

Borland/Inprise核心技术丛书

JBuilder培训教程

(美) Borland Software Corporation 著

周 鹏 龚 超 刘 超 等译

北京宝兰-英博思信息技术有限公司推荐用书

本书附盘可从本馆主页 <http://lib.szu.edu.cn/>
上由“馆藏检索”该书详细信息后下载,
也可到视听部复制



机械工业出版社
China Machine Press

JBuilder是美国Borland公司最近推出的遵循Sun公司J2EE标准的可视化集成开发工具,是目前使用最广泛的Java开发工具之一。本书主要基于JBuilder开发环境讲解怎样进行Java设计,利用JBuilder提供的各种机制、组件等加快程序开发进度。本书语言简练、结构合理,每章分专题论述了JBuilder的各个重要方面。并在书后附有大量的复习题及练习,可以帮助读者温习巩固所学知识。相信本书对JBuilder的初学者及高级用户都能提供很好的帮助。

本书中文简体字版由北京宝兰-英博思信息技术有限公司授权机械工业出版社在中国大陆境内独家出版发行。未经出版者书面许可,不得以任何形式复制或抄袭本书的任何部分。

版权所有,侵权必究。

图书在版编目(CIP)数据

JBuilder培训教程/(美)宝兰公司著;周鹏等译.-北京:机械工业出版社,2002.1
(Borland/Inprise核心技术丛书)
书名原文:JBuilder Foundations
ISBN 7-111-09484-0

I. J… II. ①宝… ②周… III. Java语言-程序设计-技术培训-教材 IV.TP312

中国版本图书馆CIP数据核字(2001)第074563号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

责任编辑:刘立卿

北京昌平奔腾印刷厂印刷·新华书店北京发行所发行

2002年1月第1版第1次印刷

787mm×1092mm 1/16·23.5印张

印数:0 001-4 000册

定价:49.00元(附光盘)

凡购本书,如有倒页、脱页、缺页,由本社发行部调换

序 言

Java语言是Sun公司推出的计算机应用程序开发语言，由于它具有与平台无关的特点及完备的面向对象特性，因此越来越得到开发人员的喜爱和认可。

美国Borland公司是世界著名的计算机软件厂商，其推出的开发工具产品如Delphi、C++Builder和JBuilder等享誉全球。JBuilder是Borland公司最近推出的遵循Sun公司J2EE标准的集成开发工具，它是目前使用最广泛的Java开发工具之一。

JBuilder提供了可视化的集成开发环境，支持开发人员构建标准的Java应用系统。其开放的环境、基于组件的开发方式以及丰富的Java开发向导，极大地简化了开发人员的工作。无论是构建标准的Java应用程序、Java Applet、Java Bean、数据库应用，还是JSP、Servlet等Web应用，或者RMI、CORBA、EJB等分布式应用系统，JBuilder都提供了大量的组件、开发向导以及应用模板，使得开发更为快捷和简便。

北京宝兰-英博思信息技术有限公司是Borland/Inprise公司在中国大陆的授权代理机构，负责向中国大陆地区用户提供Borland/Inprise公司全系列产品及其技术服务。本书素材由北京宝兰-英博思信息技术有限公司提供，由机械工业出版社出版，相信它一定会成为您的良师益友。



北京宝兰-英博思信息技术有限公司

www.inprise.com.cn

译者序

近日，Gartner Group通过对400多个IT专业人员的调查结果表明，美国目前大约80%的公司业务运作过程中应用了Java技术，有71%的人指出这些企业面临的最大问题就是缺乏经过专业培训的人。“物以稀为贵”，Java人才的各种待遇正日见攀升。

本书主要基于JBuilder开发环境讲解怎样进行Java设计，利用JBuilder提供的各种机制、组件等加速程序开发进度。本书语言简练、通俗易懂，将Java语言的方方面面以及JBuilder的功能介绍得淋漓尽致。从本书可以看出作者的思路非常清晰，组织结构非常合理：分专题对各个重要方面进行了详细论述，每一章首先给出本章的重点，然后分条介绍，最后在每章的结尾还有小结，这种组织结构非常利于初学者全面掌握Java语言和JBuilder环境。在理论介绍的同时，还提供了许多练习及时帮助你温习巩固所学的知识。

现在，我们很荣幸能够有机会承担本书的翻译工作。我们抱着认真的态度将本书的中文版奉献给您，希望您能够有所收获。这是作者的初衷，也是我们良好的愿望！

本书由周鹏、龚超、刘超组织翻译，万方工作室的全体同仁参加了本书的校正、输入等工作。其他参加本书翻译、录排、校对工作的人员有：龚建、尹建军、刘今朝、强秀丽、任宇飞、李红玲、白红利、金荣学、薛彪、叶哲、邓海燕、邢倩、王育红、李军、刘彬、钱斌、赵锁、姜南、李智、田韞、李林、张巧莉、陈曙晖、邓波、邓涛、李卓林、聂宛析、田敏、龚露娜、马军、马丽、田军、田野、田蕴哲、王小将、李素丽、天海鹏、黄传明等。本书的出版是集体劳动的结晶，在此特别感谢万方工作室的全体工作人员。

