程不功　程江　编著

Delphi 3.0

面向对象程序设计

国防工业出版社

Delphi 3.0 面向对象程序设计

程不功　程江　编著

国防工业出版社

·北京·

图书在版编目(CIP)数据

Delphi 3.0 面向对象程序设计/程不功;程江编著 . -
北京:国防工业出版社,1998.8
ISBN 7-118-01921-6

Ⅰ.D… Ⅱ.①程… ②程… Ⅲ. 面向对象语言,Delphi
-程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(98)第 09404 号

国防工业出版社出版发行
(北京市海淀区紫竹院南路 23 号)
(邮政编码 100044)
北京怀柔新华印刷厂印刷
新华书店经售
*
开本 787×1092 1/16 印张 13¼ 301 千字
1998 年 8 月第 1 版 1998 年 8 月北京第 1 次印刷
印数:1—4000 册 定价:18.00 元

前　言

半个多世纪以来，计算机科学一直处于高速发展之中。然而和硬件的发展相比，软件的发展速度显得要慢得多。

集成块以及大规模集成电路的出现，把硬件推上了新的“快车道”，从此它的体积成倍地减小，成本成倍地降低，而功能却成倍地提高。现在要装备一台微型计算机已不是件困难的事，一个较熟练的技术人员一天装上几台微型计算机已不算什么。

软件方面，尽管在计算机语言、软件工具和系统等方面都有很大发展，然而到目前为止，开发应用软件的手段却仍然十分落后，开发小组开发一个中等规模的系统少则几个月，多则半年、一年。这固然与软件本身的要求和复杂性有关，但是程序设计方法落后不能不说是一个重要原因。

80 年代提出、近几年才普及开来的面向对象程序设计方法给人以面目一新之感。什么是面向对象程序设计方法呢？简单地说，就是利用“软件集成块”设计程序。用面向对象方法开发程序时，不再是从头开始，一句句拼装语句，而是利用已有成果（软件集成块），高起点地前进，因此能在较短时间内开发出高质量的软件。

目前已出现了一大批面向对象或基于面向对象的工具，如 VB、Delphi、PowerBuilder、Java 等。它们出现的时间虽然还不太长，但已显示出强大的生命力。

Delphi 是一个快速开发 Windows 应用程序的工具，它采用完全面向对象的方法，还将可视化、事件驱动、代码自动生成等最先进的技术综合为一体。作者和很多程序设计者一样，多次用它开发程序以后已经深深地喜欢上了这个系统。

目前市场上已经有一些介绍 Delphi 的图书，但大多数都是从组件出发来介绍系统。本书则想从另一个角度——应用程序的需要出发，按照设计的步骤，通过大量的示例来介绍系统，以增强学习的针对性，使读者能在较短的时间内获得最必要的知识。

本书的第一、七、八、九章为基本理论；第二～六章为界面设计；第十～十六章为数据库管理；第十七～二十二章为文件管理、报表打印、多媒体等的程序设计。本书重点讲述程序设计方法，但也适当地介绍有关理论，使读者不仅学会方法，而且掌握面向对象设计方法的理论实质。

Delphi 目前已经发行了三个版本 1.0、2.0、3.0，本书以 3.0 版本为准，但也指出 2 .0 和 1.0 版本中的不同点。

本书最适合于自学的读者，也可作为大专院校的教材或参考书。作者曾用它作教材对计算机专业学生授课，取得很好的效果。

作　者

目　录

第一章　面向对象程序设计概述 …… 1
第一节　什么是 Delphi …… 1
第二节　程序设计方法发展简史 …… 1
第三节　面向对象程序设计的基本概念 …… 2
第四节　面向对象程序设计的特点和优点 …… 4
第二章　编程环境 …… 7
第一节　对安装环境的要求 …… 7
第二节　集成环境 …… 8
第三节　编程环境的调整 …… 11
第三章　简单界面程序设计 …… 12
第一节　应用程序的组成——项目 …… 12
第二节　开发应用程序的过程 …… 14
第三节　程序的运行和终止 …… 17
第四节　有关组件介绍 …… 18
第五节　程序设计示例 …… 21
第四章　多窗体程序设计 …… 24
第一节　在项目中增添或删除窗体 …… 24
第二节　多窗体之间的引用 …… 25
第三节　父子窗体中的几个特殊问题 …… 28
第四节　窗体的重用 …… 29
第五节　多窗体程序设计示例 …… 31
第五章　菜单及信息对话窗口的设计 …… 32
第一节　主菜单设计 …… 32
第二节　菜单的归并 …… 34
第三节　弹出式菜单 …… 35
第四节　几个信息对话窗口 …… 35
第六章　绘图 …… 40
第一节　TCanvas 组件的属性 …… 40
第二节　绘图方法 …… 42
第三节　综合示例 …… 43
第七章　进一步认识类及代码单元 …… 46
第一节　从记录到类 …… 46
第二节　进一步认识代码单元 …… 48

