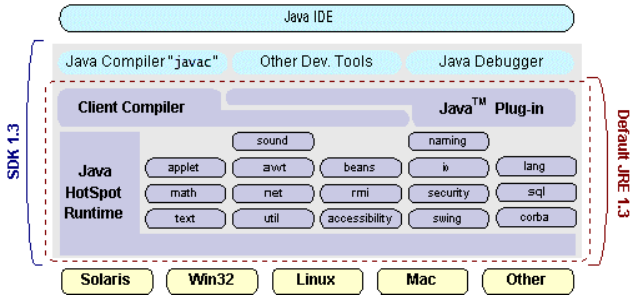

图 1-3

1.2 创建第一个 Java 程序

任务 1.1 编写一个 Java 程序，在屏幕上显示"天天好心情!"。

1.2.1 用记事本编写源程序

在 Windows 系统下，点击屏幕左下角的“开始”按钮，选“程序”→“附件”→“记事本”来打开记事本。在记事本中，敲入如下代码：

```
/*
 * 文件名: FirstApp.java
 * 功　能: 显示"天天好心情!"
 * 编写: 张三
 * 编写时间: 2004.06.03
 * 修改: 李四
 * 修改时间: 2004.08.15
 */
public class FirstApp {
    public static void main(String[] args) {
        // 显示"天天好心情!"
        System.out.println("天天好心情!");
    }
}
```

在录入源程序的时候，有两点需要特别注意。

首先是英文字母的大小写问题。Java 语言区分大小写，在 Java 语言中，

“FirstApp”和“firstapp”是两个不同的名字，“System.out.println”也不能写成“system.out.println”或“System.Out.Println”，否则编译运行时就会出错。至于具体哪些名称、变量、运算符应该大写哪些应该小写，我们会在以后的部分介绍，现在只需严格按照书中的大小写录入即可。

第二个容易出现的问题是中文标点符号问题。在 Java 源程序中，很多标点符号都有特殊的含义，例如：“.”表示对象的一个成员，“;”表示一行语句的结束，双引号“""”表示引号中的内容为字符串等。Java 编译器只认这些英文符号，不认相应的中文标点符号，如果敲入中文符号的话就会造成错误。特别是中文的分号“；”和“;”非常相似，极易搞混；录入中文字符串时，也往往会因为没有及时切换到英文状态，而出现右引号是中文的问题，例如"天天好心情！”；为了避免此类错误，建议编写程序时，除了必须要输入中文的地方以外，尽可能地在英文状态下写。在录入中文以后，也应及时地按“Ctrl”键和空格键关闭中文输入法。

1.2.2 保存源程序

在记事本顶部的菜单中选“文件”→“另存为”，在“另存为”对话框里，选定一个保存源文件的子目录。例如：C 盘下的“Java”目录，在“文件名”一栏敲入“FirstApp.java”，“保存类型”一栏选“所有文件”，如图 1-4 所示，最后单击“保存”按钮。

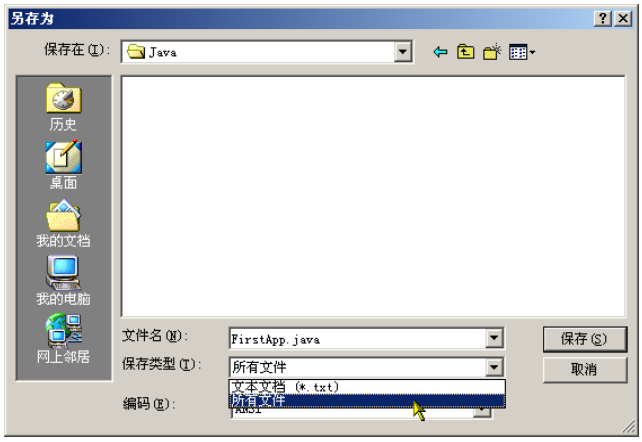

图 1-4

1.2.3 编译源程序

在屏幕左下角的开始菜单中，选择“程序”→“MS-DOS 方式”（Windows 98），或“程序”→“附件”→“命令提示符”（Windows 2000），或“所有程序”→“附件”→“命令提示符”（Windows XP），进入命令行方式，如图 1-5 所示。