由于时间仓促，且译者经验和水平有限，译文难免有不妥之处，恳请读者批评指正！

译者

2001年8月

目 录

序言	
译者序	
第1章 Java及JBuilder简介	1
1.1 Java简介	1
1.1.1 Java虚拟机	1
1.1.2 Java语言	3
1.1.3 Java语言的版本	3
1.2 JBuilder简介	4
1.3 AppBrowser	4
1.4 集成开发环境	6
1.5 类层次浏览器和挖掘功能	12
1.6 帮助系统的使用	15
1.7 小结	19
第2章 工程的使用	20
2.1 工程简介	20
2.2 使用工程向导建立一个新工程	20
2.3 使用应用程序向导	25
2.4 添加文件到工程	28
2.5 设置工程选项	29
2.6 小结	35
第3章 在用户界面设计器中使用组件	37
3.1 基于组件的程序开发	37
3.2 用户界面设计器	38
3.3 在应用程序中添加组件	40
3.3.1 改变组件大小	41
3.3.2 添加多个相同组件	41
3.4 Inspector	42
3.5 组件树	44
3.6 容器	47
3.7 改变组件名称	51
3.8 选择多个组件	52
3.9 创建事件处理程序	53
3.10 小结	56
第4章 布局管理器	58
4.1 概要	58
4.2 容器和布局管理器	58
4.2.1 XY布局管理器	60
4.2.2 边框布局管理器	60
4.2.3 网格布局管理器	62
4.2.4 网格袋布局管理器	64
4.2.5 其他布局管理器	68
4.3 小结	69
第5章 组件样例	70
5.1 JBuilder的组件	70
5.2 小结	74
第6章 菜单设计器	75
6.1 概要	75
6.2 使用菜单设计器	77
6.2.1 在JBuilder中如何使用菜单设计器	77
6.2.2 菜单设计器的工具栏	78
6.2.3 菜单设计器的上下文菜单	78
6.3 设计一个主菜单	78
6.3.1 添加一个JMenuBar组件	78
6.3.2 菜单项	81
6.3.3 分隔条	84
6.3.4 子菜单	84
6.3.5 为JMenuBar设置背景颜色	85
6.3.6 为菜单项设置事件	89
6.4 设计一个上下文菜单	89
6.5 小结	91
第7章 Java语言基础	92
7.1 Java语法	92
7.1.1 标识符	92
7.1.2 常量	93

7.1.3 关键字	95	9.1.2 抽象类	129
7.1.4 语句	96	9.1.3 使用接口	133
7.1.5 代码块	96	9.2 小结	138
7.1.6 注释	96	第10章 使用JBuilder的调试器	139
7.1.7 表达式	97	10.1 概要	139
7.1.8 运算符	98	10.2 调试器环境	140
7.1.9 赋值运算符	100	10.2.1 调试器的使用	140
7.2 Java的数据类型	102	10.2.2 执行点	143
7.2.1 变量	102	10.3 控制程序执行	143
7.2.2 内置数据类型	103	10.3.1 跟踪调试	144
7.2.3 合成数据类型	103	10.3.2 智能调试	145
7.2.4 类型转换	105	10.3.3 单步跟踪	145
7.2.5 作用域规则	106	10.3.4 运行到光标所在位置	145
7.3 流程控制结构	107	10.3.5 运行到方法结束	145
7.3.1 循环	107	10.3.6 程序复位	146
7.3.2 循环控制语句	109	10.4 使用断点	146
7.3.3 条件语句	110	10.4.1 设置断点	146
7.4 小结	112	10.4.2 查看断点	147
第8章 Java面向对象程序设计	113	10.4.3 添加及删除断点	148
8.1 OOP简介	113	10.4.4 启用和禁用断点	148
8.2 类	113	10.4.5 查找断点	149
8.2.1 类的声明及实例化	114	10.4.6 修改断点选项	149
8.2.2 数据成员	114	10.5 检查程序数值	150
8.2.3 类方法	114	10.5.1 监视表达式和变量	150
8.2.4 构造器和终结器	115	10.5.2 求值和修改表达式	152
8.2.5 案例学习: 一个简单的OOP范例	115	10.6 定制调试器	152
8.2.6 类继承	118	10.6.1 显示的控制	152
8.2.7 访问修饰符	120	10.6.2 颜色的改变	152
8.2.8 存取器方法	121	10.7 小结	152
8.3 Java包	124	第11章 事件处理	155
8.3.1 import语句	124	11.1 简介	155
8.3.2 包的声明	125	11.2 标准事件处理	155
8.3.3 一起创建一个新的类和包	126	11.3 侦听器接口	156
8.4 小结	126	11.4 JBuilder中的适配器类	158
第9章 Java高级面向对象程序设计	128	11.5 匿名事件处理	160
9.1 多态性	128	11.6 小结	161
9.1.1 方法重载	128	第12章 异常处理	163