第三节　综合示例 …… 52
第八章　VCL 类库与 Object Pascal 语言 …… 56
第一节　类库 VCL 的层次结构 …… 56
第二节　Borland Object Pascal 简介 …… 57
第三节　类库浏览器 …… 61
第九章　创建新组件 …… 63
第一节　创建组件的时机和特点 …… 63
第二节　创建组件的起点 …… 64
第三节　创建新组件的“雏形” …… 64
第四节　增加新属性 …… 67
第五节　修改事件的行为 …… 71
第六节　增加新方法 …… 73
第七节　综合示例 …… 73
第八节　其它几个有关问题 …… 77
第十章　Delphi 的数据库工具 …… 79
第一节　Delphi 数据库管理系统的特点 …… 79
第二节　数据库工作桌面(DBD) …… 80
第三节　Borland 数据库机 …… 83
第四节　数据库考察器 …… 86
第十一章　Delphi 简单数据库设计 …… 88
第一节　数据库的结构设计 …… 88
第二节　建立主索引和次索引 …… 90
第三节　设置有效性检查及口令 …… 92
第四节　在数据库环境下输入部分记录 …… 92
第五节　数据库的窗体设计 …… 93
第六节　DBGrid 字段编辑 …… 100
第十二章　数据库中多表的同步 …… 102
第一节　数据库窗体专家 …… 102
第二节　用程序实现主/从库的同步 …… 105
第三节　“一对多再对多”关系的同步 …… 106
第四节　程序中对数据库的操作 …… 106
第十三章　利用 TTable 组件进行查询 …… 110
第一节　顺序查询 …… 110
第二节　快速查询 …… 113
第三节　指定范围浏览 …… 115
第四节　程序设计示例 …… 116
第十四章　利用 TQuery 组件进行查询 …… 121
第一节　SQL 语言简介 …… 121
第二节　用 TQuery 组件进行静态查询 …… 124

第三节　建立 SQL 编辑器 …… 126
第四节　用 TQuery 组件进行动态查询 …… 128
第五节　用 TTable 与 TQuery 实现主/从库的同步查询 …… 129
第十五章　数据库统计与备份 …… 131
第一节　图表显示组件 TCHartFX …… 131
第二节　数据库统计程序示例 …… 133
第三节　数据批量移动组件 TBatchMove …… 135
第四节　数据库备份程序示例 …… 136
第十六章　决策支持 …… 139
第一节　有关组件介绍 …… 139
第二节　综合示例 …… 140
第十七章　文件管理程序的设计 …… 150
第一节　Delphi 中直接调用的 DOS 命令 …… 150
第二节　目录界面的通用设计 …… 150
第三节　窗体对文件的存/取管理 …… 152
第四节　字符串的查询与替换 …… 154
第五节　直接运行 *.exe 文件或显示 *.txt 文件 …… 158
第十八章　报表打印 …… 159
第一节　TReport 组件介绍 …… 159
第二节　设计报表格式 …… 160
第三节　生成报表 …… 165
第四节　ReportSmith 与 SQL …… 167
第五节　增加临时字段 …… 167
第六节　报表中增加图片 …… 169
第十九章　多媒体程序设计 …… 170
第一节　在数据库中使用图像 …… 170
第二节　音乐(响)程序 …… 172
第三节　运行视频(AVI)文件 …… 174
第二十章　动态数据交换(DDE) …… 176
第一节　如何建立通信管道 …… 176
第二节　数据交换的方法 …… 179
第二十一章　对象的链接和嵌入(OLE)简介 …… 182
第一节　基本概念 …… 182
第二节　建立 OLE 容器 …… 183
第二十二章　程序调试 …… 188
第一节　编译时给错误定位 …… 188
第二节　程序运行中给错误定位 …… 190
附录 A　菜单项说明 …… 193
附录 B　组件说明 …… 198
参考文献 …… 202

