

编程  
宝典

# 我的第①本Visual Basic编程书

中国软件行业协会教育与培训委员会 秘书长 邱钦伦  
微软开发工具及平台事业部 产品经理 胡德民  
Sun亚洲全球化中心 技术总监 刘 杰  
NEC信息系统(中国)有限公司 开发部长 石少峰  
《程序员》杂志、CSDN著名技术专家 尹 成  
51CTO.com 技术总监 陈德勇

倾力  
推荐

龙马高新教育 策划  
国家863中部软件孵化器 编著



## 20小时全程同步教学录像 + 18小时Oracle项目实战教学录像

- 20小时全程同步教学录像, 一线教学和开发人员贴心讲解, 配合图书高效学习
- 18小时价值6000元的独家Oracle项目实战教学录像, 帮您轻松学会Oracle数据库; 204节全国计算机等级考试二级VB教学录像, 一线特级命题研究组长亲自授课, 帮助考生轻松过关
- 71个典型范例、7个完整项目, 在实战中掌握VB编程; 70个实战测试及解析, 举一反三, 掌握更透彻
- 8个超值王牌资源大放送, 包括167页VB 6.0函数查询手册、49页VB 6.0控件查询手册、10套超值完整源代码、全国计算机等级考试二级VB考试大纲及应试技巧、160道VB常见试题及解析、50个VB 6.0常见错误及解决方案电子书、50个VB高效编程技巧、VB程序员职业规划等



人民邮电出版社  
POSTS & TELECOM PRESS

从入门到精通  
(第2版)

Visual Basic

# IT翘楚倾力推荐，十万读者的共同选择

架构新颖、合理，内容翔实、全面，讲解通俗、易懂，光盘内容丰富、实用，是一套不可多得的好书。

——中国软件行业协会教育与培训委员会 秘书长 邱钦伦

这套书根据读者的学习习惯，以循序渐进的方式，从最简单的“Hello,World”程序写起，逐步深化、细化，对每个知识点和技术要点都给予了翔实的示例及代码分析，这些示例代码不仅一针见血地指明了技术要点的本质，而且短小精炼，方便复制和试验。

——微软开发工具及平台事业部 产品经理 胡德民

现代计算机软件人才要求的是“丁”字型人才，要求横向掌握多门主流的编程语言，又需要同时至少对1-2门编程技术有深刻的认知。但往往学生的精力有限，很难在短期内完成“丁”字型人才的培养，所以我推荐这套丛书——它汇集了目前主流的编程知识，通过合理的结构和内容设计，让中国的学生通过这些书开始自己的IT软件之梦。

——Sun亚洲全球化中心 技术总监 刘杰

在我招聘程序员的时候，不问学历，一般先上机再面试。我认为实践比知识更重要，懂得如何运用知识比知道知识更重要。希望读者在学透此书的基础上，再在工作中汲取开发、管理经验，这样一定能够步入IT高薪一族或者在IT业中开创自己的一番事业。

——《程序员》杂志、CSDN著名技术专家 尹成

我一直想为广大师生推荐一套符合时代潮流的程序开发类图书。对于教师，它应该具备全面、概念讲解透彻和案例丰富（特别是大型案例）等特性。对于学生，它应该具备零基础入门、可理解性强、可自学性强和可操作性强等特性。这套书做到了。

——郑州大学信息工程学院 副院长 周清雷

现在企业所招聘的都是有项目开发经验的程序员。目前刚毕业的大学生基本都没参与过项目开发，进入企业还需一段培训才能适应，这本书就能让你深入了解企业的项目开发流程。

——NEC信息系统（中国）有限公司 开发部长 石少峥

目前中国的软件教育蓬勃发展，越来越多的人选择培训来提高自己的技能，这样能够快速融入到企业的软件开发中。这本书集结了培训教材结构合理、专业等特点，而且教学录像如同培训老师讲课，学起来更轻松。