12.1 概述	163	16.1.3 属性管理	209
12.2 异常处理模型	163	16.1.4 事件处理	210
12.2.1 try和catch语句	164	16.1.5 自我测量	211
12.2.2 多catch语句块与单catch语句块	165	16.1.6 持久性	211
12.2.3 异常传播	165	16.1.7 应用程序构建器支持	211
12.2.4 throw语句	165	16.2 用JBuilder创建JavaBean	211
12.2.5 finally语句	165	16.3 添加属性	214
12.3 Throwable类	166	16.4 往组件面板添加JavaBean	216
12.4 Java中的内置异常	166	16.5 小结	219
12.5 实例学习：简单计算器	167	第17章 主从数据库框架	221
12.6 小结	168	17.1 主从关系	221
第13章 JBuilder数据模型	169	17.2 创建主从用户界面	222
13.1 在JBuilder中使用关系数据库简介	169	17.3 小结	228
13.1.1 Java数据库连接API	169	第18章 编程操作QueryDataSet	229
13.1.2 InterClient介绍	170	18.1 使用Java操作数据库	229
13.2 JBuilder中的数据访问组件	173	18.1.1 获取和修改数据	229
13.2.1 配置数据库组件	176	18.1.2 导航	232
13.2.2 设置QueryDataSet组件	181	18.1.3 保存	235
13.2.3 使用数据感知控件	186	18.2 小结	236
13.2.4 添加事件处理代码	188	第19章 数据模块	237
13.3 小结	190	19.1 数据模块介绍	237
第14章 数据库工具	191	19.2 创建数据模块	237
14.1 JDBC管理器	191	19.3 数据划分	242
14.1.1 数据库URL创建	191	19.4 小结	244
14.1.2 查看与编辑数据	192	第20章 列组件	246
14.1.3 使用SQL	193	20.1 列组件	246
14.2 SQL构造器	193	20.1.1 改变列的默认标题	247
14.3 JDBC监视器	198	20.1.2 编辑和显示掩码	248
14.4 小结	199	20.1.3 数据验证	250
第15章 使用数据感知控件	200	20.2 选择列表	251
15.1 开始使用数据感知控件	200	20.3 计算列	256
15.2 高级数据感知控件	203	20.4 小结	259
15.3 小结	206	第21章 参数化查询和数据集视图	260
第16章 JavaBean和BeanExpress概要	208	21.1 参数化查询	260
16.1 简介	208	21.1.1 ParameterRow组件	265
16.1.1 JavaBean的优势	208	21.1.2 数据延迟获取	270
16.1.2 JavaBean组件模型	209	21.1.3 执行不返回结果集的查询	272

21.2 小结	273	26.1 概述	307
第22章 存储过程	274	26.2 Servlet	307
22.1 存储过程定义	274	26.3 会话	310
22.2 返回记录集	274	26.4 Servlet向导和JWS	313
22.3 小结	280	26.5 安全性	313
第23章 高级异常处理技术	282	26.6 小结	313
23.1 创建一个定制异常类	282	第27章 使用Java和HTML创建报表	314
23.2 异常处理用于资源保护	283	27.1 Java的打印功能	314
23.3 DataSetException类	283	27.2 HTML报表	314
23.4 在控件中重载默认DataSetException 处理	285	27.3 报表范例	318
23.5 小结	286	27.4 小结	318
第24章 企业版Java Bean综述	287	第28章 串行化	319
24.1 概述	287	28.1 概述	319
24.2 企业版Java Bean 体系结构	287	28.2 为什么要串行化	319
24.3 企业版Java Bean 组件	288	28.3 在JDK 1.1中实现串行化	319
24.3.1 会话Bean	289	28.4 串行化接口	320
24.3.2 实体Bean	289	28.5 使用输出流	321
24.4 容器	289	28.6 ObjectOutputStream类方法	322
24.5 EJB的开发	290	28.7 使用输入流	322
24.6 EJB客户端	295	28.8 ObjectInputStream类方法	323
24.7 将EJB部署到Borland应用程序服务器	297	28.9 对象流的读和写	323
24.8 小结	298	28.10 小结	323
第25章 CORBA基础	299	第29章 部署应用程序	324
25.1 概述	299	29.1 部署策略	324
25.2 CORBA	299	29.2 命令行Java	324
25.3 CORBA 向导	301	29.3 Java运行环境	325
25.4 注册ORB	301	29.4 部署软件	326
25.5 服务器	303	29.5 JAR文件	326
25.6 客户端	304	29.6 存档文件生成器向导	326
25.7 小结	306	29.7 优化部署	331
第26章 Servlet简介	307	29.8 小结	331
		附录 JBuilder类练习集	332

第1章 Java及JBuilder简介

本章解决以下问题：

- Java编程语言的特点是什么？
- Java虚拟机(JVM)的功能是什么？
- 什么是JBuilder？
- JBuilder的主窗口在哪里？
- 什么是Project(工程)窗格、Content(内容)窗格、Structure(结构)窗格？它们分别在什么位置？
- 怎样打开Designer(设计器)？
- 怎样自定义Source code editor(源代码编辑器)？及怎样改变字型、键盘映射、语法提示等？
- 怎样通过“挖掘”功能来查看一个类的定义？
- 使用JBuilder将得到何种在线帮助？
- 怎样查找在线帮助？
- 在源代码编辑器和Inspector（检查器）中怎样获得帮助信息？

1.1 Java简介

在使用Java这个词的时候，会产生许多概念混淆现象。这也许是因为：从某些方面来说，Java这个词指的是两种不同的事物，即Java虚拟机及Java编程语言。在实际应用中，Java编程语言是用来创建应用程序及小应用程序的。第7~9章会深入地介绍有关Java编程语言方面的内容。而在介绍Java编程语言有关概念之前，我们有必要先介绍一下Java虚拟机。

