Designed for Microsoft Windows NT Windows 95

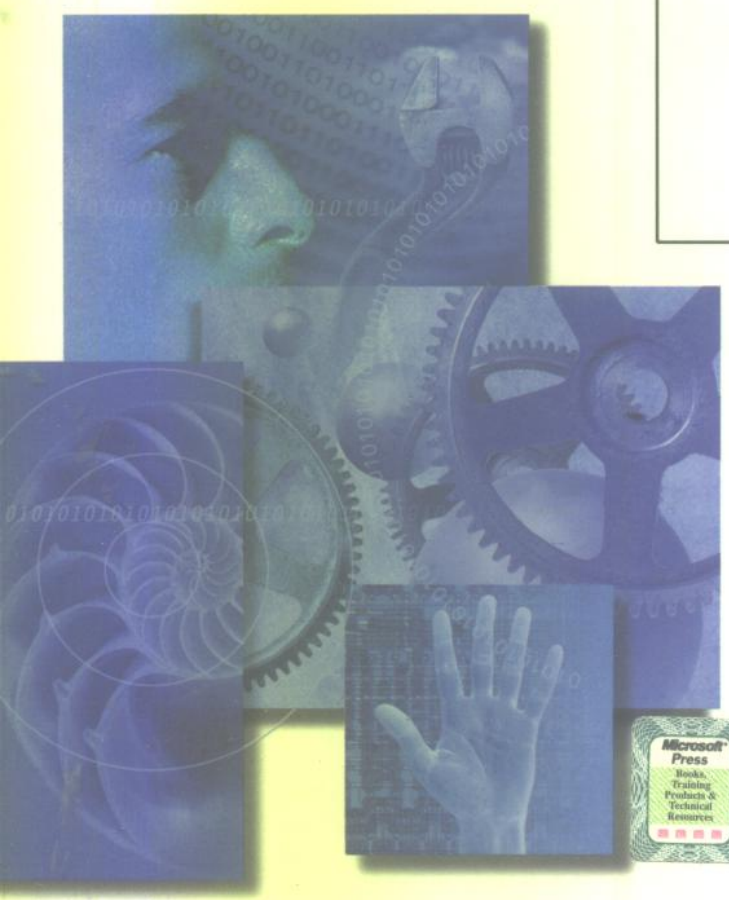
CD-ROM INCLUDED

Microsoft 程序设计系列

For Version 1.1

活学活用 Visual J++™

利用 Java™, ActiveX™和组件进行完整的应用程序开发



[美] Scott Robert Ladd 著
克立兹软件研究所 译

清华大学出版社
http://www.tup.tsinghua.edu.cn

Microsoft Press

(京)新登字 158 号

内 容 提 要

Java 是目前最流行的程序设计语言之一。Visual J++ 1.1 是 Microsoft 公司推出的 Java 开发平台工具。本书是关于 Java 和 Visual J++ 1.1 的一本专著,它首先介绍了 Java 语言的基础知识,然后介绍 Java 与 JavaBean 和 ActiveX 等组件之间的编程技术,最后介绍 Java 的高级应用技术,如图形用户界面、图形显示、数字签名等。

本书叙述简洁,工具先进,程序详尽,适合所有 Java 技术人员参考。

活学活用 Visual J++
Active Visual J++
Scott Robert Ladd

图书在版编目(CIP)数据

活学活用 Visual J++ /(美)莱德(Ladd, S.R.)著;克立兹软件研究所译 . -北京:清华大学出版社,1999.1

ISBN 7-302-03260-2

Ⅰ.活… Ⅱ.①莱… ②克… Ⅲ.J 语言 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(98)第 37308 号

出 版 者:清华大学出版社(北京清华大学校内,邮编 100084)
http://www.tup.tsinghua.edu.cn
责任编辑:胡先福
印 刷 者:清华大学印刷厂
发 行 者:新华书店总店北京发行所
开 本:787×960 1/16 印张:21.5 字数:471 千字
版 次:1999 年 1 月第 1 版 1999 年 1 月第 1 次印刷
书 号:ISBN 7-302-03260-2/TP·1747
印 数:0001~5000
定 价:50.00 元(带光盘)