第一章　面向对象程序设计概述

本章将结合 Delphi[⊖]系统，介绍面向对象程序设计追求的目标、设计方法及其优点，以使读者对这些概念有一个初步的了解。这些概念将贯穿于全书各个章节，因此只有当你学习了后面的章节以后才会逐步加深对这些概念的认识和理解。

第一节　什么是 Delphi

Delphi 是一种快速开发 Windows 应用软件的工具。Borland 公司在 1994 年的国际会议上首次推出了它的 1.0 版。并于 1995 年、1997 年分别推出了基于 Windows 95 和 Windows NT 3.51 或以上的、32 位的 Delphi 2.0 和 Delphi 3.0 版。

Delphi 采用完全面向对象的程序设计方式，并综合可视化、事件驱动、程序代码自动生成等最先进技术，加上它强大的数据库和网络数据库功能（客户机/服务器功能），高效的编译器以及强大的对异常处理的能力，为程序设计提供了一个十分理想的工作平台。在这个平台上，程序设计者能用最简易的方法、最短的时间，开发出界面优美、运行可靠、功能强大的 Windows 应用软件。因此，Delphi 一出现就立即得到了计算机界的极高评价，并以巨大的魅力吸引着程序设计人员。

第二节　程序设计方法发展简史

一种新的程序设计方法的产生不是偶然的，它常常是为满足某种客观上的需要，并在一定硬件的基础之上发展起来的。

最早使用的设计方法是面向过程的设计方法。当时的软件规模较小，硬件的能力也有限，只要将指令按照问题的求解顺序排列即可。衡量程序好坏的标准，主要看它能否做到指令条数少，存储量省，执行速度快。作为面向过程的程序设计语言，最初是机器语言，而后是汇编语言、高级语言。高级语言包括 Basic、Fortran 等。

60 年代，随着一系列大型软件应用中出现的问题，在计算机界引发了一场关于“软件危机”的大讨论。讨论的结论表明，面向过程的程序设计方法需要进一步规范，否则很难保证大型程序的可靠性。由此产生了结构化程序设计。

1969 年，Dijkstra 首先提出了结构化程序设计的概念。一个所谓好的结构程序，是指结构清晰，易于理解也易于验证的程序。结构化程序设计简称为 SPP（Structured Procedural Programming）。

⊖ Delphi 读作['delfai]，原是古希腊一个城市的名字，当时的古希腊人以为 Delphi 位于世界的中心。Borland 公司用它为软件命名，是一种隐喻。

从效率上看，一个好的结构化程序“时空”效率不一定高。但是，它能提高程序的可靠性，便于阅读、检查和维护。用结构化程序设计时，采用自顶而下，逐步分解的办法，将一个大程序划分成若干功能模块。为保证模块之间有条不紊地调用，各模块均只能有单一的入口，其中只采用顺序、选择和循环三种基本控制结构。模块化程序设计语言很多，如 Pascal、C、Module－2 等。

与此同时还出现了函数式程序设计（如 Lisp）与逻辑程序设计（如 Prolog）等，但由于各种原因，这两种设计方法均没有得到普遍地采用。

80 年代提出的面向对象程序设计，是结构化程序设计方法的延伸和发展。面向对象程序设计方法简称为 OOP（Object－Oriented Programming），其概念主要来自 Simula 67 语言中的类（Class）以及 Smalltalk 语言中的对象（Object）。目前，面向对象程序设计的语言已有不少，如 Smalltalk－80，由 C 发展成的 C＋＋，由 Pascal 发展成的 Object Pascal 等。

第三节　面向对象程序设计的基本概念

面向对象程序设计的出发点和追求的基本目标，是使人们认识系统的方法与设计和实现这个系统的方法尽可能接近，也就是使描述问题的问题空间和解决问题的方法空间在结构上尽可能一致。基本方法是：对问题空间进行自然分割，对客观事物进行结构模拟，建立问题域模型，从而使设计出的软件尽可能直接地描述现实世界，构造出模块化的、可重用的、维护性好的软件，从而降低软件的复杂性、提高软件开发效率并减少开发维护的费用。