1.1.1 Java虚拟机

Java虚拟机（以下简称JVM）是一个抽象的概念，它的功能主要是为开发者提供一个与众不同的独立的Java开发平台。JVM可以被看做是一台计算机，而这台计算机具有特殊的机器语言可以运行Java程序。它是一个以堆栈为基础的32位机。机器语言指令被称为字节码（bytecodes）。这些字节码包括数值进栈指令、数值运算指令及数值出栈指令等。JVM还有一个特殊的内存区域，相当于微处理器的寄存器区。这种方法的优点是多数计算机都可以被模拟为JVM并实现其功能。

字节码主要是由用Java语言编写的源程序经过编译而得到的，除此之外，还有其他高级语言及汇编语言的编译器。图1-1描述了高级源代码、编译器及JVM之间的关系。

JVM为用户提供了四个主要功能，包括：类库的装载，内存垃圾收集，安全性，与本地（与平台相关的）代码的接口。

1. 字节码的装载

当某Java程序需要调用其他类或者库中代码时，JVM负责从磁盘或者通过网络加载该程序需

要的字节码。如果其他类需要调用刚才所访问类中某函数时，JVM能够知道需要的代码已经加载了。JVM还可以有选择地卸载一些代码，从而节省内存空间。

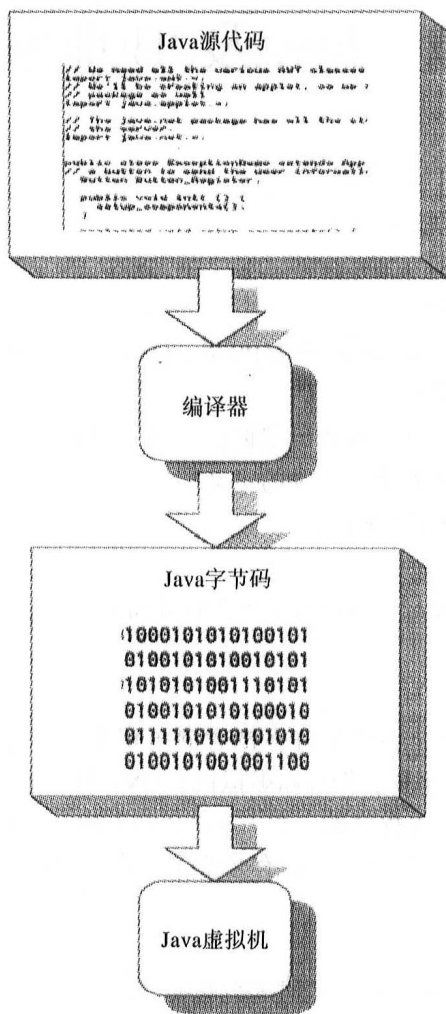


图1-1 Java虚拟机与Java语言

2. 内存垃圾收集

编程中所遇到的最常见的问题就是对内存的管理和分配。程序员经常在使用了内存之后忘记或改变了该块内存的指针，从而不知不觉地生成内存漏洞。

使用Java，问题就会有所改变。JVM会自动地跟踪每一个对象的标识，当发现已经没有任何标识指向某个对象时，JVM就会自动地释放该对象所使用的内存。

3. 安全性

Java最引人入胜的地方也许就是它的内置安全特性。当字节码加载进JVM后，代码将经历原理校验器，这个校验器将检测代码的安全性。尤其要检测以下几点内容：

1) 代码是否访问未初始化的内存。

2) 是否执行了非法的数据类型转换。举例来说, JVM不允许将整型数据用做指针变量。

3) 程序是否存在堆栈溢出(包括上溢出和下溢出)。

该校验器不仅使Java程序提高了安全性, 而且使程序更加健壮, 并大大减小了系统崩溃的可能性。总之, 一个程序如果不满足以上提到的三点检测要求, 那么它要么是违反安全限制的程序, 要么就是存在许多毛病(bug)的程序。

Java的安全性还可以阻止一些不确定的程序访问本地存储器, 如硬盘; 而且可以阻止该程序任意访问网络。本书将在第29章中详细介绍这些限制对用户的重要性。

4. 本地代码访问

许多操作系统拥有一些在Java标准中并未定义的性能。举例来说, Java并不直接支持某些硬件设备, 如串行端口、音频数字转换器及特殊的I/O设备。为解决这一问题, JVM可以调用本地代码, 即这些代码充当适用于本系统的微处理器。有一点开发者应当注意, 应当尽可能少地使用本地代码。因为如果一个应用程序使用了本地代码, 那么该应用程序将失去Java程序的相对于平台的独立性。

1.1.2 Java语言

本书的主要作用是增加读者对Java编程语言的理解。Java是一种类似于Smalltalk的面向对象的程序设计语言。它最初是由Sun公司的负责开发电视机顶盒和嵌入式系统的工作组创建的。随之, Sun公司的许多员工看到了Java在开发适于网上发布的独立于平台的软件方面的巨大潜力。

第一眼看上去, Java程序与C++程序极为相似。这主要是由于两种语言的语法和控制结构很相似(见第7章)。这种相似性使得熟悉C++的程序员在学习Java时相对容易一些。但这里要注意的是, Java在许多方面与C++是完全不同的。总体来说, Java屏弃了C++中许多经常会导致错误的复杂特性, 比如完整的多重继承、预处理器宏命令、运算符重载等。通过创建一种相对简单的语言, Java的创建者们希望提高程序的易懂性和创造性。