——广东拓思软件科学园有限公司 副总经理 黄万民

这套书内容全面、知识结构安排合理，以实例驱动学习，更以项目实战来总结书中所讲内容，易学、易用，对于初学编程的读者来说，是不可多得的好书。

——领先的中文IT技术网站51CTO.com 技术总监 陈德勇

《从入门到精通》系列图书是编程图书中比较专业的辅导书，是读者朋友学好编程语言的工具，也是一把打开软件行业之门的钥匙，是编程类图书中的典范。

——西安863软件孵化器有限公司 总经理 楼文晓

这是一本值得推荐给软件行业和即将步入软件行业的数千万白领用户阅读的好书。透过软件行业可以看到金灿灿的明天，透过《从入门到精通》可以得到行业精通的技能。

——东忠集团 副总裁 李朝阳

写给初学者看的软件开发类书籍最大的挑战就是作者必须同时具有教育背景和技术背景，而该丛书的特色就是由一线软件专家提供优秀的经典案例，再由教育专家深入浅出地详细讲解，同时“源代码+讲解+视频”的立体模型弥补了传统书籍纯文字化的不足。因此，在华大锐志教育集团下的很多学员都将此丛书作为入门必备的自学书籍。

——华大锐志软件人才孵化基地 技术总监（微软MVP）王豫翔

美术编辑：董志桢

分类建议：计算机 / 程序设计

人民邮电出版社网址：www.ptpress.com.cn



ISBN 978-7-115-37720-3



9 787115 377203 >

ISBN 978-7-115-37720-3

定价：69.80 元（附光盘）





# Visual Basic

## 从入门到精通

(第2版)

龙马高新教育 策划  
国家863中部软件孵化器 编著

人民邮电出版社  
北京

## 图书在版编目 (CIP) 数据

Visual Basic从入门到精通 / 国家863中部软件孵化器编著. — 2版. — 北京: 人民邮电出版社, 2015.3  
ISBN 978-7-115-37720-3

I. ①V… II. ①国… III. ①BASIC语言—程序设计  
IV. ①TP312

中国版本图书馆CIP数据核字(2015)第011062号

## 内 容 提 要

本书以零基础讲解为宗旨,用实例引导读者学习,深入浅出地介绍了 Visual Basic 的相关知识和实战技能。

本书第1篇【基础知识】主要讲解 Visual Basic 的基础知识、语言基础、算法和程序控制结构、数组以及内置函数与过程等;第2篇【核心技术】主要讲解可视化编程、窗体和系统对象、标准模块和类模块、标准控件、ActiveX 控件、工具栏和状态栏、鼠标和键盘事件、菜单和对话框设计、程序调试与错误处理等;第3篇【高级应用】主要讲解数据库与 SQL 语言基础、Visual Basic 6.0 中的数据库编程、数据报表、API 编程、网络编程、图形图像与多媒体编程、文件系统编程以及应用程序打包等;第4篇【应用开发】主要讲解项目规划、VB 实现远程控制、仿 Windows 画图程序、播放器、文件分割与合并程序以及 VB 连连看等各种实用程序的开发;第5篇【项目实战】介绍了个人账目管理系统和超市进销存管理系统2个项目的开发流程。

本书所附 DVD 光盘包含了与图书内容全程同步的教学录像。此外,还赠送了大量相关学习资料,以便读者扩展学习。

本书适合任何想学习 Visual Basic 的读者,无论您是否从事计算机相关行业,是否接触过 Visual Basic,均可通过学习快速掌握 Visual Basic 的开发方法和技巧。

- 
- ◆ 策 划 龙马高新教育
  - 编 著 国家 863 中部软件孵化器
  - 责任编辑 张 翼
  - 责任印制 杨林杰
  - ◆ 人民邮电出版社出版发行 北京市丰台区成寿寺路 11 号
  - 邮编 100164 电子邮件 315@ptpress.com.cn
  - 网址 <http://www.ptpress.com.cn>
  - 北京圣夫亚美印刷有限公司印刷
  - ◆ 开本: 787×1092 1/16
  - 印张: 40
  - 字数: 1 112 千字 2015 年 3 月第 2 版
  - 印数: 12 501—16 500 册 2015 年 3 月北京第 1 次印刷
- 

定价: 69.80 元 (附光盘)