译者前言

Java 是目前最流行的程序设计语言之一。Visual J++ 1.1 是 Microsoft 公司推出的 Java 开发平台工具。Microsoft 公司现在正处在软件业的霸主地位，Visual J++ 在 Java 开发工具市场中的地位自不待言。本书以 Visual J++ 支持的功能为基础，并以它为全书的编程环境，介绍了 Java 语言基础，Java 与 ActiveX 和 JavaBeans 等组件之间的编程技术以及 Java 的高级应用，如 AWT 和数字签名等。通过阅读本书，读者可以掌握以下几个方面的内容：

- Visual J++ 1.1 的新特征和新功能，以及如何充分利用这些新特征和新功能。
- Java 与 ActiveX 和 JavaBeans 等组件模型之间的编程技术。
- 利用可重用组件来构造小程序和正规应用程序。

虽然我们竭力想使本书的翻译完美无缺，错误和不当仍在所难免，敬请读者不吝赐教。

译　者

1998 年 1 月

关于作者

在过去的10年中，Scott Robert Ladd出版了数以百计的文章和超过一打的计算机程序设计书籍。他在70年代中期开始程序设计，他所使用过的系统包括PDP-8s和IBM大型机直到最新的PC技术。当他不研究编译器和软件时，Scott喜欢天文学和徒步旅行。Scott与他的妻子及三个女儿住在Colorado（科罗拉多州）的Silverton，这是位于Colorado的西南群山上的一个历史性的矿业小镇。

作者献词

在70年代中期，个人计算机还是凤毛麟角，但我所在小镇的高中与附近的大学利用现在看来堪为落伍的电传打字机建立起了联系。我的父母看到我对这种技术的痴迷，就给我买回了一台个人计算机，以此为基础，我开创了一项事业。20年后的今天，我仍在砰砰地敲出字符，只不过是在日渐复杂的机器上。

是我父母的支持，而非其他的原因，才使这本书能够呈现于读者的面前。在此，我谨将它献给我的母亲和父亲，无论何时何地当我想享受为人子的欢乐时，他们都在我的身边。愿所有的孩子都能如此幸运。

前　言

本书是有关人类之间和机器之间的接口的。首先是编程人员和 Java 语言之间的接口，这将在本书的第一部分介绍，它将教给你面向对象的 Java 编程思想。另一个接口存在于软件组件之间，这将在本书的第二部分介绍，它探讨了利用 COM（Component Object Model，组件对象模型）和更通用的 JavaBeans 技术产生独立软件组件的 Java 模型。第三部分介绍了最终的接口，该接口处于应用程序及其用户之间，即应用程序的外观及其给人的感觉，它表现为利用 Abstract Window Toolkit（抽象 Window 工具包）来开发 Java 程序。

至目前为止，Java 主要用于产生美观的视觉效果和漂亮的 Web 页。如果你浏览 Web，你就会感受到正规应用程序的可爱。这种状况部分是因为 Java 相对年轻：从其产生之日到现在刚刚两年。然而，从其问世以来，Java 发展得非常之快，以至于几乎每天都有新的语言特性及包出现。在这样一种令人头晕目眩的情况下，无疑 Java 的编程人员仍在不断学习他们利用这件新工具的能力。与权威们关于 Java 不适应于大型团体的抱怨相反，Java正在逐渐向适用于所有开发任务的目标迈进。本书谈到了用 Java 编写正规应用程序的必要性，它能创造性地、有效地创建复杂的、有用的程序。在以后的章节中，将展示如何利用 Java 开展工作，以及如何摆脱它立足于 Web 的桎梏而取得更大的进展。

读代码片段和小段程序不能使你学会怎样写 Java 应用程序，我通常喜欢通过给出程序来教授编程技巧，而在本书中我就是这样做的。其中的一些程序范例很大，但它们包含了基础性的技巧。我希望这些应用程序会是有趣的，带有挑战性的，甚至带有某种娱乐性。

Scott Robert Ladd
Silverton, Colorado
1997 年 6 月
Srladd@frontier.net
http://www.frontier.net/~srladd