以下是Java语言的几个主要特性:

- 它彻底是面向对象的, 这就意味着在类以外没有定义任何数据及函数。
- 所有对象从单一基类(即Object)继承而来, 并且所有对象共享Object类对象的所有功能。
- 它通过接口只支持有限形式的多重继承。关于接口以及面向对象程序设计将在第8和第9两章中详细讨论。
- Java中的简单数据类型, 比如浮点型或整型, 它们的数值范围是确定的, 而不会根据所实现的操作平台而改变。
- 程序员无须显式地释放对象占用的内存, Java语言中的内存垃圾收集功能(由我们前面讨论的JVM实现)可以自动完成这项操作。
- Java语言自带许多标准类库, 它们可以完成许多任务, 比如网络通信、用户图形界面显示, 以及数据结构管理等。
- Java语言不允许对指针进行算术运算, 从而大大提高了程序的安全性。

1.1.3 Java语言的版本

Java是一项崭新的并且正在发展的技术, 随着Sun公司不断为其添加新的特征, 新版本的

Java不断产生。这些新版本的序号是根据Sun公司的Java开发工具包（JDK，Java Development Kit）确定的，其中该工具包是最早的Java开发工具。版本号是以X.Y.Z的形式命名的，这里X表示主版本，Y表示带有小的新特征修订的次版本，Z表示错误修订或程序补丁。因此，如果某人说某Java程序是为JDK 1.2编写的，这就意味着Java环境（包括JVM及类库）应当支持Java 1.0标准版的第三个补充版本。到现在为止，JBuilder能够支持JDK 1.3及更老的版本，将来还可以支持更新的版本。

Sun的JavaSoft子公司通过因特网发布关于Java的最新信息。网址为：<http://java.sun.com/>。

1.2 JBuilder简介

如果你选择了JBuilder作为你的Java程序的开发环境，那么几乎可以说你已经选择了一个功能最为强大的、最容易使用的、程序开发速度最快的Java开发工具。在以往许多软件开发人员在使用快速程序开发工具（RAD）的时候犹豫不决，因为他们担心这样的开发环境及语言会使他们失去想要的程序的灵活性及可控性。JBuilder的出现改变了这些，它将Java语言的强大功能与带有用户图形接口的快速程序开发环境很好地结合在了一起。

JBuilder具有许多优点，本书介绍的主要优点包括：

- 怎样借助于JBuilder的向导（Wizard），使我们创建应用程序、类及工程变得异常容易。
- 怎样使用Designer（设计器）和Two-Way Tools（双向工具），使用户界面设计变得非常简单。
- JBuilder如何充分使用图形用户界面环境，来创建关系数据库的客户端/服务器应用程序。
- 如何轻松地创建类和JavaBeans以辅助当前的应用程序或者小应用程序（applet）的开发过程，以及帮助提高未来项目的性能。

在讨论以上几个问题之前，有必要先介绍一下JBuilder的开发环境。图1-2是JBuilder安装完毕后运行的第一个界面。

可以注意到，当运行JBuilder时，系统将打开一个以Welcome为题的工程。这个工程为用户提供了熟悉JBuilder的很好的方法，开发者可以通过打开welcome.html，并且点击按钮“Begin your tour”来进一步熟悉JBuilder。这些将引导开发者了解JBuilder的基本操作。Welcome工程还为用户提供了有关提示、技巧和快捷方式等方面的功能。

1.3 AppBrowser

在JBuilder中，有一个窗口叫做AppBrowser（工程浏览器）。本窗口包含开发过程中用到的大多数窗格、工具条和菜单。在这个窗口中，开发者通过使用主工具栏和主菜单可以访问所有其他JBuilder向导和窗口。如图1-3所示。

- Main Menu(主菜单)：开发者可以通过主菜单访问AppBrowser窗口中不能直接访问的JBuilder中的所有其他区域。比如说：所有的Wizard都可以通过主菜单进行访问。同理，大多数设置JBuilder配置的对话框也可以通过主菜单进行访问。一些常用选项除了可以通过主菜单进行访问外，我们还可以通过主工具栏进行访问。实际上，主工具栏就是常用菜单项的快捷方式的一个集合。