读者服务热线: (010) 81055410 印装质量热线: (010) 81055316

反盗版热线: (010) 81055315

广告经营许可证: 京崇工商广字第 0021 号



# 前言

“从入门到精通”系列是专为初学者量身打造的一套编程学习用书，由知名计算机图书策划机构“龙马高新教育”精心策划而成。

本书主要面向 Visual Basic 初学者和爱好者，旨在帮助读者掌握 Visual Basic 基础知识、了解开发技巧并积累一定的项目实战经验。当读者系统地学习完本书内容之后，就可以骄傲地宣布——“我是一名真正的 Visual Basic 程序员了！”。

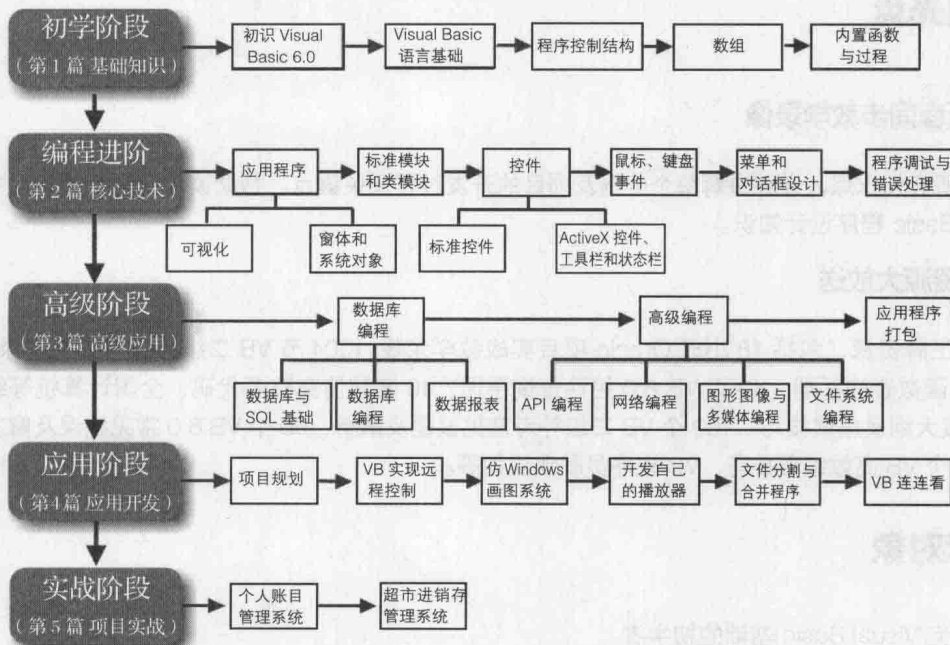
## 为什么要写这样一本书

荀子曰：不闻不若闻之，闻之不若见之，见之不若知之，知之不若行之。

实践对于学习的重要性由此可见一斑。纵观当前编程图书市场，理论知识与实践经验的脱节，是很多 Visual Basic 图书的写照。为了杜绝这一现象，本书立足于实战，从项目开发的实际需求入手，将理论知识与实际应用相结合。目标就是让初学者能够快速成长为初级程序员，并拥有一定的项目开发经验，从而在职场中拥有一个高起点。

## Visual Basic 的最佳学习路线

本书总结了作者多年的教学实践经验，为读者设计了最佳的学习路线。



# 目 录

## 第0章 Visual Basic 学习指南 ..... 1

- 0.1 Visual Basic 的来源 ..... 2
- 0.2 Visual Basic 的特点 ..... 2
- 0.3 Visual Basic 无处不在 ..... 3
- 0.4 Visual Basic 学习经验 ..... 4
- 0.5 Visual Basic 的学习路线 ..... 4