目　录

第一部分　面向对象程序设计

第 1 章　体验 Java …… 2
1.1　作为通信系统的 Internet …… 2
1.2　Web 程序设计 …… 3
1.2.1　用 Java 进行虚拟计算 …… 3
1.2.2　为什么要用 Java …… 4
1.3　面向对象程序设计 …… 5
1.3.1　面向函数的程序 …… 5
1.3.2　用对象进行程序设计 …… 6
1.3.3　范例 …… 7
1.4　展望 …… 16
第 2 章　Java 类设计 …… 17
2.1　类 …… 17
2.1.1　类语法 …… 17
2.1.2　对象 …… 19
2.1.3　重载 …… 20
2.1.4　初始化函数和构造函数 …… 22
2.1.5　终结函数 …… 24
2.1.6　类和实例成员 …… 25
2.1.7　常数 …… 27
2.1.8　访问修饰符 …… 28
2.2　继承 …… 29
2.2.1　继承限定符 …… 30
2.2.2　范例 …… 30
2.3　界面 …… 40
2.4　Microsoft Visual J++ 和类 …… 43
2.5　均匀偏差：一个示例 …… 46
2.6　一些面向对象的思想 …… 56
2.7　展望 …… 57

第 3 章 Java 不是 C++ …… 58
3.1 不同点 …… 58
3.1.1 程序结构 …… 59
3.1.2 类型 …… 60
3.1.3 数组 …… 62
3.1.4 串和字符 …… 63
3.2 把 C++ 程序转换成 Java …… 67
3.2.1 分数 …… 68
3.2.2 C++ 分数类 …… 68
3.2.3 Java 分数类 …… 80
3.3 展望 …… 88
第 4 章 Java 的实际应用 …… 89
4.1 问题 …… 89
4.2 工具:遗传算法 …… 90
4.3 初步设计 …… 90
4.4 解决方案 …… 92
4.4.1 染色体 …… 92
4.4.2 轮盘 …… 97
4.4.3 再生 …… 101
4.4.4 变异 …… 109
4.4.5 群体 …… 114
4.5 GATest 应用程序 …… 117
4.6 展望 …… 127

第二部分 面向组件的 Java

第 5 章 Java 和 ActiveX …… 130
5.1 动态内容 …… 130
5.1.1 历史 …… 130
5.1.2 进入 ActiveX …… 131
5.1.3 COM 基础 …… 131
5.2 连接动态内容和 Java …… 133
5.2.1 Visual Basic,脚本编辑和 Java 小程序 …… 133
5.2.2 在 Java 下使用 COM 对象 …… 134
5.2.3 ActiveX 控件和 Java …… 138
5.3 范例 …… 139

5.3.1 控件 …… 140
5.3.2 Web 页 …… 140
5.3.3 Java 小程序 …… 142
5.4 展望 …… 148
第 6 章 了解 COM 和 Java …… 149
6.1 COM 的基础知识 …… 149
6.1.1 界面和类标志符 …… 151
6.1.2 Java 对 COM 对象的访问 …… 151
6.1.3 IUnknown …… 152
6.2 界面定义 …… 154
6.2.1 类型库 …… 154
6.2.2 IDL 文件 …… 154
6.3 关于界面的更多说明 …… 161
6.3.1 引用计数 …… 161
6.3.2 对象一致性 …… 162
6.3.3 属性 …… 162
6.3.4 聚集 …… 162
6.4 展望 …… 165
第 7 章 用 Java 创建 COM 对象 …… 166
7.1 COM 替代物 …… 166
7.2 把 Java 小程序转换为 COM …… 176
7.3 使用一个 Java COM 类 …… 183
7.4 展望 …… 185
第 8 章 Java Bean …… 186
8.1 Bean 的定义 …… 186
8.1.1 一个基本的 Bean …… 186
8.1.2 生成一个 JAR …… 188
8.1.3 生成一个好的 Bean …… 189
8.2 事件 …… 189
8.2.1 事件示例 …… 189
8.2.2 事件模型的比较 …… 190
8.3 属性 …… 191
8.3.1 边界属性 …… 191
8.3.2 受约束属性 …… 192
8.4 Slideshow Bean …… 194