一、对象（Object）及封装（Encapsulation）

世界上的事物，小到一个苹果大到一个系统，我们都称之为“实体”。世界就是由这些实体及实体之间的相互关系构成的。为了将客观世界映射到程序空间中来，首先要找到一种描述客观实体的办法。

人们认识实体的方法很多，一种常用的方法就是通过它们的属性和行为来识别它们。以一辆汽车为例，它的外形、结构是它的属性；它能载人、运货是它的行为。怎样来模拟这种认识呢？用单纯的数据或方法已经不够用了，应该找到一种新的结构来描述它们，这种结构就是程序中的“对象”。

程序中的“对象”，是数据与方法的统一体，是数据和方法的进一步概括和抽象。方法在这里是函数或过程的统称。对象将数据和处理这些数据的方法封装在一个逻辑实体中，用其中的数据（或称数据结构）来描述实体的属性，用其中的方法来描述实体的行为。一个对象就对应于客观世界中的一个实体。

什么叫对象的封装呢？对象的封装是指逻辑上的“信息隐藏”，封装把内部的实现方法与对外的界面分离开来。从对象的外部不能直接访问对象内部的数据，而只能按照对象提供的接口，通过内部的方法间接地访问它们。同时，对象内部对实现方法的修改也不会波及到对象外面来。

对象的封装增强了对象的独立性，也只有封装才使得对象内部的数据与方法有机地组合成一体。

二、消息(Message)

客观世界中，实体之间靠消息取得联系。模拟这一点，对象之间也用消息传递，并通过消息的传递达到协同动作的目的。在 Windows 环境下运行的程序，各类事件的发生（如用户的操作、某个规定时间的到达、对象状态的改变等）都将发出相应的消息，而消息又是引发对象产生某种动作的信号。可以说对象的行为是靠消息（或称事件）驱动的。

例如用鼠标在某个按钮上单击一下，这一事件就发出了一条明确的消息：消息类型——鼠标单击；接受消息的对象——按钮。

从形式上看，消息驱动与结构化程序中的过程调用有些相似，但是它们之间却有着本质的区别。结构化程序中的过程调用所涉及的过程及函数都属于同一个程序实体，它们之间的调用带有一定的强制性。被调用的模块被动地接收传来的参数，并以固定的方式执行调用命令，执行完毕以后，还必须将控制权返回给调用者。而对象之间的消息驱动是实体之间的一种合作和协同的关系，当对象收到来自外部的消息后响应还是不响应，以及如何响应等均由对象自行确定，发出消息者无权干预。对象的封装和多态，赋予了对象这种智能因素——选择的权利。

传统程序的函数和过程之间相互依赖，整个程序是一种“紧耦合”的关系；而面向对象程序中的对象彼此独立，整个程序结构是“松耦合”的关系。

三、类(Class)

有些对象具有相似的特点，可用“类”将这些共同点概括起来。类就是这些相似对象的共同描述，或者说类是一组对象的“模板”。由对象概括成类，是由个性到共性的抽象。“类”整体地代表一组对象，但只定义它们的数据结构和方法，不接收也不存储具体值。

对象则是类的实例(Instance)，从类派生出对象是从共性再回到个性的过程。派生出的对象除拥有类定义的全部特点外，还可以扩展和补充自己特有的数据和方法。

类是抽象的概念，而对象代表具体事物。只有从类派生成对象后，才能接收和存储具体的值。这种关系与传统程序中的数据类型与变量之间的关系相似。类相当于数据类型，对象相当于变量。有的语言使用抽象数据类型(ADT)，抽象数据类型既包括数据，又包括处理数据的函数和过程，它与类有更多的相似点。

例如，汽车作为类，代表着汽车甲、汽车乙的共性；而汽车甲、汽车乙是派生出的对象，除拥有汽车的共性以外，还可拥有自己的某些特殊性。窗口作为一种类，代表着窗口 1、窗口 2 的共性；而窗口 1、窗口 2 除具有窗口的共性以外，还可以增加自己新的数据和方法。

四、继承(Inheritence)