## 第1篇 基础知识

开启 Visual Basic 6.0 之门。

## 第1章 步入 VB 开发之门——初识 Visual Basic 6.0 ..... 6



本章视频教学录像: 48 分钟

本章将带领你步入 Visual Basic 6.0 的世界, 并教会你如何用自己的双手开启知识之门——创建第 1 个 VB 应用程序。

- 1.1 Visual Basic 简介 ..... 7
  - 1.1.1 Visual Basic 的发展 ..... 7
  - 1.1.2 Visual Basic 6.0 的功能特点 ..... 7
- 1.2 Visual Basic 6.0 的安装与启动 ..... 8
  - 1.2.1 Visual Basic 6.0 的安装 ..... 8
  - 1.2.2 Visual Basic 6.0 开发环境的定制 ..... 9
  - 1.2.3 启动与退出 ..... 12
- 1.3 Visual Basic 6.0 的集成开发环境 ..... 12
  - 1.3.1 认识 Visual Basic 6.0 的工作界面 ..... 12
  - 1.3.2 主窗口 ..... 14
  - 1.3.3 窗体设计 / 代码设计窗口 ..... 16
  - 1.3.4 属性窗口 ..... 17
  - 1.3.5 工程资源管理器窗口 ..... 17
  - 1.3.6 工具箱窗口 ..... 18
  - 1.3.7 其他窗口 ..... 18
  - 1.3.8 Visual Basic 帮助系统的使用 ..... 19



1.4 用 Visual Basic 6.0 管理工程 .....	20
1.4.1 工程介绍 .....	20
1.4.2 新建、保存工程 .....	21
1.4.3 向工程中添加窗体和模块 .....	22
1.4.4 运行和关闭工程 .....	23
1.4.5 删除工程 .....	23
1.4.6 生成可执行文件 .....	24
1.5 来自 VB 世界的第一声问候——第 1 个应用程序 .....	24
1.5.1 VB 程序设计的一般步骤 .....	25
1.5.2 创建应用程序的界面 .....	25
1.5.3 设置控件属性 .....	25
1.5.4 编写代码 .....	26
1.5.5 调试、运行程序 .....	26
1.6 实战练习 .....	28

## 第2章 Visual Basic 的入门钥匙——Visual Basic 语言基础 .....29



本章视频教学录像: 1 小时 5 分钟

任何一种编程语言, 基础知识都必不可少, 本章是 VB 开发中最基础的知识。

2.1 标识符和数据类型 .....	30
2.1.1 标识符 .....	30
2.1.2 数据类型 .....	31
2.2 常量和变量 .....	36
2.2.1 常量 .....	36
2.2.2 变量 .....	38
2.3 运算符 .....	41
2.3.1 算术运算符 .....	41
2.3.2 赋值运算符 .....	42
2.3.3 关系运算符 .....	42
2.3.4 逻辑运算符 .....	42
2.3.5 连接运算符 .....	43
2.3.6 特殊运算符 .....	43
2.3.7 运算符的优先级 .....	44
2.4 表达式 .....	45
2.4.1 算术表达式 .....	45
2.4.2 字符串表达式 .....	45
2.4.3 日期表达式 .....	45

2.5 代码编写规范 .....	45
2.5.1 Visual Basic 6.0 标识符的定义规则 .....	46
2.5.2 Visual Basic 6.0 中变量及控件的命名规则 .....	46
2.5.3 程序书写规则 .....	46
2.5.4 添加注释 .....	47
2.5.5 格式化缩排程序语句 .....	47
2.6 高手点拨 .....	48
2.7 实战练习 .....	48

## 第3章 Visual Basic 的秘密——算法和程序控制结构 ..... 49



本章视频教学录像: 1 小时 2 分钟

算法和程序控制结构是编程语言中最基础也是最重要的内容。

3.1 算法 .....	50
3.2 结构化程序设计 .....	50
3.3 顺序结构 .....	52
3.3.1 赋值运算符 .....	53
3.3.2 数据的输入与输出 .....	53
3.4 选择结构 .....	55
3.4.1 If 条件语句 .....	55
3.4.2 Select case 语句 .....	56
3.4.3 条件函数 .....	58
3.5 循环结构 .....	60
3.5.1 For 循环语句 .....	60
3.5.2 Do-Loop 循环语句 .....	61
3.5.3 循环的嵌套 .....	62
3.6 其他辅助控制语句 .....	64
3.6.1 End 结束语句 .....	64
3.6.2 Exit 退出语句 .....	64
3.6.3 GoTo 跳转语句 .....	65
3.6.4 On Error 语句 .....	65
3.6.5 复用语句 With-End With .....	66
3.7 高手点拨 .....	67
3.8 实战练习 .....	68