8.5 作为 ActiveX 控件的 Bean …… 205
8.6 展望 …… 211

第三部分 应用 Java

第 9 章 抽象窗口工具箱 …… 214
9.1 AWT 组件体系 …… 214
9.1.1 Component 父类 …… 214
9.1.2 Container 子类 …… 218
9.1.3 Canvas 子类 …… 220
9.1.4 Button 子类 …… 220
9.1.5 Checkbox 子类 …… 221
9.1.6 Label 子类 …… 222
9.1.7 TextArea 子类 …… 222
9.1.8 List 子类（列表框） …… 224
9.1.9 Choice 子类（组合框） …… 226
9.2 布局 …… 227
9.2.1 BorderLayout …… 227
9.2.2 FlowLayout …… 228
9.2.3 GridLayout …… 228
9.3 实用类型 …… 229
9.3.1 颜色 …… 229
9.3.2 字体 …… 230
9.3.3 字体规格 …… 231
9.4 展望 …… 232
第 10 章 用户界面设计要素 …… 233
10.1 坐标变换 …… 233
10.2 窗格、窗口和图文框 …… 234
10.2.1 窗格 …… 234
10.2.2 窗口 …… 234
10.2.3 图文框 …… 235
10.3 标准事件 …… 236
10.3.1 Action 事件 …… 237
10.3.2 Component 事件 …… 237
10.3.3 Container 事件 …… 238
10.3.4 Item 事件 …… 238

10.3.5 Key 事件 …… 239
10.3.6 Mouse 事件 …… 240
10.3.7 Window 事件 …… 243
10.4 菜单 …… 245
10.5 展望 …… 252
第 11 章 安全和认证码 …… 253
11.1 Java sandbox …… 253
11.2 ActiveX 和 Internet Explorer …… 254
11.3 代码签名 …… 256
11.3.1 证书类型 …… 259
11.3.2 Cabinet(CAB)文件 …… 260
11.4 展望 …… 261
第 12 章 构造组件 …… 262
12.1 更多的颜色 …… 262
12.2 分隔条 …… 264
12.3 有边界的窗格 …… 266
12.4 展示框架 …… 267
12.5 固定堆栈 …… 268
12.6 检查框列 …… 271
12.7 单元窗格 …… 275
12.8 展望 …… 283
第 13 章 用 Java 编写的“生命(Life)” …… 284
13.1 人工生命 …… 284
13.2 细胞自动机 …… 285
13.2.1 Life 游戏 …… 285
13.2.2 思维 …… 288
13.2.3 一般的自动机 …… 289
13.2.4 拓扑构形 …… 289
13.3 LifeBox 应用程序 …… 289
13.3.1 自动机 …… 290
13.3.2 自动机规则 …… 298
13.3.3 LifeBox …… 306
13.4 展望 …… 328
附录 A Java 1.1 中不推荐的方法 …… 329
附录 B 在 Microsoft Visual J++ 1.1 中使用 Microsoft Java SDK 2.0 …… 332

第一部分

面向对象程序设计

第 1 章

体 验 Java

程序设计不是一件随心所欲的事情。在编写出高质量的软件之前，你必须了解所使用的工具及其原理。那么，Java 是怎样一种计算机技术呢？从最基本的层次上讲，它是一种新型的编程语言，是为在 Internet 上传送信息而设计的。但是 Java 有更广泛的意义，它在很多方面代表着软件开发的现有水平。我把 Java 视为软件开发演化历程中的一个组成部分。在这个开篇章节里，我将把 Java 的来龙去脉作一介绍，这样的话，你不但能了解这种新型开发工具是什么，还能了解它的起源。

1.1 作为通信系统的 Internet

技术是使人类文明成为可能的放大器。我们自身力量不能举起的东西，可以利用机器把它举起来；如果听众离我们太远而无法听到我们，可以利用设备来传送声音。如果有什么东西能把人类同其他物种分离开来的话，那就是我们所依赖并痴迷于的想象创造力。

> **“技术不是现实世界的一种描述，**
> **而是操作它的一种方式。”**
> **Octavio Paz(1914—)**
> **墨西哥诺贝尔奖诗人**

我们很少会知道某种新技术会把我们带向何方。Internet 最早是一种在军事、教育和政府部门间进行可靠通信的系统。那时，Internet 仅对那些掌握了它的神秘接口的人有用，而且它的绝大多数信息都是令人乏味的文本。这就是为什么当时绝大多数的人们不知道或者说没有注意到世界上的计算机可以互相彼此通信的原因。当个人计算机用户的机器具有图形界面时，谁还会想看文本文件并键入神秘的指令呢？即便到了 1992 年，当我同计算机用户谈到 Internet 时，也仅能听到诸如“那对我有什么用呢？”之类的问题。