继承是指从已定义的类导出一个新类时，新类将自动包括原有类的全部数据和方法的一种属性。

继承是一个传递的过程。“类”可以导出“子类”，子类又可导出自己的子类；类与类之间既有平行的兄弟关系，又可有前辈和后代的层次关系，其关系如同一棵倒立的树(树根在上)。这样，越是后面的类，包括的数据和方法越丰富。继承使得类从纵的方向建立了

联系，这种联系在传统设计方法中是没有的。

由 Delphi 提供的“类库”，是一个庞大的类的层次体系。Delphi 通过类库给编程人员提供了强有力的支持(关于类库的情况见第八章)。

五、多态(Polymorphism)

多态是指允许存在同名而行为方式不同的方法。它可表现于一种类派生的多个对象之间，还可表现于类的继承之间。虚拟继承是多态的重要方面。在虚拟继承中，可先为动作赋一个名称(对外接口)，这个名称为继承的各个层次中的对象所共享，但是各对象又可对它的实现方法重新定义，用适合自身需要的方式操作(参见第八章)。

多态增加了类使用中的灵活性。

第四节　面向对象程序设计的特点和优点

一、面向对象程序设计的特点——组装对象

在传统的编程语言中，程序从上而下编写，程序运行时沿着解决问题的过程从前到后移动，直到任务完成。过程是程序的主线，数据结构紧紧围绕这条主线而存在，并为提高处理过程的效率服务。

在面向对象的程序中，不存在独立的数据和过程，数据和过程都从属于对象，被封装在一个个对象之中。对象是数据与处理这些数据的统一体，在这里数据和过程并重。整个程序只由对象组成，对象间的联系通过消息传递进行。系统运行就是多个对象经过消息传递，共同合作，完成某一处理活动的过程。建立程序的方法就是组装对象，并建立对象之间的消息联系。从宏观上看，包括三个步骤：

(1)将相同特性的对象抽象成类，建立“类库”；

(2)从类派生出对象组建程序；

(3)为对象处理消息编写程序段。

由于系统已经提供了类库，因此对于大多数程序设计者来说，只需完成第二、三步。即先组装对象，然后确定对消息的响应方式。在“类库”的支持下，这两个任务都非常容易完成。

上述过程我们可以用硬件来作类比。比如我们将一套软件系统比作一台计算机，对象就相当于集成块或插件板(有人将对象称为“软件集成块”)。用现代方法组装计算机，面对的是一块块集成块(或插件板)而不是一个个分立元件。同样，面向对象的程序设计，面对的是一个个对象，而不再是程序中的一条条语句。

二、面向对象程序设计的优点

面向对象不仅是一种程序设计方法，还是一种分析方法、思维方法和构造系统的方法。从程序设计的角度看，面向对象程序设计最大的优点是“软件重用”(Software Reuseable)方面的突破。

软件重用能提高程序的开发效率，增强程序的可靠性，因此长期以来一直是各类程序设计方法追求的目标，但一直未能找到突破口。结构化程序在这方面曾经做了大量的工

作，建立了庞大的函数库和头文件，供各应用程序调用，在软件重用的工作方面取得了不小的成绩。然而用结构化程序开发一个应用系统时，仍然不得不从头开始，一句句拼装语句，基本上没有摆脱“手工业”的落后模式。

问题在哪里呢？问题就在于模块的不完整性（数据与方法分离），对应用环境强烈的依赖性（紧合），再加上以过程为中心（过程常常是程序中易变的因素），大大降低了模块的可重用度。

与此相反，面向对象程序中对象的完整性和封装，以及对象之间的较为松散的耦合关系，给软件重用提供了极为有利的条件和环境。

以 Delphi 为例，它提供的软件重用功能极为强大，具体表现在三个层次：类的实例、类的继承和窗体或项目的重用。

对象是一个与应用环境无关、紧密封装的模块。类作为对象的抽象，可作为独立于应用环境的标准块进行设计、开发、测试和说明，并可以在市场上发售以供许多应用环境使用。

从类派生成对象，是软件重用的主要方面。Delphi 2.0 提供了 100 种以上、Delphi 3.0提供了 130 种以上的可视组件（类）。这些类不仅体系完整，能满足各类程序的需要，而且每种都经过精心设计和严格测试，并以图标的形式放置在程序设计的集成环境中。利用这些组件就意味着利用已有的成果，使程序设计“站在巨人的肩膀上”前进，不仅速度快、质量高，而且运行可靠。和传统程序中编写枯燥的代码截然不同，面向对象加可视化的设计过程简直可以说变成了一种“享受”。