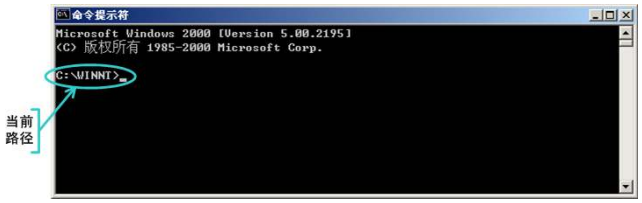

图 1-5

在命令行方式下，有一个闪烁的光标，表示在这里输入命令。光标前的C:\WINNT为当前路径。为编译源文件，需要将路径转换到你的源文件所在路径下，例如“C:\Java”。敲入指令：

```
cd \Java
```

你就会进入“C:\Java”目录下：C:\Java>。在这里敲入指令：

```
dir
```

可以列出该目录下的所有文件，如图 1-6 所示，其中“FirstApp.java”就是你刚才保存的 Java 源文件。

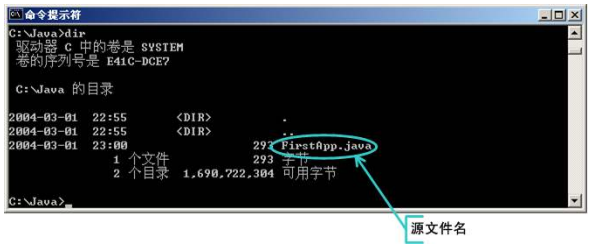

图 1-6

敲入命令（记住最后要敲回车键）：

```
javac FirstApp.java
```

几秒钟后，如果提示符C:\Java>重新出现，没有提示任何信息，说明编译成功了。这时再敲命令：

```
dir
```

可以看到当前目录下多了一个文件，如图 1-7 所示。

图 1-7

“FirstApp.class”就是编译产生的字节码文件，它相当于 Windows 操作系统下的“.exe”可执行程序。

1.2.4 执行程序

在当前目录下，敲入命令：

java FirstApp

你可以看到程序运行结果，如图 1-8 所示。

图 1-8

祝贺你，你的第一个程序运行成功了！

1.3 JBuilder 概述

JBuilder 是 Borland 公司开发的用于 Java 程序设计的一套集成化软件开发环境。JBuilder 环境包含了应用程序开发生命周期从开发、调试、测试、构建直到发布的所有阶段。JBuilder 企业版支持 Java 开发的各个方面，从 Swing 到 JavaServer Pages(JSP)、servlets（小服务程序）、Enterprise JavaBeans(EJB)、数据库应用程序、Web 业务、移动应用程序、Struts、XML 等众多应用程序。利用新的、可配置的个性化设置，可以选择所从事开发项目的具体类型，简化开发环境并显示当前项目所需的工具。JBuilder 是 100% 的 Java 程序，对 Windows、Linux 与 Solaris 平台上面的开发提供跨平台支持。

JBuilder 中引入快速生成项目的 Unified Modeling Language(UML，即统一建模语言)模型，使得程序员可以更加方便地检查、分析、沟通设计信息并理解代码结构，项目组也可以更加方便地对大型复杂项目进行组织与管理。对新的或现有的代码提供了代码格式化支持，这有助于在开发小组内部维持一致的标准。项目内使用的书签与 to-do（要做的事情）有助于跟踪工作进展的情况。JBuilder 还可以与 Borland StarTeam、Rational ClearCase、Concurrent Versions System (CVS)以及 Microsoft Visual SourceSafe.的集成，简化了大型分布式开发小组的源代码并行管理。

JBuilder 除了支持基本的 Java 应用程序以外，还支持 J2EE 平台应用，利用增强的双向可视化 EJB 设计工具，可以快速生成 JavaBeans 插件。JBuilder 使用 Struts、servlets、JavaServer Pages 与 JSP 描述语言等众多工具与向导，可以更加容易地开发动态 Web 应用。新的 Struts 设计工具具有拖放能力，将开发基于 Struts