80 年代末，欧洲粒子物理实验室(原先被称为 CERN)的科学家们产生了一个想法：开发一种用来描述图形化信息页的通用语言，并编制一个称为“浏览器”的程序来解释“超文本标志语言(HyperText Markup Language ，即 HTML)”。CERN 的发明可以让 Internet 用户创

建并观看“点击(point-and-click)”文档,这种文档可以包含彩色图像、格式化文本以及其他内容。研究人员把他们的 Internet 扩展部分命名为 World Wide Web。

他们不曾想象 Web 今天的普及程度,谁也不能让人们停止谈论 Internet! 这个系统目前充满了从药方到太空图片之类的各种东西,广告商试图向你兜售新汽车和新的观念,等等。每天,Web 的使用方式都有所增加。

Internet 是所谓技术放大器的一个突出例子,它是一种在并不相识的人们之间传递图像和文档的工具。Internet 的价值远远超过了联机销售或学术研究,Internet 的真正价值是它可以让人们交互式地共享观念。一些人可能对随心所欲的智力交流感到可怕;其他人则看到了人类成长和探索的绝好机会。

1.2　Web 程序设计

像任何强壮的生物一样,World Wide Web 也在不断地改进。HTML 的第一个版本仅限于显示文本和图像,在一些人希望他们的 Web 页能够交互之前,这是足够的。这些人认为,Web 不仅能够传送美观的文档,而且还能够分发软件。要实现这一点,他们需要定义一种编写程序的方法,该程序能通过 Internet 连接在所有的计算机系统上运行。

这并不像听起来那么容易。虽然我们中的大多数人在单位和家庭里都使用基于 Intel 芯片的 PC 机,但 Web 遍布世界,应用范围很广。浏览器运行在 Macintosh、工作站、UNIX 系统或一些大型机上,而这些机器在绘图和显示文本时是完全不兼容的。你无法知道别人用什么设备来看你的 Web 主页。让我们设想一下未来:今天的高技术设想将是明天彻头彻尾的废物。如果你非常聪明,并且早作打算,你需要构造的 Internet 软件应该是通用的,即不依赖于某种特定的硬件或操作系统体系结构。

Web 本身的设计提供了答案。作者创建遵循某种标准的 HTML 文档,这些文档在任何运行了支持此标准的浏览器的机器上都可以被阅读。作者不需要知道 Windows、UNIX 或其他系统的复杂技术,他们只需知道联接到 Web 的所有机器都拥有可以显示他们的文档的浏览器。当新的操作系统或硬件体系结构出现时,它的编程人员可以提供一种浏览器程序以访问已有的全部 Web 信息。这比重新编写 Web 中的所有文档以适合每种新机器要现实得多。

Visual J++ 把这个思路又向前拓展了一步,它允许开发人员创建通用软件,就如同 HTML 允许开发人员创建通用文档一样。

1.2.1　用 Java 进行虚拟计算

浏览器是一种虚拟的文档显示器,它阅读 HTML 文档并根据特定计算机的要求来对其进行解释。比浏览器更进一步的是“虚拟机(Virtual Machine)”,你可以把它想象为物理计算机内部的理论计算机。虚拟机软件把通用操作转换成适合给定硬件和软件环境的相

应指令。如果你为虚拟机设计了一个程序,它将可以在任何实现了兼容仿真程序的计算机上运行。编程人员不再需要担心他们的程序能在哪儿运行,他们仅需为虚拟机编写代码就可以了。

Java是一种为虚拟机提供的程序设计语言。在你编写了一个Java程序后(通常被称为Applet,即小程序),你将把它编译成一种特殊的"虚拟机语言"。以这种形式,你的程序将能够在任何实现了此虚拟机的计算机上运行。随着计算机的改进,新的虚拟机将允许旧的Java程序在不做改动的情况下运行。另外,关注的焦点也已发生了变化:不再需要为特定的计算机编写单独的程序,程序设计人员仅需要为虚拟机编写一个程序,而把与硬件相关的细节交给虚拟机开发者去处理。这样就使得软件能够像HTML文档一样通用。

