

开发专家
之 Delphi

飞思科技
FECIT Sci-Tech
www.fecit.com.cn

Delphi

高级编程

飞思科技产品研发中心

编著

本书附赠
光盘包括
全部实例
代码



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>

开发专家之 Delphi

Delphi 6 高级编程

飞思科技产品研发中心 编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书属于《开发专家之 Delphi》系列。Delphi 6 是 Borland 公司推出的最新版本, 该版本在以前的基础上进行了升级和改进, 使得它成为 Windows 2000 平台上最受欢迎的开发工具之一。

本书重点以 Delphi 6 的各种实用开发技术为主线。全书分 8 篇, 共 19 章, 主要包括 OLE 自动化、DLL 的使用、多线程编程、Delphi 的文件处理、GUI 开发、多媒体应用的开发、组件开发、网络编程等方面的内容。所附光盘包含书中全部实例源代码。

本书内容翔实, 实例丰富, 解析深刻, 适用于中高级的专业和准专业的程序开发人员, 也可作为网络管理员、系统分析员的技术参考书。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。

版权所有, 翻版必究。

图书在版编目 (CIP) 数据

Delphi 6 高级编程 / 飞思科技产品研发中心编著. —北京: 电子工业出版社, 2002.1

(开发专家之 Delphi)

ISBN 7-5053-7360-9

I .D... II. 飞... III. Delphi 语言—程序设计 IV. TP 312

中国版本图书馆 CIP 数据核字 (2001) 第 093139 号

丛 书 名: 开发专家之 Delphi

书 名: Delphi 6 高级编程

编 著: 飞思科技产品研发中心

责任编辑: 郭 晶 杨章玉

排版制作: 电子工业出版社计算机排版室监制

印 刷 者: 北京大中印刷厂

出版发行: 电子工业出版社 <http://www.phei.com.cn>

北京海淀区万寿路 173 信箱 邮编: 100036

经 销: 各地新华书店

开 本: 787×1092 1/16 印张: 41.25 字数: 1056 千字

版 次: 2002 年 1 月第 1 版 2002 年 1 月第 1 次印刷

书 号: ISBN7-5053-7360-9
TP · 4241

定 价: 58.00 元 (含光盘)

凡购买电子工业出版社的图书, 如有缺页、倒页、脱页、所附磁盘或光盘有问题者, 请向购买书店调换。

若书店售缺, 请与本社发行部联系调换。电话 68279077

出版说明

“开发专家”是电子工业出版社计算机图书研发部长期以来精心培育的计算机科学技术类本版品牌。这个品牌是由多个专题系列组成的横向大系列，涵盖了计算机技术的各个方面，特别是一直受到极大关注的程序开发类系列，例如《开发专家之数据库》、《开发专家之网络编程》、《开发专家之 Delphi》以及《开发专家之 Sun ONE》等。这些专题系列基于各自的角度，从纵向上包含了该专题的所有内容。因此，整个“开发专家”的品牌架构纵横交错，囊括了所有的计算机技术和所有的技术层面，海纳百川而又极具可扩展性。

“开发专家”的作者队伍主要依托于“飞思科技产品研发中心”。“飞思科技产品研发中心”是由专业的策划人员、权威的技术专家和资深的作者队伍共同构成。在图书的出版上，形成了以研发为基础、以出版为中心、以服务为支持的专业化出版框架和流程。通过深入的市场调查和技术跟踪，在综合了技术需求和读者焦点等因素的基础上，形成各系列丛书的写作重点和大纲，然后聘请业界的最前沿学者进行写作。同时，策划工作全程介入写作进程，严格控制写作质量，用最专业的技术背景、最深刻的理论基础、最具代表性的案例、最能为专业读者接受的形式，为读者提供品质最佳的图书产品，体现了出版者和作者的完美结合。

多年来，计算机图书研发部始终把创造社会效益摆在首位，秉承一切为国内计算机技术专业读者服务的精神，为推动国内 IT 技术发展、为体现国内技术的原创水平，穷尽所有的创意与努力，将出版者的命运与读者的支持紧紧地连在了一起。