**第4章 同类型批量数据管理的技巧——数组 .....69**

本章视频教学录像: 1 小时 2 分钟

$N$  个数字有序地放在 1 组就是数组, 就好比将零散的物品放置在同一个箱子中, 移动这些物品只需要搬动箱子即可。

4.1 数组的概念 .....	70
4.1.1 定长数组及声明 .....	70
4.1.2 动态数组及声明 .....	71
4.2 数组基本操作 .....	71
4.2.1 数组的引用 .....	71
4.2.2 数组的初始化 .....	72
4.2.3 数组元素的输入、输出 .....	72
4.2.4 数组元素的插入、删除和查找 .....	73
4.2.5 数组元素的应用及排序 .....	75
4.3 数组相关函数及语句 .....	76
4.3.1 Array 函数 .....	76
4.3.2 UBound 函数和 LBound 函数 .....	77
4.3.3 Split 函数 .....	77
4.3.4 Option Base 语句 .....	78
4.4 高手点拨 .....	80
4.5 实战练习 .....	80

**第5章 应用程序提升的法宝——内置函数与过程 .....81**

本章视频教学录像: 1 小时 13 分钟

本章将为你揭秘 VB 代码是如何高效运行的。

5.1 秘密武器——常用的内置函数 .....	82
5.1.1 数学函数 .....	82
5.1.2 字符串函数 .....	83
5.1.3 转换函数 .....	84
5.1.4 日期时间函数 .....	87
5.1.5 随机函数 .....	90
5.1.6 判断函数 .....	91
5.1.7 格式化函数 .....	92
5.1.8 Shell 函数 .....	94
5.2 提升法宝——过程 .....	94
5.2.1 事件过程 .....	94

5.2.2 Sub 过程（子过程） .....	95
5.2.3 Function 过程（函数过程） .....	97
5.2.4 参数的传递 .....	100
5.2.5 过程的嵌套与递归 .....	102
5.3 高手点拨 .....	106
5.4 实战练习 .....	106

## 第 2 篇 核心技术

掌握了基础知识，你已经跨进了 Visual Basic 的门槛。本篇将带领你更上一层楼，去探索 Visual Basic 的核心世界。

### 第 6 章 应用程序的精髓——可视化编程..... 108



本章视频教学录像：30 分钟

本章将为你介绍可视化编程的相关技术。

6.1 对象概念 .....	109
6.1.1 对象和类 .....	109
6.1.2 VB 中对象的建立和编辑 .....	109
6.2 对象的属性、方法和事件 .....	110
6.2.1 对象的属性及设置 .....	110
6.2.2 对象的方法及调用 .....	112
6.2.3 对象的事件及事件过程 .....	113
6.3 高手点拨 .....	113
6.4 实战练习 .....	114

### 第 7 章 应用程序的脸——窗体和系统对象..... 115



本章视频教学录像：1 小时 5 分钟

应用程序的界面好比是人的脸，本章教你如何给应用程序“化妆”。

7.1 窗体简介 .....	116
7.1.1 窗体的基本概念 .....	116
7.1.2 在工程中添加窗体的方法 .....	116
7.2 控制窗体表情——窗体的属性、方法和事件 .....	117
7.2.1 窗体的属性 .....	117
7.2.2 窗体的方法 .....	121



7.2.3 窗体的事件 .....	122
7.3 窗体的生命周期 .....	123
7.3.1 选择启动窗体 .....	123
7.3.2 快速显示窗体 .....	124
7.3.3 结束窗体 .....	124
7.4 多窗体设计 .....	125
7.4.1 创建多窗体应用程序 .....	125
7.4.2 多窗体特性 .....	126
7.5 登录窗体设计实例 .....	126
7.6 系统对象 .....	130
7.6.1 应用程序 APP 对象 .....	130
7.6.2 屏幕 Screen 对象 .....	131
7.6.3 剪贴片 Clipboard 对象 .....	132
7.6.4 调试 Debug 对象 .....	133
7.7 高手点拨 .....	133
7.8 实战练习 .....	134