虚拟机还提供另一种东西:一个预定义的程序组件库。所有的编程语言都有一个函数或类型的标准库,它通常在编译时被链接到程序中以提供I/O和视频控制之类的服务。但是,组件库通常是与系统相关的,并且常常是多余的。举例而言,我为UNIX编写的每个C程序都包含了标准的I/O包。即便利用了动态链接技术,每种计算机类型仍都需要标准组件的定制版本。

Java的应用程序接口(API)是一组在虚拟机内实现的功能包。如果你的Java程序使用了标准的I/O和文件包,那么编译过的代码将不再包含这些函数,小程序将带有到虚拟机中的等效函数的链接。这种概念类似于动态链接库(DLL),动态链接库将编译过的代码放入模块中,模块可以被许多程序调用。但DLL代码与系统相关,而Java的API存在于所有的虚拟机之中。

API的设计使Java特别有效率,因为只有唯一的一种程序代码将在Internet上传送。如果API被链接成小程序,就像C++中的库函数一样,人们就必须一次又一次地下载标准包。API函数属于虚拟机,而虚拟机通过一个通用接口有效地实现与系统相关的功能。

1.2.2 为什么要用Java

现在已经有了这么多的语言,为什么还要开发一种新语言呢?在过去10年中,我曾经用十几种语言编写过专业软件,这些语言从FORTRAN和COBOL直到Smalltalk和C++。为什么不用已有的一种程序设计语言来进行Web程序设计呢?C++不是被认为是编写程序的最终工具吗?Java难道不是C++的变种吗?

C++几年来一直是计算机行业的热宠。问世10年来,它与C保持着高度的兼容,并支持C的许多种特性。同样地,C的祖先可以追溯到更古老的语言,直到60年代晚期。只要不陷入继承的陷阱,向后兼容还是一件好事。但不幸的是,C++携带了不少古怪的语法,这就是历史的遗产。C++的ANSI/ISO标准整整占满了我的两栏书架。

而Java拒绝与C和C++兼容,它的语法流畅,并增加了许多迫切需要的新特性。尽管Java的语法大多来自于C++,其目的是通过Internet提供联机软件。用于创建大型应用程序的模板和工具被抛弃了,危险的指针被一种扩充的(并且非常可靠)引用定义所代

替。Java 的 API 库与 C++ 所包含的标准库大为不同。在第 3 章,你将看到对这些差异的详细介绍。

Microsoft Visual J++ 是 Java 的一种交互式开发环境,它集成在 Microsoft Developer Studio 之中。利用 Visual J++,你能够创建 Java 应用程序或小程序。Developer Studio 还提供了一个强大的交互式调试器,以及用于管理 Java 类和资源的工具。传统的 Java 工具是 UNIX命令行模式,但 Visual J++ 能让你利用 Microsoft 公司在 Visual C++ 产品中首次引进的流行工具,在 Microsoft Windows 环境下创建 Java 应用程序。

此外,在定义 Java 时充分考虑了安全性。在当今世界中,你不能轻易相信 Internet 中的程序的可靠性和安全性。因为 Java 程序在加载 Web 页时自动执行,因而引发了各种安全问题。虚拟机可以防止小程序在计算机的特定范围之外操作。对于那些超出虚拟机的限制(例如,通过访问 ActiveX 控件)的小程序,Microsoft Visual J++ 利用认证码(Authenticode)技术实现了一个安全系统。我将在第 11 章详细介绍认证码。

1.3　面向对象程序设计

Java 被称为是一种"面向对象"的编程语言。"面向对象"的含义是什么呢? 它只是一个时髦的行话吗? 或许,绝大多数新的程序设计语言都有其自称"面向对象"的原因?