一个新类可以在原有类的基础上扩展。新类从某个功能接近的类中派生出来，只需定义增加的数据和方法部分，其它部分将自动继承。继承在这里就是一种重用。当然创立新类并不是经常要做的事，只有从类库中找不到合适的类，或者经常需要为类增加同样的数据和方法才能满足需要时，才有必要开发新类（创建新组件见第九章）。

窗体或项目的重用是部分设计的重用。一个样板窗体已经包括若干对象，程序设计者可以在它上面修改和扩展新对象。缺省情况下，系统已提供了若干通用的样板。程序设计者也可随时保留自己的窗体作为样板，以便在以后的设计中重用。窗体的重用方法见第四章。

面向对象程序设计的上述优点，大大提高了软件的可靠性，加快了软件的开发过程，促使软件产业发展到一个新的水平。同时还给“软件工程”提出了新的思想，注入了新的活力。

程序设计准备阶段中的需求分析，是程序设计的基础，但同时又是最难做好的环节。在这一阶段中，要求用户和设计者共同确定出尽可能准确的设计目标，然后由程序设计者进行设计。设计的过程专业性强，主要是和代码打交道，往往只能由程序设计者单独进行，用户无法参与。其结果很可能是，设计人员费了九牛二虎之力，好不容易将一个比较完整的系统交给用户，反馈回来的却是一大堆不满意的信息。

其实，这种情况是很难避免的，因为用户不经过实际应用，常常提不出完整而准确的要求。即使提出来了，设计者也不一定能完全理解。

面向对象加上可视化的程序设计方法，可以使程序设计采用更为开放的方式，让用户有更多机会参与设计过程，使得程序在设计者与用户的交互过程中逐步完善。用户的更

多参与不仅成为可能，而且变成了一种对双方都有利的设计手段。

软件的维护与扩展，在社会高速发展的今天已经变得更为重要。现在的用户已不满足于“交钥匙”的软件产品，它还要为不久后可能出现的新需要，为扩展系统作准备。面向对象程序中的对象组装方式以及“松耦合”的程序环境，为程序的修改和扩充提供了极为便利的条件。

当前计算机正朝着分布式处理、并行处理、网络化、智能化和软件生产工程化等方向发展，而面向对象方法是实现这些目标的关键技术之一。因此可以预计，在未来的相当一段时间内，面向对象必将成为程序设计的主流方向。

最后要指出的是，尽管面向对象程序设计有如此多的优点，然而它的出现在相当长的一段时间内没有引起软件产业的重视，原因之一在于面向对象程序设计要求更大的硬件支持，例如需要大的内存空间等。另外，开发出的产品运行效率不高也制约了它的适用范围。

今天，硬件的发展已经解决了高效率、大容量、低成本的问题，软件方面高效编译、动态链接等技术也日趋成熟，面向对象程序设计因为有了这些强大的物质基础和技术支持，才得到了蓬勃的发展。

第二章　编 程 环 境

本章将介绍 Delphi 的安装环境和编程环境，重点讲述 Delphi 3.0，同时介绍 Delphi 2.0和 Delphi 1.0 版本的不同之处。本章要讨论的问题有：

(1)对安装环境的要求；

(2)编程环境；

(3)编程环境的调整。

第一节　对安装环境的要求

Delphi 必须在 Windows 环境中以光盘（CD－ROM）作为源盘安装。在文件管理器中，用鼠标双击 Delphi 的 Install. exe 文件即可开始安装。安装过程中，用户根据提示，在 Complete（全部安装），Minimum（最小安装）或 Custom（选择安装）三种方式中任选其一。