在此，我们临出版之残酷竞争而不惧，旌旗猎猎而异军突起，这与广大读者的支持是分不开的。为使我们的脚步更坚实、使我们的队伍永葆活力和创造力，我们期待着您能为我们的前进贡献出您的意见和建议。同时，我们也在等待着您的加入。

我们的联系方式：

电 话： (010) 68134545

E-mail: support@fecit.com.cn

网 址: <http://www.fecit.com.cn> <http://www.fecit.net>

电子工业出版社计算机图书研发部

前 言

关于本套丛书

Delphi 是美国 Borland 公司推出的功能强大的应用程序开发工具。它具有功能强大、运行速度快、易于学习和使用以及开发效率高等特点。Delphi 是可视化应用编程开发环境、可重用性面向对象编程语言、快速编译器和数据库的完美结合。

Delphi 6 于 2001 年 6 月发布。新版本的 Delphi 6 开发功能更加强大，除了能够有效帮助开发者个人或其开发团队快速建立 Windows 应用程序，快速简化 Windows 与浏览器客户、Web 服务器、中间件以及后台数据库系统的集成等这些传统意义上的开发之外，Delphi 6 是目前惟一全面支持所有主要工业标准（XML、SOAP、WSDL 和 XSL 等）的开发工具，同时，也支持基于 Web 服务的 Microsoft .NET 和 Sun ONE 体系，而且提供给 Web 开发者需要的可伸缩性与可靠性。新的 Delphi 6 框架中还包括了 BizSnap、WebSnap 和 DataSnap，用户可以用它开发支持 Web 服务特性的服务器端和客户端应用，而这一切都是通过一套高度集成的可视化开发工具、先进的编译技术和可重用的组件完成的。特别是在电子商务愈加流行的今天，通过 Delphi 6，任何企业都能很快地转移到未来基于 Web 服务的电子商务应用程序开发上，而不用丢弃以往的开发方式、技巧和源代码。可见，要创建一流的 Web 应用程序，Delphi 6 无疑是目前的最佳选择之一。

《开发专家之 Delphi》系列丛书就是针对 Delphi 6 的整个开发体系和特色进行组织的，涵盖了 Delphi 开发所有重要方面，为开发人员提供了完整的知识架构，无论要进行怎样的 Delphi 开发，都可以在这套丛书中找到明确的解决方案和经验之谈。

同时，专业的作者队伍、完整的解决方案和详尽的实例剖析是这套丛书高质量的基础和保证，也是本套丛书的最大特色所在。

总之，哪怕是最熟练的程序员也需要专业的技术文献，这套丛书就是经典的开发经验及实例的集成，所以必将受到专业人士的关注和欢迎。

关于本书

《Delphi 6 高级编程》涵盖了 Delphi 专业编程的最重要的几个方面。

全书共分 8 篇。

第一篇介绍 OLE 自动化，这是 Windows 应用程序之间相互操作的一个典型技巧。

第二篇介绍了 DLL 的使用，重点讨论了 Win32 动态链接库 DLL 的工作原理，DLL 的创建、使用以及调试等各方面的要点，这是用来编写 Windows 应用程序的关键组成部分。

第三篇介绍了多线程编程，主要包括线程的基本概念、TThread 对象、多线程的管理及其多线程在数据库领域的应用等问题。

第四篇 Delphi 的文件处理，对处理不同类型的文件进行了详细介绍，包括如何使用 TFileStream 类封装文件的输入/输出以及如何利用 32 位 Windows 的主要特征——内存映射文件，还提供了一些关于选择驱动器、在树型目录中查找文件以及获得版本信息的方法。

第五篇 GUI 开发，详细介绍了如何开发出用户自己的图形效果。要特别指出的是，用 Delphi 6 进行图形界面设计是相当方便的，这是为人称道的优点之一。

第六篇多媒体应用开发，包括了视频和音频两方面的内容，这也是 Delphi 6 开发的特长之一。

第七篇组件开发，详细介绍了可视化组件的开发过程，为高级程序员提供了大型应用程序开发的可用性和技巧，本篇还介绍了功能强大的 ActiveX 控件。

第八篇网络编程，详细介绍了基于 TCP/IP 和 Socket 的网络编程，这是目前很流行的编程开发方向和应用。

全书包含了 Delphi 专业编程者的所有开发实例和经验，非常有实用价值。

本书由飞思科技产品研发中心策划并组织编写，徐敦忠、杨袅、陆正武等参加了本书的写作，陆正中对全书进行了统稿。同时，在本书的写作过程中得到了刘文智先生的大力支持和协助，他提出了大量的参考性意见更使本书增色不少，在此表示衷心的感谢。