应用的各个方面集成在一起。新的 JSP、HTML、XML 与 JSP 表述语言的句法标记，再加上拖放能力，加快了编码过程，减少了错误。WebApp 分发描述编辑器可以更快地在 Web 服务器上面启动并运行；JBuilder 提供的数据库连接工具可以更加方便地通过 JDBC 建立数据库连接。利用 JBuilder 内众多的数据感知型 dbSwing 与 DataExpress 部件，可以加速并简化数据库应用的构建。使用定义 XML 文件结构与规则、数据装订与计划性操作的工具，JBuilder 可以迅速完成 XML 与数据库文件之间的数据交换；JBuilder 提供了一个灵活、开放的解决方案，可用于开发、测试、构建并分发基于 Java 的移动应用程序。JBuilder 移动开发支持 Java2 Platform、Micro Edition (J2ME)、Mobile Information Device Profile (MIDP)，并可以与领先移动设备制造商的多种仿真程序协同工作。

利用 JBuilder，可以方便地编写、调试、运行 Java 程序，特别是图形界面的程序。JBuilder 提供许多功能强大的向导，能够让你几乎不用写代码就产生漂亮的图形界面程序。下面我们通过一个例子来一步步详细地演示用 JBuilder 创建图形界面程序的过程。

1.4 用 JBuilder 创建第一个图形界面的 Java 程序

任务 1.2 编写一个图形界面的 Java 程序，如图 1-9 所示。程序的中央显示蓝色的“天天好心情!”，当点击上方的“按这里”按钮之后，字体变成洋红色。

图 1-9

1.4.1 创建一个工程

在用 JBuilder 编写一个 Java 程序之前，需要先创建一个工程，然后在这个工程中编写程序。

（1）运行 JBuilder，在 JBuilder 的菜单栏中选“File”（文件）→“New Project”

（新的工程），会出现一个“Project Wizard”，也就是工程向导，它会引导你一步步创建一个工程。

（2）在“Name”（名称）一栏中敲入“HelloWorld”作为工程名。当你敲入工程名以后，在下面的“Directory”（目录）一栏中会自动创建一个同名的子目录。工程中的所有文件，包括源代码、Java 可执行程序都放在这个目录下。

（3）选中“Generate project notes file”（生成工程注释文件）项，这样工程向导会自动产生一个 HTML 文件，里面保存着工程的注释，可以用浏览器浏览这个文件。

（4）向导中的其他选项保持不变，也就是说：“Type”（工程文件类型）一栏选“jpx”，“Template”（模板）一栏选“Default project”（缺省工程）。当全部选好后，这部分应当显示，如图 1-10 所示。

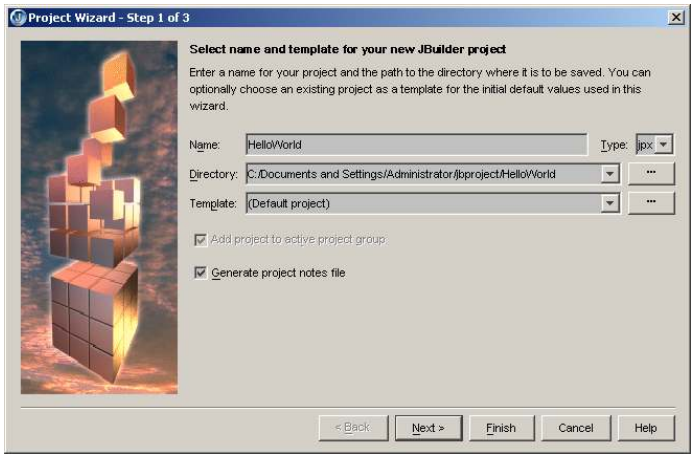

图 1-10

（5）单击“Next”（下一步）进入下一个步骤。

（6）第二步没有要改动的内容，这部分界面应当如图 1-11 所示。

其中“JDK”表示当前 JBuilder 所用的 Java 软件的版本，“Output path”（输出路径）表示编译生成的 Java 可执行程序所在的子目录的位置，“Backup path”（备份路径）表示备份文件所在的子目录的位置，“Working directory”（工作目录）就是这个工程所在的目录。

单击“Next”进入第三步。

（7）在“Title”（标题）部分敲入“Hello World”，在“@author”（作者），“Company”（公司），“Description”（程序描述）部分敲入作者姓名、公司名称和对程序的简单描述，其他内容不用修改。此时向导应该如图 1-12 所示。