## 第8章 标准模块和类模块 ..... 135



本章视频教学录像: 25 分钟

标准模块是在应用程序庞大复杂时创建的一个独立模块, 用它实现代码公用; 而类模块可以包含共享代码与数据。

8.1 标准模块 .....	136
8.1.1 标准模块概述 .....	136
8.1.2 添加标准模块 .....	136
8.2 类模块 .....	137
8.2.1 类模块概述 .....	138
8.2.2 添加类模块 .....	138
8.3 标准模块和类模块的区别 .....	139
8.4 高手点拨 .....	141
8.5 实战练习 .....	142

## 第9章 VB 的简易之道——标准控件 ..... 143



本章视频教学录像: 2 小时 14 分钟

Visual Basic 6.0 之所以易学易掌握, 是因为有控件, 它非常简单易用。

9.1 控件概述 .....	144
----------------	-----

9.2 标签控件 .....	144
9.2.1 标签控件的主要属性 .....	144
9.2.2 标签控件 (Label) 的主要事件 .....	145
9.2.3 标签控件应用示例 .....	146
9.3 文本框控件 .....	148
9.3.1 文本框的主要属性 .....	148
9.3.2 文本框控件常用的事件 .....	149
9.3.3 文本框控件应用示例 .....	150
9.4 命令按钮控件 .....	151
9.4.1 命令按钮控件的主要属性 .....	152
9.4.2 命令按钮控件的事件 .....	152
9.4.3 命令按钮控件应用示例 .....	153
9.5 单选按钮控件 .....	154
9.5.1 单选按钮的主要属性 .....	154
9.5.2 单选按钮的常用事件 .....	155
9.5.3 单选按钮控件应用示例 .....	155
9.6 复选框控件 .....	157
9.6.1 复选框的主要属性 .....	157
9.6.2 复选框的常用事件 .....	157
9.6.3 复选框控件应用示例 .....	158
9.7 框架控件 .....	161
9.7.1 框架的主要属性 .....	161
9.7.2 框架控件应用示例 .....	161
9.8 列表框控件 .....	164
9.8.1 列表框的主要属性 .....	164
9.8.2 列表框的主要事件 .....	165
9.8.3 列表框控件的方法 .....	166
9.8.4 列表框控件应用示例 .....	166
9.9 组合框控件 .....	172
9.9.1 组合框控件的主要属性 .....	172
9.9.2 组合框的事件和方法 .....	173
9.9.3 组合框应用示例 .....	173
9.10 图像框控件 .....	177
9.10.1 图像框控件的主要属性 .....	177
9.10.2 图像框控件的主要事件和方法 .....	177
9.10.3 图像框应用示例 .....	178
9.11 滚动条控件 .....	184
9.11.1 滚动条控件的主要属性 .....	184
9.11.2 滚动条控件的主要事件 .....	185
9.11.3 滚动条应用示例 .....	185

9.12 程序中的闹钟——定时器控件.....	188
9.12.1 定时器控件的主要属性.....	188
9.12.2 定时器控件的主要事件.....	188
9.12.3 定时器控件应用示例.....	188
9.13 文件系统控件.....	192
9.13.1 驱动器列表框控件.....	192
9.13.2 目录列表框控件.....	193
9.13.3 文件列表框控件.....	193
9.13.4 文件系统应用示例.....	193
9.14 控件数组.....	195
9.14.1 控件数组的概念.....	195
9.14.2 控件数组的创建.....	196
9.14.3 控件数组的使用.....	197
9.15 高手点拨.....	199
9.16 实战练习.....	200

## 第10章 扩展你的需求——ActiveX 控件、工具栏和状态栏 ..... 201



本章视频教学录像: 44 分钟

扩展控件, 顾名思义, 能更好地满足编程需求。

10.1 ActiveX 控件的使用.....	202
10.1.1 ActiveX 控件的添加.....	202
10.1.2 ActiveX 控件的删除.....	202
10.1.3 ActiveX 控件的注册.....	202
10.2 图像列表控件.....	204
10.2.1 向图像列表控件添加图片.....	204
10.2.2 图像列表控件与其他控件关联.....	205
10.2.3 图像列表控件的应用实例.....	205
10.3 树状视图控件——统筹全局的好工具.....	208
10.3.1 树状视图控件的主要属性、事件和方法.....	209
10.3.2 树状视图控件的应用实例.....	210
10.4 选项卡控件.....	212
10.4.1 选项卡控件的主要属性.....	212
10.4.2 选项卡控件的应用实例.....	213
10.5 进度条控件.....	216
10.5.1 进度条控件的主要属性和方法.....	216
10.5.2 进度条控件的应用实例.....	217