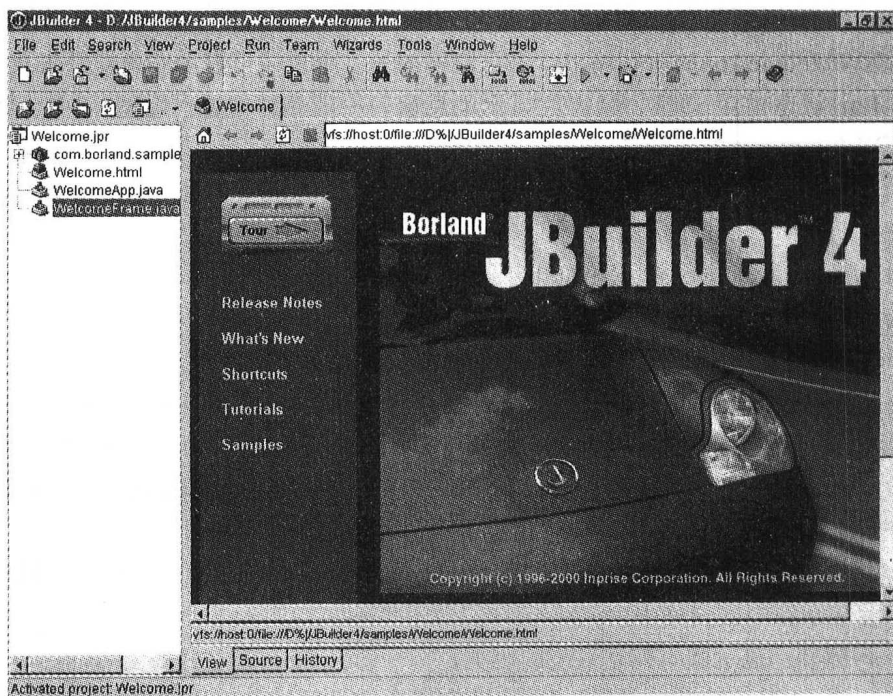


图1-2 JBuilder第一次运行的界面

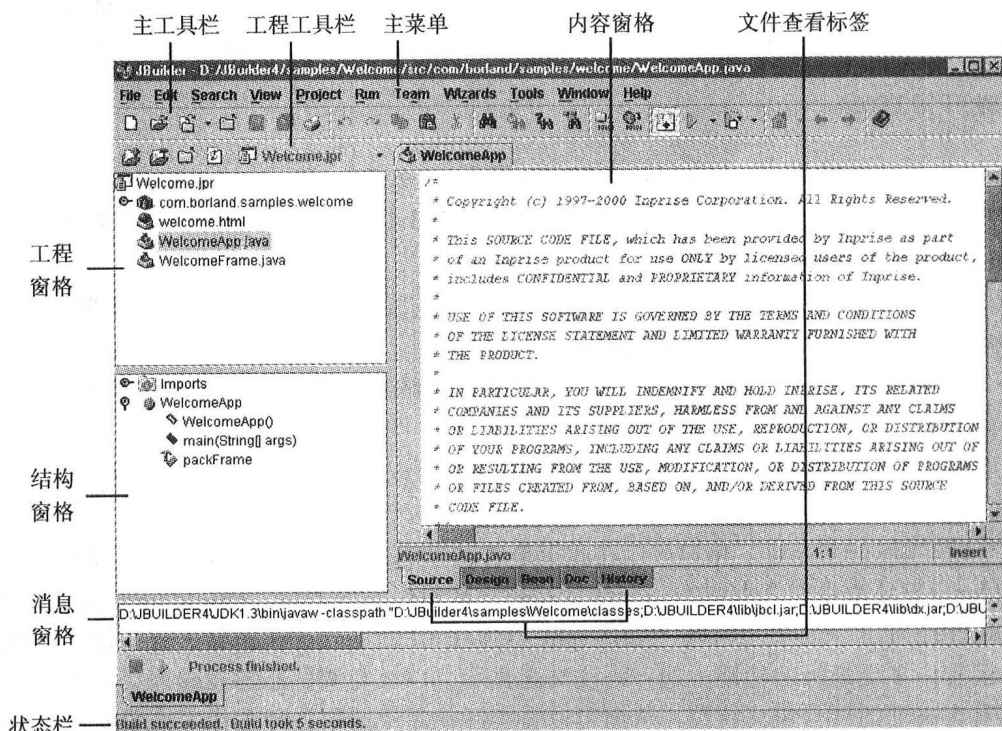


图1-3 AppBrowser窗口

- **Message Pane(消息窗格)**: 显示特定任务产生的详细信息。图1-3中所示的消息窗格会显示应用程序WelcomeApp的所有输出。消息窗格的另一个用途是显示编译器警告及错误信息。
- **Status Bar(状态栏)**: 通常, 状态栏显示消息窗口中消息的概要。在图1-3中, 状态栏显示出程序WelcomeApp的编译已经完成。
- **Project Toolbar(工程工具栏)**: 提供了与工程管理有关的一些常用任务的快捷方式。这个工具栏将在第2章中详细介绍。
- **Project Pane(工程窗格)**: 此窗格列出了当前工程所包含的文件。
- **Content Pane(内容窗格)**: 该窗格包含了以下五个文件查看标签:
 - Source标签**: 显示当前文件的源代码并使之可编辑。
 - Design标签**: 该选项包含组件选项板, 经常用于进行框架、对话框等的可视化设计。
 - Bean标签**: 该标签包含一个BeanExpress Designer (BeanExpress设计器), 它可以使用户查看并编辑当前Bean文件 (比如java源文件) 的属性。有关Bean方面的内容将在第16章中详细介绍。
 - Doc标签**: 显示Javadoc Viewer (Javadoc查看器); 如果存在文档的话, 则当前文件的文档会显示在内容窗格中。
 - History标签**: 显示版本控制系统的历史。
- **Structure Pane(结构窗格)**: 显示当前内容窗格中文件的结构。本窗口有两种模式, 如果内容窗格正在显示源代码, 则结构窗格将显示源代码中元素的层次结构。这些源代码的元素包括变量及方法等。如果内容窗格处于设计模式, 则结构窗格会显示内容窗格中的文件的组件 (如Bean) 的层次结构。这个层次结构也许包括按钮、菜单等组件的实例。