（8）单击“Finish”（结束）按钮，一个 JBuilder 工程就创建成功了。

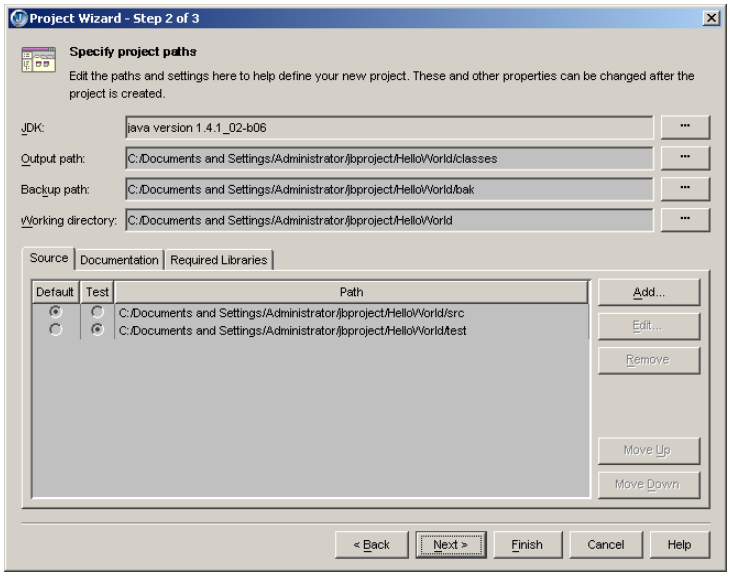

图 1-11

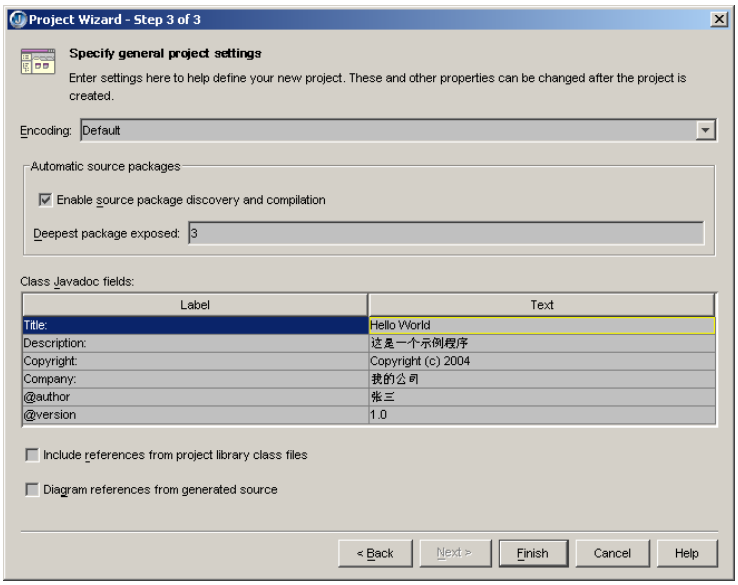

图 1-12

1.4.2 产生源代码

前一步产生的 JBuilder 工程只是提供一个空的架构，这一步要往工程里加入真正的 Java 程序。

（1）在菜单栏中选择“File”→“New”，或者单击工具栏中的按钮，打

开“Object Gallery”（对象陈列室）对话框，在左边的分类列表中选“General”（通用），右边出现这一分类中的对象，单击“Application”（应用程序）图标，再单击“OK”按钮以添加一个应用程序。JBuilder 会自动打开另外一个“Application Wizard”（应用程序向导）来一步步引导你创建应用程序（参见图 1-13）。

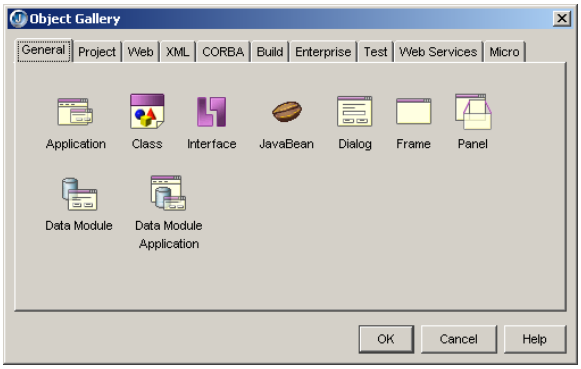

图 1-13

（2）应用程序向导也分三步，第一步是输入应用程序类的基本信息，在“Class name”（类的名称）一栏中填入“HelloWorldClass”，再选中“Generate header comments”（产生文件头注释），以使最后产生的 Java 源文件开头部分包含工程信息。这部分完成后的界面应该如图 1-14 所示。