1.3.1　面向函数的程序

"面向对象"指的是一种计算机程序设计思想。最早的程序实际上没有什么思想——它们在起始处开始执行,并按一般顺序执行到达终点。它们的工作仅是完成范围很窄的任务,比如执行一次运算等。随着程序不断变得复杂, Pascal 和 C 等语言出现了。这些语言允许编程人员把一段软件组织成功能模块。一个具体任务(例如,读取一个数据文件)可以被集中到单个函数中。如果在读取文件的过程中出现了错误,问题的根源可以在函数中找到,这样就简化了调试过程。面向函数的程序设计允许编程人员把函数编译成库。一旦一个函数能够工作,它就可以被保存起来以供任意多的程序使用。

面向函数设计的一个后果是:程序将用结构的形式来定义复杂的信息,并且将定义相关函数的集合。在小程序中,问题还不明显;但在具有几千行代码的大型应用程序中,不可能操作某个特定数据的所有实例。程序越复杂,处理的信息类型就越多,越容易忽视负作用和无意的数据破坏。在许多编程人员同时参与一个项目时,更容易造成混乱和冲突。

为帮助编程人员构造可靠的软件,计算机科学家引入了一种新的设计思想,它基于"数据处理 (data processing)"的字面解释。尽管处理过程当然也重要,但他们感到真正的着眼点应该放在"要处理什么"和"如何处理"之上。换句话说,程序应围绕数据对象来设计。这样,数据类型及其相关函数是面向对象软件开发的焦点。

1.3.2 用对象进行程序设计

面向对象的程序设计随着 Smalltalk 和 C++ 等语言的出现在 80 年代开始起步。“对象(object)”是程序处理的某种事物——一个动作或一种数据记录类型。面向对象程序设计语言允许程序定义描述了内部结构和外部接口的组件。处理数据的函数变成了数据类型的一个部分,用一种称为“封装(encapsulation)”的技术与数据类型密切链接起来。当某个函数是对象的一部分时,它被称为“方法(method)”。这是个新的术语,但是曾经编写过“结构化”代码的人都能理解这种把函数与数据联合起来的思想。面向对象的编程并不像它听起来那么新奇,它只是成功程序设计实践的一个提炼。

从本质上说,面向对象的程序设计改变了程序设计的焦点,由强调一个程序该完成什么工作转移到强调它应操作什么。这里存在着编程思想的转变,可能难以理解。实际上,一个对象是一个黑匣子:你通过对象的方法来访问它,你根本不需要了解对象的内部正在做什么。与此相应的术语叫做“抽象(abstraction)”,你只要编程就会用到它。举例来说,你有多少次需要了解一个浮点数的内部数据结构呢?可能从来就没有过,你只是定义一个变量,并遵照一组规范来使用它。当你编写表达式 2.0+3.5 时,你不需要知道如何把这些值加载到数学处理器——具体的细节由编译程序来处理。同样的原理也适用于你所定义的对象:对象的定义(被称为“类”)通过声明公共可访问的方法来告诉你该对象能接受怎样的操作。你不需要了解黑匣子的内部是怎么回事,你的程序只须利用对象的明确定义展开工作。可以把“类”想作是程序中的一种新数据类型的定义;如果你创建了一个名为 Record 的类,你可以用它来创建对象,就像你利用 double 之类的固有类型一样。

采用面向对象还有另一个原因:它改善了另一种由来已久的实践——代码重利用。面向函数的程序允许编程人员编写一个例程,并把它保存起来以供以后使用,但是如果他们想要改变或增加一个已有例程的工作方式,他们就不得不重新编写它。面向对象的程序设计语言通过采用“可扩展性”的概念解决了这个问题,它允许编程人员在已创建组件的基础上编写程序。编程人员可以从现有类导出一个新的类,通过增加新的或改变过的特性扩展原始的类。以这种方式,新的类继承并增强了已有的软件组件。

让我们假定你需要构造一组类来处理不同格式的文档。你可以先构造一个通用的类,它定义了所有文档类型的共同属性。接下来,你为特殊的文档格式导出新的类:TextDoc、WordDoc 等等。如果必要,TextDoc 可以是 RTFDoc 和 HTMLDoc 的基础。RTFDoc 对象将继承 TextDoc 的特征——并且通过 TextDoc,RTFDoc 还继承了 Document 的特征。

图 1-1 展示了这些类之间的关系。线段发源自底层的基类并且把基类和派生类连接起来,显示了类层次的结构和继承的流程。我在第 2 章将详细介绍类层次的创建,第 4 章则给出了 Java API 包的组织结构。

TextDoc 和 WordDoc 扩展了 Document 类;同样地,RTFDoc 和 HTMLDoc 继承了 TextDoc。