1.4 集成开发环境

JBuilder为开发者提供一个非常灵活的集成开发环境 (IDE), 甚至允许开发者定制开发环境。要改变开发环境的外观, 可以选择主菜单中的Tools | IDE Options...菜单项。一旦选中了该菜单项, 系统就会弹出如图1-4所示对话框:

该对话框有三个标签允许开发者定制开发环境。它们分别是: **Browser(浏览器)**、**File Types(文件类型)**、**Run/Debug(运行/调试)**。

第一个标签即Browser。该标签允许开发者定制AppBrowser的外观 (Look and Feel), 以及AppBrowser的Keymap (键盘映射) 等。

第二个标签即File Types。该选项用于定制JBuilder可以识别的文件扩展名 (文件类型), 如图1-5所示。

第三个标签即Run/Debug。用户可以在此指定进程/状态变化的查询时间间隔, 如图1-6所示。

JBuilder开发环境中非常重要的一个部分就是它的源代码编辑器。JBuilder的源代码编辑器与Borland公司出品的其他开发环境 (如Delphi、C++Builder的代码编辑器) 有些相似。并且这个编辑器还有许多更高级的特性, 如: 语法高亮显示, 以及与调试器的集成等。除此之外, 该编辑器还具有高度的可自定义特性, 从而可以在实际使用中适应各种工作方式的要求。要改变编辑器的工作方式, 开发者可以从主菜单中选择Tools | Editor Options...菜单项。当该菜单项被选