不同的安装版本以及不同的安装选择，对系统配置的要求也不相同。

一、安装 Delphi3.0(2.0)版本

由于 Delphi3.0(2.0)版本工作在 32 位的计算机上，因此它的最小配置要求是：

(1)中文或英文 Windows 95 或 WindowsNT 3.51 或以上的版本，或者与它们 100% 兼容的系统；

(2)CPU 80486 或以上；

(3)内存最少 8MB(若需用 Client/Server 版，最少 16MB)；

(4)全部安装时，磁盘可用空间 3.0 版本至少为 123MB(2.0 版本至少为 111MB)。

安装过程中，系统将提示缺省的目录，并允许用户更改这些目录(建议不修改)。

二、安装 Delphi 1.0 版本

由于 Delphi 1.0 版本可工作在 16 位或高于 16 位的计算机上，因此它的配置要求是：

(1)中文或英文 Windows 3.1 以上，或者与它们 100%兼容的其它系统；

(2)CPU 80386 以上；

(3)内存最少 6MB(采用 Client/Server 版本时最少 8MB)；

(4)磁盘可用空间至少 30MB(全部安装时至少 80MB)。

安装完毕后，还必须在 CONFIG.SYS 文件中写上下列语句，以保证数据库共享：

INSTALL = C: \ Dos \ Share.exe/F:4096/L:40

其中 4096 为文件共享空间，40 为文件锁定数目。

第二节　集 成 环 境

Delphi 的编程环境是一个集成环境(Integrated Development Enviroment,IDE)。所谓"集成环境"是指:程序设计所需工具的大部分以及求助系统都集中在一个环境中,从程序的编辑、编译到运行也都在同一个环境中进行,因此操作起来非常简便。

整个编程环境包括 6 个组成部分,如图 2-1 所示:

①主菜单(Main Menu);

②快捷键组(Speedbar);

③组件板(Component Palette);

④窗体(Form);

⑤对象检阅器(Object Inspector);

⑥程序编辑器(Code Editor)。

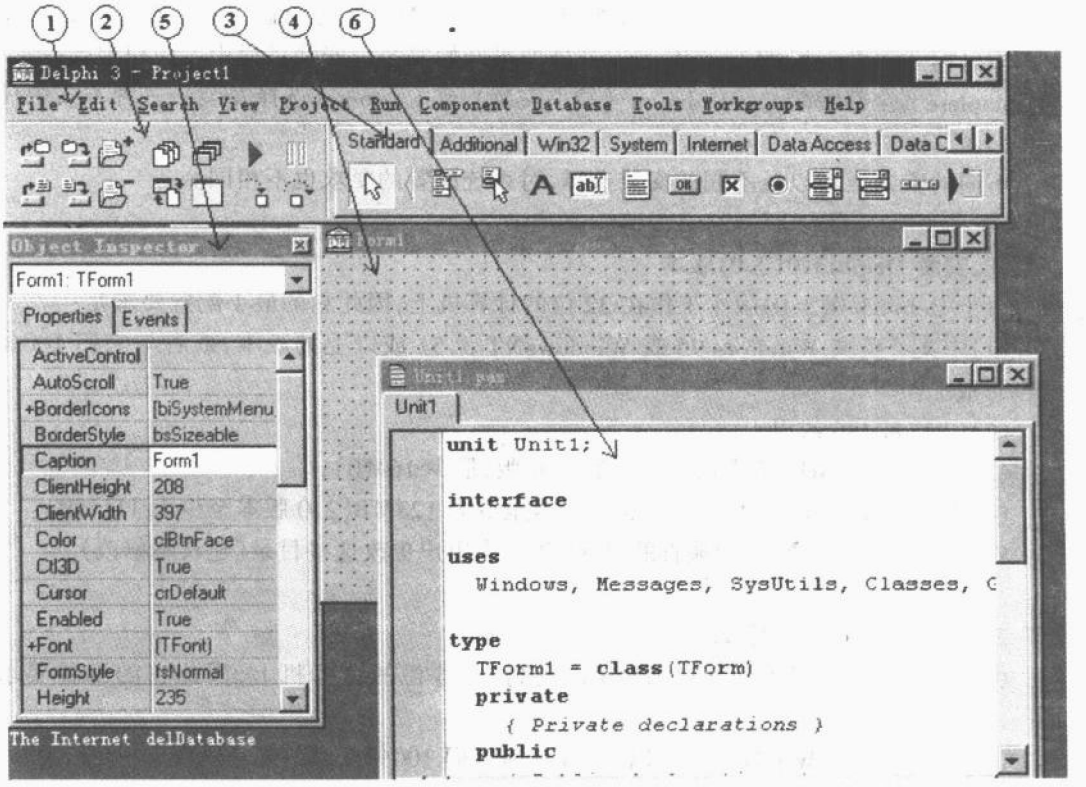

图 2-1　Delphi 3.0 的编程环境

一、主菜单

Delphi 3.0 与 2.0 版本的主菜单项目大致相同,以 3.0 版本为例,主要项目如下(菜单项的具体内容见附录 A):