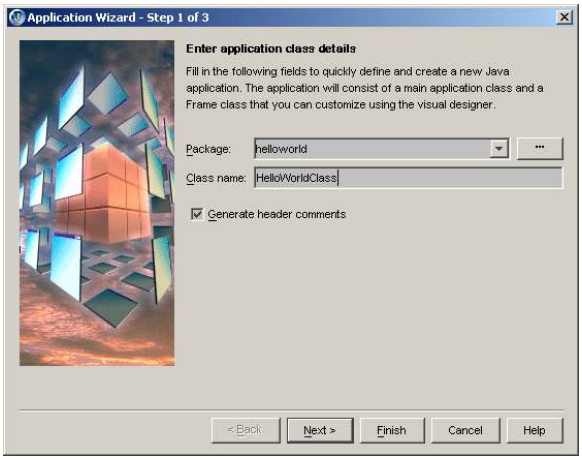

图 1-14

其中“Package”（包）表示程序所在的包的名称。在 JBuilder 中，包的名字与工程名字一致，但是为符合 Java 程序的要求，不至于产生错误，包的名字都是小写。

单击“Next”进入下一步。

（3）这一步产生一个窗体框架类，窗体框架是图形界面的应用程序中最顶级的窗口，程序在窗体框架中显示。在“Class”一栏中敲入“HelloWorldFrame”，在下面的“Title”（标题）一栏中敲入“Hello World”，它将在应用程序最顶端的标题栏中出现。

点中所有的选项：“Generate menu bar”（产生菜单栏），“Generate toolbar”（产生工具栏），“Generate status bar”（产生状态栏），“Generate About dialog”（产生“关于”对话框），“Center frame on screen”（将窗体框架置于屏幕的中央）。

完成后的界面如图 1-15 所示。

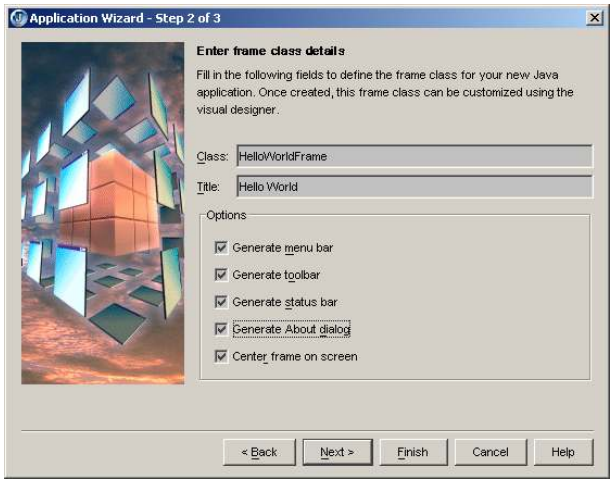

图 1-15

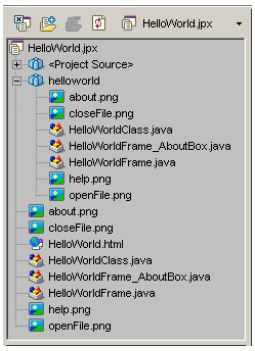

图 1-16

（4）单击“Finish”按钮，跳过第三步，直接完成应用程序向导。向导自动生成三个.java 文件：HelloWorldClass.java ， HelloWorldFrame.java ，HelloWorldFrame_ AboutBox.java，以及工具栏用到的四个图标：about.png, closeFile.png, help.png, openFile.png。它们都放在工程的“helloworld”包里，从 JBuilder 左边的“Project”子窗体中可以看到，如图 1-16 所示。

（5）从菜单中选“File”→“Save All”（保存全部）或单击工具栏中的按钮，以保存全部的工程文件和.java 源文件。

1.4.3 编译和运行程序

在前一步所产生的应用程序中，向导已经将程序运行必需的代码写好了，因

此，虽然我们还没有动手写一行代码，但这个程序已经可以运行了。

从菜单中选“Run”→“Run Project”（运行工程）或者单击工具栏的  按钮，JBuilder 就会自动编译并运行程序，此时的程序运行界面是这样的，如图 1-17 所示。