当然，限于作者水平，加之时间仓促，书中不足之处难免，敬请读者批评指正。

我们的联系方式：

电 话： (010) 68134545 68134811

E-mail: support@fecit.com.cn

网 址: <http://www.fecit.com.cn> <http://www.fecit.net>

飞思科技产品研发中心

目 录

第一篇	OLE 自动化.....	1
第 1 章	OLE 自动化.....	3
1.1	创建 OLE 自动化对象.....	3
1.2	OLE 对象的两种状态.....	5
1.3	对象链接与嵌入的比较.....	5
1.4	OLE 容器.....	6
1.5	OLE 的数据类型.....	9
1.5.1	OLE 的基本数据类型.....	10
1.5.2	OleVariant 类型.....	10
1.6	OleVariant 类型的应用.....	13
1.6.1	创建 OLE Automation 对象的实例.....	13
1.6.2	访问 Automation 对象的特性.....	14
1.6.3	访问 Automation 对象的方法.....	15
第 2 章	开发 OLE 自动化服务器程序.....	17
2.1	编辑类型库.....	17
2.2	操纵 Automation 对象.....	18
2.2.1	添加新的属性.....	18
2.2.2	添加新的方法.....	19
2.2.3	添加新的事件.....	19
2.3	制作一个简单的自动化客户.....	20
2.4	制作一个简单的自动化服务器.....	23
第 3 章	OLE 在 Office 系统中的实现.....	33
3.1	OLE 文件.....	33
3.2	Delphi 6 与 Office 系统的连接.....	34
3.2.1	设计多文件窗体.....	34
3.2.2	代码实现.....	34
3.2.3	运行程序.....	44
3.3	导出数据到 Office 系统.....	46
3.3.1	从 Delphi 6 导出数据到 Word 2000.....	46
3.3.2	从 Delphi 6 导出数据到 Excel 2000.....	51
3.3.3	从 Delphi 6 导出数据到 PowerPoint 2000.....	53
第二篇	DLL 的使用.....	59
第 4 章	DLL 概述.....	61
4.1	认识 DLL.....	61
4.2	为什么使用 DLL.....	62

4.2.1	共享代码、资源和数据	63
4.2.2	隐藏实现的细节	63
4.2.3	自定义构件	63
4.3	设置 DLL 的首选基地址	64
4.4	DLL 工程文件	65
4.4.1	DLL 工程文件与普通 Delphi 工程文件的区别	65
4.4.2	DLL 中的变量	66
4.4.3	DLL 与 EXE 文件	67
4.5	在 Delphi 中创建简单的 DLL	67
4.5.1	使用关键字 Exports	67
4.5.2	创建 DLL	68
4.5.3	DLL 的初始化代码	72
4.5.4	例程——创建一个简单的 DLL	73
4.6	显示 DLL 中的窗体	76
4.6.1	显示 DLL 中的模式窗体	76
4.6.2	显示 DLL 中的非模式窗体	78
第 5 章	窗体重用的实现	81
5.1	重用 DLL 的设计	81
5.1.1	口令设置窗体	81
5.1.2	口令检查窗体	84
5.1.3	保存 DLL	84
5.2	重用 DLL 的调用	85
第 6 章	DLL 的调用及调试	87
6.1	调用 DLL	87
6.1.1	静态调用 DLL	87
6.1.2	动态调用 DLL	91
6.1.3	静态调用与动态调用的区别	97
6.2	访问 DLL	97
6.2.1	DLL 的入出口函数	97
6.2.2	一个允许共享数据的 DLL	102
6.2.3	访问 DLL 中的共享数据	104
6.2.4	访问 DLL 中的对象及方法	107
6.3	DLL 调试技巧	112
6.3.1	设置调试选项	113
6.3.2	使用内部调试器	114
6.3.3	单步执行	117
6.3.4	使用监视窗口	117
6.3.5	其他的调试工具	118
6.3.6	调试 DLL	119