中时系统将弹出如图1-7所示对话框：

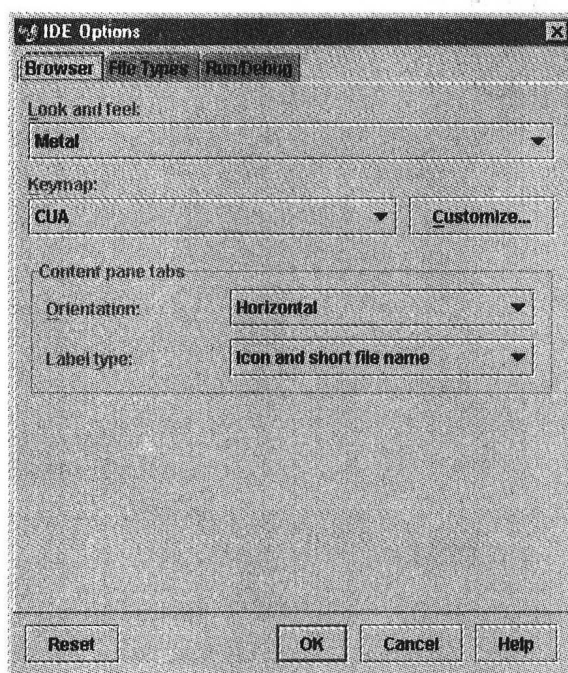


图1-4 IDE Options对话框的Browser标签

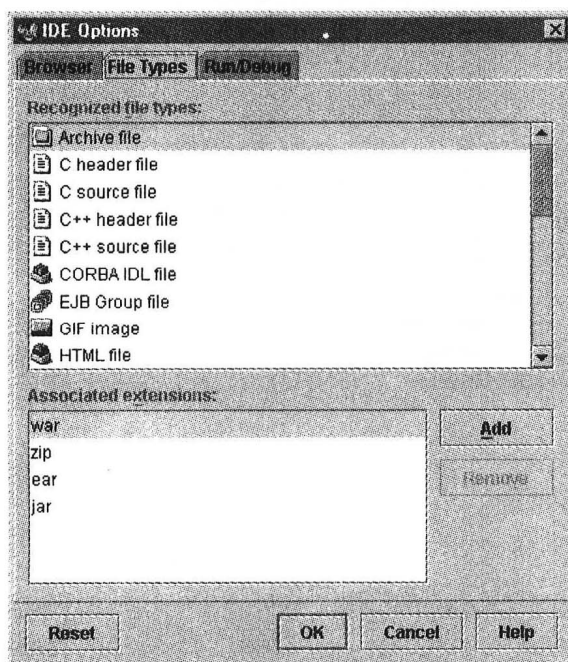


图1-5 IDE Options对话框的File Types标签

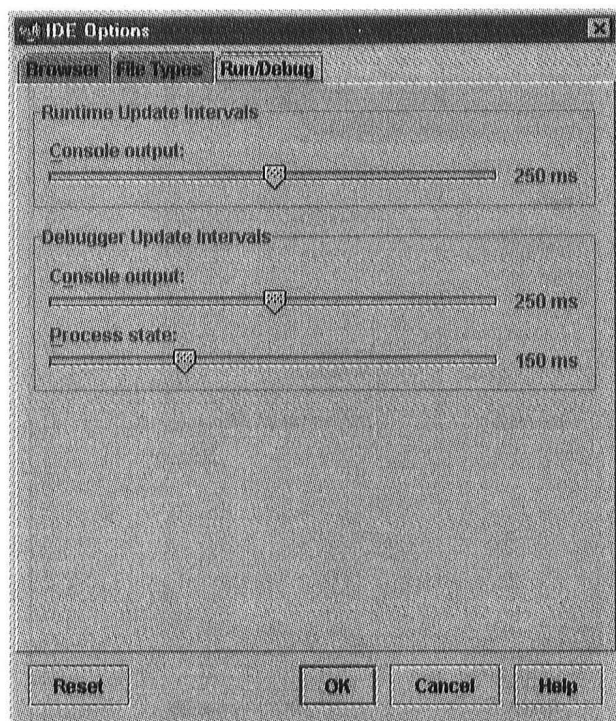


图1-6 IDE Options对话框的Run/Debug标签

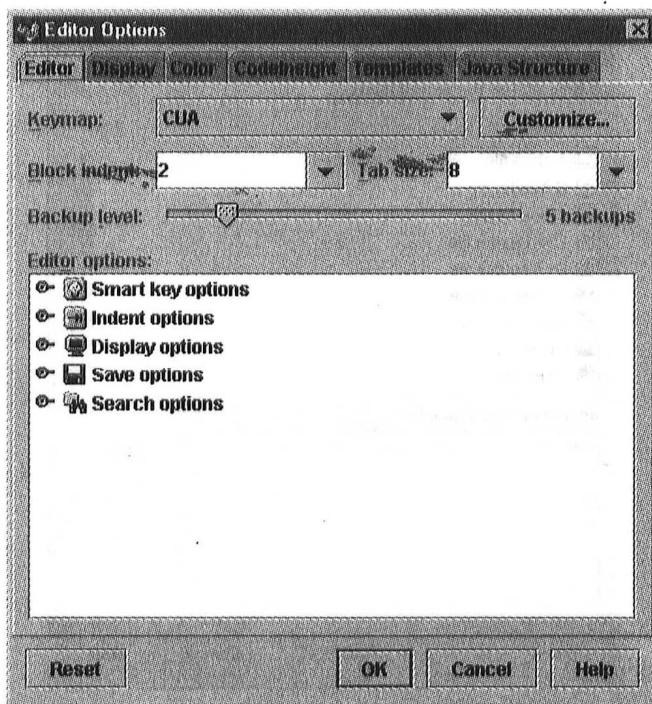


图1-7 Editor Options对话框的Editor标签

从图1-7可以看出, 有多个选项可以对编辑器进行定制。这些选项被分为六个标签。包括: Editor、Display、Color、Code Insight、Templates、Java Structure。

该对话框的第一个标签是Editor。该标签允许用户对代码编辑器的选项进行设置。Keymap选项可以设置键盘布局, 此选项包括: CUA、Emacs、Brief 以及VJ++。Block indent选项用于设置当新建代码段时, 编辑器在新段落首行缩进的空格数。Tab Size选项指明了一个Tab键代表多少个空格。Editor options (编辑器选项) 窗格包含配置选项的层次结构, 用户点击每项最左侧图标可以展开层次, 并查看该选项的详细内容。

对话框的第二个标签是Display (见图1-8)。该标签允许用户对编辑器的另外一些选项进行设置。

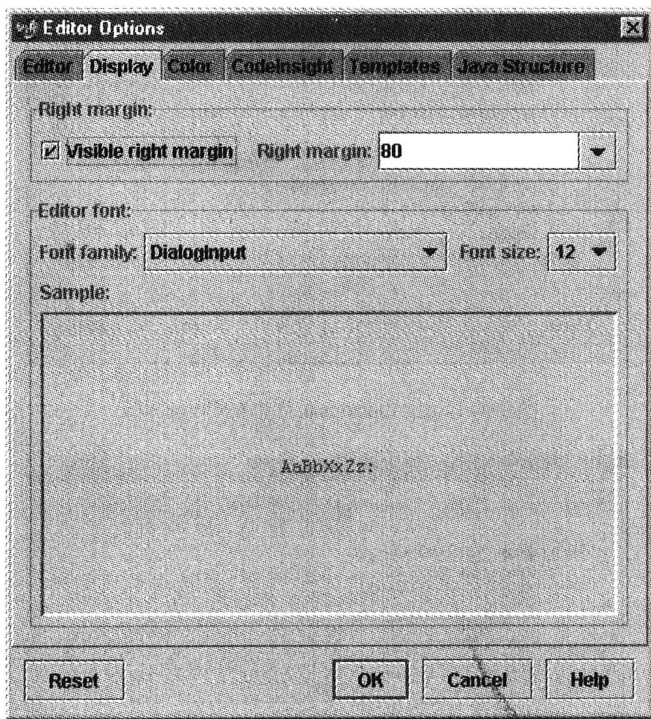


图1-8 Editor Options对话框的Display标签

第三个标签是Color (见图1-9), 该标签允许开发者定制JBuilder的语法高亮显示时所用的颜色。开发者可以在Screen element (屏幕元素) 列表中选择一个元素, 然后指定它的颜色。举例来说, 我们可以选择滚动条, 然后指定它的颜色。这样可以使滚动条的颜色区别于源代码编辑器中的其他元素。

第四个标签是Code Insight (见图1-10)。该标签是JBuilder 2.0的新特点, 它可以帮助开发者编写代码。它为开发者提供一个上下文相关的弹出窗口, 该窗口可以使用户观察并选择成员或者参数。在默认情况下, Code Insight是被启用的。这在程序开发过程中对开发者很有帮助, 因为它只会显示仅适用于当前所给对象或方法的选项。虽然Code Insight是一个优秀的编写代码的辅助方法, 然而该标签的部分或全部功能都是可以进行开关设置的。开发者只需简单地选择