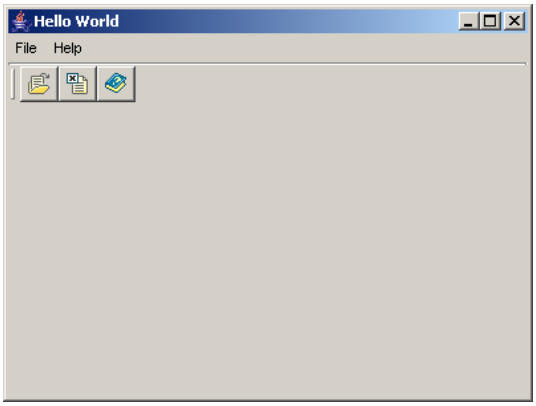

图 1-17

在程序的菜单中选“File”→“Exit”以退出程序。

1.4.4 设计图形用户界面

下面我们在这个空的界面中添加一些内容。前面说过，应用程序向导自动生成三个.java 文件：HelloWorldClass.java，HelloWorldFrame.java，HelloWorldFrame_AboutBox.java，其中 HelloWorldFrame.java 就是主管图形界面的。

（1）如果 HelloWorldFrame.java 文件还没有在 JBuilder 中打开，则双击 JBuilder 左上角的“Project”区，点击“helloworld”包左侧的“+”号展开它，然后双击“HelloWorldFrame.java”以打开该文件。

（2）单击底部的“Design”（设计）标签，切换到设计视图。设计视图的中心部分是界面，其上是组件栏，右侧是组件属性区，左下角是组件树，程序界面中用到的所有组件都以树的形式显示在组件树上，如图 1-18 所示。

（3）放置一个面板：面板就是放在程序界面上的一块平板，图形、文字、按钮等组件都放在面板上。首先单击组件栏中的“Swing Containers”（Swing 容器）标签，再单击 JPanel 组件（将鼠标移到组件图标上，稍停片刻，就可以看到组件名），最后在程序界面的中间点击一下鼠标，这个面板就会自动放在界面上。同时，在左侧的组件树中也可以看到该组件。