第三篇 多线程编程	121
第 7 章 Windows 系统的消息、事件和多任务	123
7.1 进程与线程	123
7.2 Win32 的内存管理	126
7.3 Win32 的消息	127
7.3.1 Win32 的消息机制	127
7.3.2 Windows 的消息处理	130
7.3.3 消息分类和消息句柄	133
7.3.4 消息陷阱	135
7.4 Windows 事件	137
7.5 发送消息与时间处理	138
第 8 章 TThread 对象	141
8.1 在应用程序中使用多线程	141
8.2 线程的控制	141
8.2.1 线程的创建	142
8.2.2 线程的挂起和恢复	146
8.2.3 线程的终止	147
8.3 设置线程的优先级	149
8.4 线程的调度	151
8.4.1 抢先式调度	151
8.4.2 非抢先式调度	151
8.5 测试线程代码的执行时间	151
8.6 设计一个简单的多线程	153
8.6.1 设计窗体	153
8.6.2 创建线程对象	154
8.6.3 添加事件代码	156
8.6.4 运行程序	156
8.7 复杂例程——利用多线程排序	157
8.7.1 设计窗体	157
8.7.2 程序实现	158
8.7.3 运行程序	165
8.8 多线程与数据库	165
第 9 章 多线程的管理	171
9.1 线程的局部存储	171
9.1.1 缓存线程对象	171
9.1.2 线程的局部变量	172
9.1.3 局部存储示例	173
9.2 多线程的同步	175
9.2.1 多线程的同步机制	175

9.2.2	使用 Synchronize 方法	176
9.2.3	使用 VCL 类的 Lock 对象方法	177
9.2.4	使用 WaitFor 方法	177
9.2.5	利用 Windows API 实现同步	182
第四篇	Delphi 6 的文件处理	191
第 10 章	文件处理	193
10.1	文件的输入/输出	193
10.1.1	文本文件的处理	193
10.1.2	类型文件的处理	197
10.1.3	无类型文件的处理	198
10.1.4	TFileStream 类	201
10.1.5	应用举例	202
10.2	内存映射文件	208
10.2.1	内存映射文件的应用	209
10.2.2	创建/打开内存映射文件	209
10.2.3	创建文件映射对象	210
10.2.4	建立文件视图的映射	210
10.2.5	取消文件视图的映射	211
10.2.6	关闭文件映射对象	211
10.2.7	内存映射文件的一致性	211
10.2.8	内存映射文件实例	212
10.2.9	一个文本搜索程序	214
10.3	目录和驱动器	222
10.3.1	获得可用的驱动器列表	222
10.3.2	获取驱动器信息	223
10.3.3	获取 Windows 目录的信息	225
10.3.4	获取 System 目录所在位置	226
10.3.5	获取当前目录	226
10.3.6	在目录中查找文件	227
10.3.7	复制和删除目录树	230
10.3.8	获取文件的版本信息	235
第五篇	GUI 开发	245
第 11 章	坐标系统和映射模型	247
11.1	常用的坐标系统与坐标转换	247
11.1.1	屏幕坐标系	247
11.1.2	窗口客户区坐标系	247
11.1.3	逻辑坐标系	248
11.1.4	窗体坐标系	248
11.2	GDI 坐标系统和映射模式	248

11.3 设置映射模式	250
第 12 章 图像显示与图形绘制	251
12.1 图像组件 TImage	251
12.1.1 TImage 组件的属性	252
12.1.2 TImage 组件的方法	270
12.1.3 TImage 组件的事件	273
12.2 TImage 组件的应用	275
12.2.1 TImage 组件在 VCL 库中的地位	275
12.2.2 TImage 组件应用举例	276
12.3 图形组件——TCanvas	296
12.3.1 TCanvas 的属性	297
12.3.2 TCanvas 的方法	307
12.3.3 TCanvas 控件的事件	312
12.4 TCanvas 控件应用举例	312
第 13 章 高级图形程序开发技术	337
13.1 编写绘画程序	337
13.2 编写动画程序	345
13.2.1 第一种实现：绘制—擦除法	346
13.2.2 第二种实现：TAnimate 组件的使用	352
13.3 制作字体动画	356
第六篇 多媒体应用开发	361
第 14 章 创建媒体播放器	363
14.1 多媒体的基本概念	363
14.1.1 多媒体系统的层次结构	363
14.1.2 数字视频的基本概念	364
14.1.3 数字音频的基本概念	364
14.2 TMediaPlayer (媒体播放器) 组件	364
14.2.1 TMediaPlayer 组件概述	364
14.2.2 TMediaPlayer 组件的属性	365
14.2.3 TMediaPlayer 组件的方法	375
14.2.4 TMediaPlayer 组件的事件	382
14.3 制作媒体播放器	383
第 15 章 播放视频和音频	395
15.1 播放音频	395
15.1.1 用 TMediaPlayer 播放 WAV 文件	395
15.1.2 用 PlaySound 函数播放 WAV 文件	396
15.2 播放视频	398
15.3 制作 CD 播放器	403
第七篇 组件、控件开发	407