File(文件):　　包括打开、存储、打印、关闭、删除文件等功能;

Edit(编辑)：	包括剪切、粘贴、复制剪贴板及其它编辑操作；
Search(查询)：	包括字符串查询、取代等操作；
View(查看)：	用于隐含或显示之间的切换；
Project(项目管理)：	在项目中增删文件，编译项目中的文件等；
Run(运行)：	程序的编译、连接、运行以及暂停、停止程序运行等；
Component(组件)：	安装新组件、打开组件库、重新编译组件库等；
Database(数据库)：	打开数据库查看器、SQL 监控器，启动数据库窗体专家等；
Tools(工具)：	确定组件板显示的组件、启动数据库桌面等；
Workgroups(开发组管理)：	对开发小组及档案进行管理；
Help(帮助)：	显示求助的主题。

二、快捷键组

菜单的内容全面，几乎包括了系统全部功能，但由于项目太多，不便使用。为了突出重点，简化和加速操作过程，系统将其中的常用项取出来，以快捷键组的方式集中放在窗体的左上角，以便选用。快捷键以及它们的作用如图 2－2 所示。

Open Project(打开项目文件)

Save All(保存所有文件)

Add file to Project(项目中增加文件)

Select Unit from list(从列表中选取单元文件)

Select Form from list(从列表中选取窗体)

Run(开始运行)

Pause(暂停进行)

Open file(打开文件)

Save file(存储文件)

Remove file from Project(从项目中删除文件)

Toggle Form/Unit(窗体/单元之间切换)

Nem Form(创建新窗体)

Trace into(单步运行，进入调用的过程中)

Step over(单步运行，跳过调用的过程)

图 2－2　快捷键组

三、组件板

组件板是存放各种组件（类）的地方。Delphi 将组件分类放置在不同的页中，组件板的上方为页标志，缺省情况 3．0 版本包括 13 页，它们是：Standard、Additional、Win 32、System、Internet、Data Access、Data Control、Decision Cube、QReport、Dialogs、Win 3．1、Samples、ActiveX。每页包括若干组件，其主要的内容如下（各组件的作用见附录 B）：

Standard 页：	放置 Windows 的标准组件，使用得比较频繁；
Additional 页：	放置附加的组件；
Win 32 页：	放置有关 32 位的 Windows 的组件（相当于 2.0 版本的 Win 95 页）；
System 页：	放置系统组件；
Internet 页：	放置支持国际互连网络组件；
Data Access 页：	放置存取数据库的组件；
Data Controls 页：	放置数据控制组件；
Decision Cube 页：	放置决策支持的组件；
QReport 页：	放置快速打印组件；
Dialogs 页：	放置有关文件及其对话的组件；
Win 3.1 页：	放置有关 Windows 3.1 的组件；
Samples 页：	放置用作范例的组件；
ActiveX 页：	放置 ActiveX 标准的组件（相当于 2.0 版的 OCX，1.0 版的 VBX）。

四、窗体

窗体内可以放置各类组件，是程序运行的主要场所。初打开时，窗体是空的，里面布满了点阵，这是用来对齐放置对象的。当你放置新对象时，对象的左上角会自动对准最邻近的格点。

窗体的其它功能与 Windows 95 中的窗口一样，它的右上角有三个控制键，其作用如图 2－3 所示。

图 2－3　窗体控制键

将鼠标放到窗体标题上按下并拖动，可以移动整个窗体。用鼠标在窗体的边角上按下并拖动，可以改变窗体的大小。

五、对象检阅器

对象检阅器分两页：Properties（属性）页和 Events（事件）页。

Properties 页载明对象的属性，全页分左右两栏，左边是属性名，右边是属性值。若属