（4）设置面板的背景色为白色：在组件属性区，点击“background”（背景色）项，在下拉框中选背景色为“White”（白色），如图 1-19 所示。

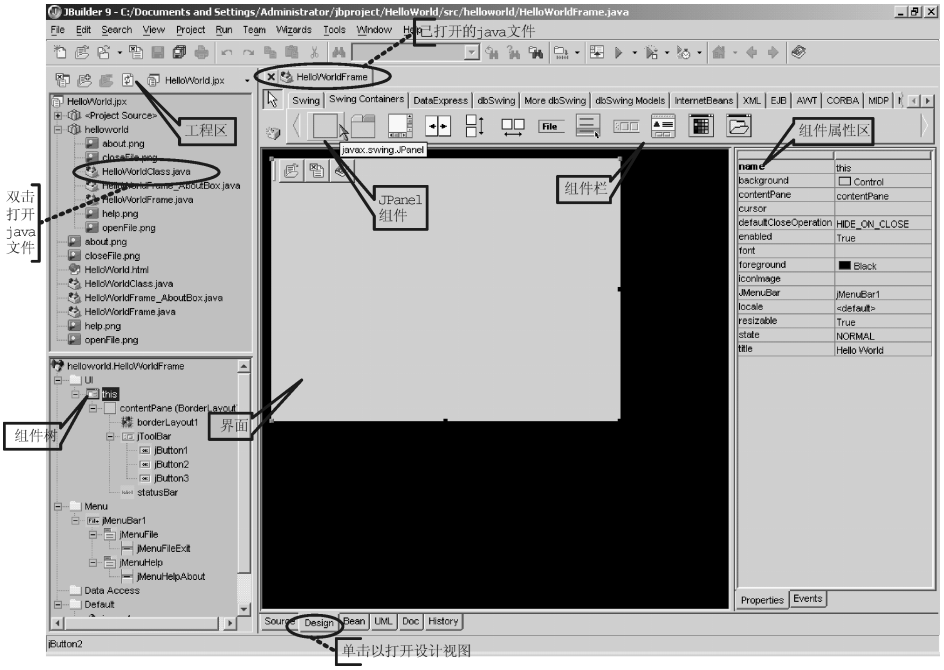

图 1-18

如果想给面板另外设一种颜色，但下拉框里给出的颜色太少，不能满足需要，那么可以点击“背景”项最右边的按钮，弹出调色板，如图 1-20 所示。在调色板里可以分别设定红、绿、蓝三原色的亮度，0 为没有，255 为最大，总共可以有 255×255×255=16 581 375 种颜色。

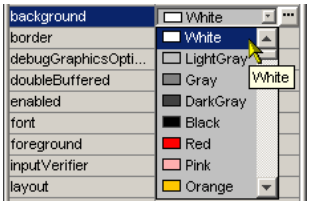

图 1-19

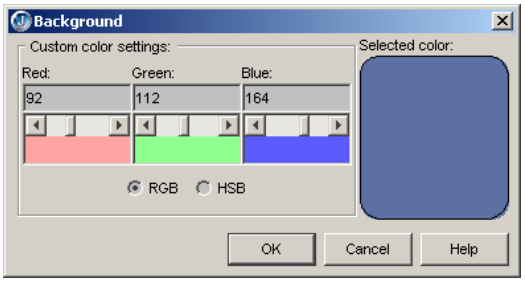

图 1-20

（5）给面板增加灰色的边框：在组件属性区里，单击“border”（边框）项，从下拉框中选择边框类型为“Line”（线条）。单击边框项右边的省略号按钮，出现一个设置边框属性的对话框，点击“Options”（选项）→“Color”（颜色）项，在下拉框中选边框颜色为“Gray”（灰色），最后单击“OK”键关闭对话框（参见图 1-21）。

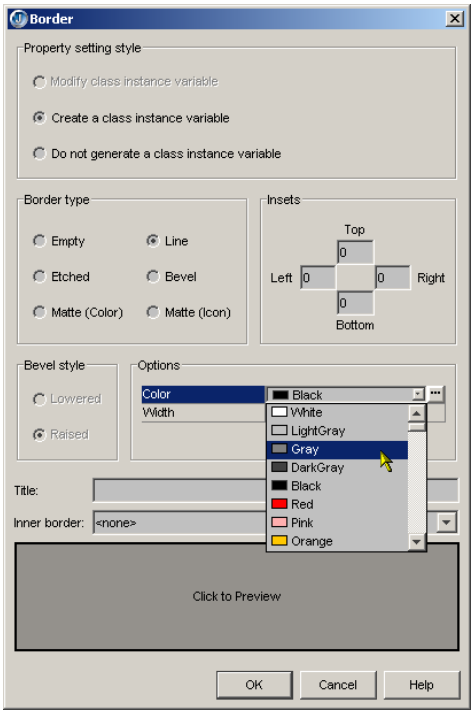

图 1-21

（6）设置属性中的“layout”（布局）项为“null”（空）。在最初设计界面阶段，这样设置很方便，因为此时可以将组件放到任何想放的位置。但带来的问题是：组件是以绝对位置放置的，例如将一个按钮放在(10,20)位置，那么无论窗口放大或缩小，这个按钮都会在这个位置，而不会移动，这样就破坏了整体界面的效果。因此，在正式设计界面时，一定不要设布局为空。我们在后面的章节中还会详细介绍布局问题，以及如何选择合适的布局。