第 16 章 组件开发	409
16.1 Delphi 组件编程概述	409
16.1.1 Delphi 可视组件类库	409
16.1.2 建立组件的起点	410
16.1.3 手工创建组件	411
16.1.4 测试未安装的组件	412
16.1.5 测试已安装的组件	414
16.2 面向对象技术	414
16.2.1 组件的访问控制	414
16.2.2 派送方法	417
16.2.3 创建属性	419
第 17 章 组件设计	447
17.1 创建图形组件	447
17.1.1 创建和注册组件	447
17.1.2 公布继承属性	448
17.1.3 增加图形功能	448
17.2 定制表格组件	450
17.2.1 创建并注册组件	450
17.2.2 发布继承的属性	451
17.2.3 改变初始值	452
17.2.4 调整单元格大小	453
17.2.5 填充单元格	454
17.2.6 跟踪日期	454
17.2.7 浏览月和年	459
17.2.8 浏览日期	459
第八篇 网络编程	463
第 18 章 利用 Delphi 6 设计 Web 浏览器	465
18.1 浏览器的工作原理	465
18.1.1 URL 与域名系统	466
18.1.2 HTTP 协议	469
18.2 浏览器的设计	490
18.2.1 浏览器组件介绍	491
18.2.2 设计 Web 浏览器	503
18.2.3 页面生成组件	530
第 19 章 设计 Web 服务器	549
19.1 服务器工作原理	549
19.1.1 Internet 组件	550
19.1.2 服务器的调度与管理	552
19.2 应用层协议控件及应用	578

19.2.1	TNMFTP	578
19.2.2	TNMUDP	599
19.2.3	TNMNNTP	604
19.2.4	XML	618
19.3	电子邮件系统的开发	625
19.3.1	TNMSMTP	625
19.3.2	TNMPOP3	632
19.3.3	一个复杂的 E-mail 收发程序	638

第一篇 OLE 自动化

OLE Automation 是 Windows 应用程序之间相互操纵的一个技巧。被操纵的一段为 Automation 服务器，典型的 Automation 服务器有 Microsoft Office 程序和 Internet Explorer。而操纵 Automation 服务器的一端称为 Automation 控制器。一个程序可以兼具服务器和控制器两种角色。

在 Delphi 6 中，要创建 Automation 控制器（即 Automation 客户），有两种方式：一是引入 Automation 服务器的类型库；二是 OleVariant 类型的变量。

第 1 章 OLE 自动化

在 Windows 中，应用程序之间是相互联系的。然而，越来越多的程序提供一个 OLE 自动化接口，以使得其他应用程序可以对它们进行操作。这样，我们可以使用 Delphi、C++、Visual Basic 等语言来驱动 OLE 自动化服务器，而不必考虑用于编写它们的编程语言。

1.1 创建 OLE 自动化对象

在 Delphi 中，实现 OLE 自动化是非常简单的，无论何时，为了支持 OLE 自动化，Delphi 提供了一个简单的向导——功能很强的类型库编辑器。

要打开类型库编辑器，其具体操作步骤如下。



- (1) 选择【File】→【New】→【Other】命令，打开“New Items”对话框。
- (2) 选择“ActiveX”标签页下的“Automation Object”图标。
- (3) 单击【OK】按钮，这时 Delphi 即可打开类型库编辑器，如图 1-1 所示。