（7）在程序界面上增加一行文字。在组件栏中单击“Swing”页，再单击“JLabel”按钮以选择这个组件，如图 1-22 所示。

图 1-22

下面把这个组件放到 jPanel1 面板上，有三种方法可以放置：

- 鼠标移至左侧的组件树区，在 jPanel1 面板组件上单击，一个 jLabel1 标签组件就会放到这个面板上，自动放到面板的左上角。

- 在程序界面上适当的位置单击鼠标，一个 jLabel1 标签组件就会放到这个面板上，标签的位置就在鼠标单击的位置，但是标签大小还是缺省大小。

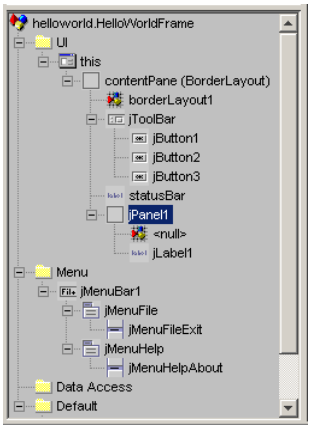

图 1-23

- 在程序界面上按住鼠标左键并拖动，放开左键后一个 jLabel1 标签组件就会出现在面板上，它的位置和大小都由你决定，这是最灵活的一种方式。

jLabel1 标签组件应该加在 jPanel1 面板上，从组件树可以看到这一点。如果拖动时将标签放到了错误的位置，可以在组件树上单击选中错误的标签，然后按“Del”键删除它（参见图 1-23）。

将标签拖到程序的面板中央，双击组件属性区里的“text”（文本）项，敲入“天天好心情！”，回车，此时标签里的文字变成“天天...”，只有开头几个字，暂时先不管它。

单击“font”（字体）属性右侧的按钮，会弹出“字体”对话框，将字体设为“楷体”，字号（Size）设为 28，单击“OK”关闭对话框。再用鼠标把标签拉大到适当大小，并重新放到界面中央。点击“foreground”（前景色）属性，将前景色改为蓝色。设计出的界面如图 1-24 所示。

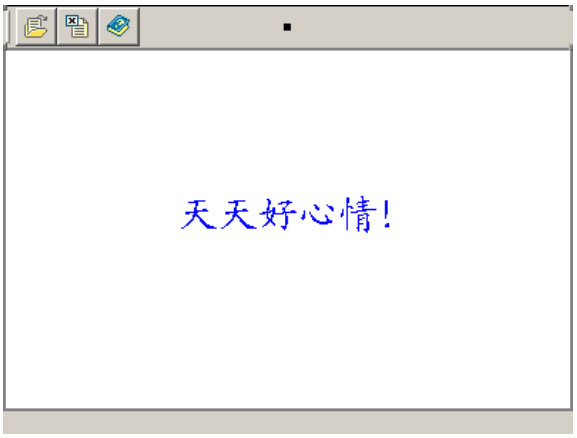

图 1-24

（8）添加一个按钮。在组件栏的“Swing”页中选 JButton 组件，将它拖到文字标签的上方。将 jButton4 组件的“text”属性改为“按这里”，字体改为“宋体”，14 号字，适当拉大按钮以使按钮上的文字显示完全。最后设计出来的界面是这样的，如图 1-25 所示。

图 1-25

（9）为按钮添加事件。按钮的作用就是让用户点击它，然后来实现一项操作。从程序设计的角度来说，“用户点击按钮”就是一个事件，程序员的任务就是为这个事件编写一段程序，指示计算机应该做什么。本节中，我们编写一段程序，用户点击按钮时，文字标签就会由蓝色变成洋红色。

双击按钮，JBuilder 会自动为按钮添加一个“ActionPerformed”（行动已执行）事件，并跳转到编程界面，光标停留在一段空的子程序代码之中：

```
113   void jButton4_actionPerformed(ActionEvent e) {
114
115   }
```

敲入下面一段代码（以加粗字体显示的部分）：

```
void jButton4_actionPerformed(ActionEvent e) {
  jLabel1.setForeground(new Color(255,0,255));
}
```

这段代码的含义是设置 jLabel1（也就是程序界面上显示的那行字）的前景色为洋红色。程序运行的时候，当你单击按钮时，就会激发这个事件，让计算机执行粗体字显示的代码，将文本标签的颜色设为洋红色。

（10）编译并运行程序。在菜单中选择“Project”→“Make Project "HelloWorld.jpx"”（编译工程）以编译 java 程序，再从菜单中选“Run”→“Run Project”运行程序。程序运行后，点击“按这里”按钮，你会看到字体由蓝色变成洋红色了，如图 1-26 所示。

（11）从命令行方式运行程序。上面的程序运行方式需要在 JBuilder 的开发环境下运行，实际中当然不能要求每个用户都安装一个 JBuilder。在 JBuilder 的开发环境以外，程序可以在命令行方式下开始执行，具体执行过程与 1.2 节很相