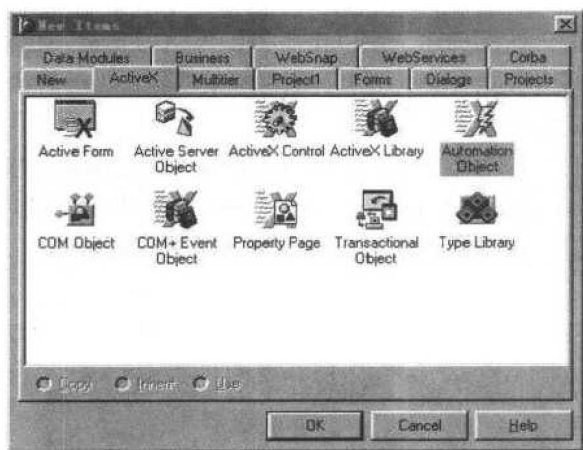


图 1-1 选中“Automation Object”图标

在图 1-2 中可以看到，Delphi 6 可以自动生成 OLE 自动化服务器，该服务器也可以输出事件。此时只需选择自动化对象向导，Delphi 6 将在类型库及其生成的源代码中添加相应的项目。

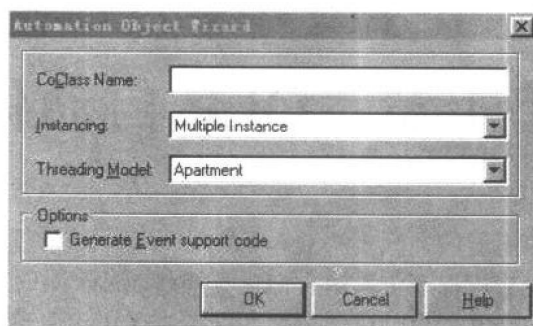


图 1-2 打开“自动化对象向导”对话框

当使用进程间 DLL 时，客户机应用程序可以使用服务器并直接调用其对象方法，因为它们处在相同的地址空间中。当使用 OLE 自动化时，由于客户机与服务器是两个分别运行在相同地址空间中的独立应用程序。因此，系统必须使用一种复杂的机制来进行对象方法调用。在 OLE Automation 中，客户机有两种基本方法来调用服务器输出的“对象方法”：

- 要求执行“对象方法”与 DLL 动态调用的方法类似。当使用 Variant 调用 OLE 自动化服务器时，就使用该方法。该技术的使用相对简单，但执行效率很低，而且提供的编译器类型监测也非常少。
- 在服务器上为对象输出 Delphi 接口定义，并只需分配一个数值就可以调用其“对象方法”。该技术基于接口，允许编译器监测参数类型并生成执行速度更快的代码，但这无疑会增大程序员的工作量。

在以上两种方法之间，最重要的区别在于第二种方法通常需要一个类型库，类型库是类型信息的集合，该集合通常描述了可以由服务器使用的元素，包括对象、接口以及其他类型信息。类型库是与语言无关的。类型元素由 OLE 定义为编程语言标准元素的子集合，而且它们可以交付给任何开发工具使用。

在前台编写的 OLE 自动化控制器没有与它所使用的服务器相关的类型信息。这就要求在后台，必须向使用 `Idispatch.Invoke` 的服务器发送每个函数调用，将函数名称作为字符串参数传递，并希望名称与服务器的已有函数相对应，例如：

```
var
  VarWord: Variant;
begin
  VarWord := CreateOleObject('Mydocuments.Basic');
  VarWord.FileNew;
  VarWord.Insert('执行宏调用');
```

在第 2 行变量类型声明中，Variant 是一种变体数据类型，它可以假定其值为不同的数据类型，包括支持 `Idispatch` 接口的 COM 对象。Variant 是运行时的类型检测，这也是编译器不知道 OLE 自动化服务器的对象方法时仍然可以编译代码的原因。

一旦编译器进行了检测，它就可以使用两种不同的技术向服务器发送请求。它可以使用一个普通的 `Vtable`（也就是一个接口类型中的项），或使用一个 `dispinterface`，`dispinterface` 是一种将接口的每个入口映射为数字的方法。此时，对服务器的调用就可以使用数字发送了。可以认为这是一种中间技术，即在使用函数名称发送与在 `Vtable` 中使用直接调用之