10.6 视图控件 (ListView) .....	219
10.6.1 ListView 控件简介 .....	220
10.6.2 添加数据 .....	220
10.6.3 创建报表视图 .....	221
10.6.4 创建大图标视图 .....	221
10.7 日期 / 时间控件 (DateTimePicker) .....	221
10.7.1 认识 DateTimePicker 控件 .....	221
10.7.2 设置和返回日期 .....	222
10.7.3 实时读取 DTPicker 控件中的日期 .....	223
10.7.4 使用 CheckBox 属性来选择无日期 .....	223
10.7.5 使用日期和时间的格式 .....	223
10.7.6 使用 DTPicker 控件计算日期或天数 .....	225
10.8 工具栏控件 .....	225
10.8.1 工具栏控件的主要属性和事件 .....	225
10.8.2 工具栏控件的应用实例 .....	226
10.9 状态栏控件 .....	231
10.9.1 状态栏控件的属性 .....	231
10.9.2 状态栏控件的方法 .....	232
10.9.3 状态栏控件的事件 .....	232
10.10 高手点拨 .....	233
10.11 实战练习 .....	234

## 第 11 章 鼠标、键盘的另类编程应用—— 鼠标、键盘事件 .....235



本章视频教学录像: 21 分钟

轻轻一点, 结果就变。

11.1 鼠标事件 .....	236
11.1.1 “鼠标按键按下”事件 (MouseDown) .....	236
11.1.2 “鼠标按键释放”事件 (MouseUp) .....	239
11.1.3 “移动鼠标”事件 (MouseMove) .....	241
11.2 键盘事件 .....	242
11.2.1 “键盘按键”事件 (KeyPress) .....	242
11.2.2 “键盘按下”事件 (KeyDown) .....	244
11.2.3 “键盘弹起”事件 (KeyUp) .....	245
11.3 高手点拨 .....	246
11.4 实战练习 .....	246

**第12章 程序与用户的交互——菜单和对话框设计 .....247**

本章视频教学录像: 43 分钟

你的问题我回答, 沟通无限常对话。

12.1 魅力化妆师——菜单设计 .....	248
12.1.1 菜单编辑器 .....	249
12.1.2 下拉式菜单设计 .....	252
12.1.3 弹出式菜单设计 .....	254
12.1.4 自定义菜单设计 .....	258
12.2 模式对话框和无模式对话框 .....	261
12.3 预定义对话框设计 .....	262
12.3.1 输入对话框设计 .....	262
12.3.2 消息对话框设计 .....	264
12.4 通用对话框设计 .....	268
12.4.1 添加通用对话框控件 .....	268
12.4.2 通用对话框设计实例 .....	269
12.5 高手点拨 .....	272
12.6 实战练习 .....	272

**第13章 编程错误终结者——程序调试与错误处理 .....273**

本章视频教学录像: 32 分钟

程序调试——让“臭虫”无处遁形。

13.1 Visual Basic 6.0 程序中的错误类型 .....	274
13.1.1 语法错误 .....	274
13.1.2 逻辑错误 .....	274
13.1.3 运行时错误 .....	274
13.2 程序工作状态 .....	275
13.2.1 设计状态 .....	275
13.2.2 运行状态 .....	275
13.2.3 中断状态 .....	275
13.3 程序调试 .....	276
13.3.1 使程序进入中断状态 .....	276
13.3.2 调试工具 .....	277
13.3.3 调试方法 .....	281
13.4 除虫行动——Visual Basic 6.0 中的错误处理 .....	283



13.4.1 Err 对象 .....	283
13.4.2 On Error GoTo 语句 .....	284
13.4.3 Resume 语句 .....	284
13.4.4 错误处理实例 .....	284
13.5 高手点拨 .....	288
13.6 实战练习 .....	288

## 第 3 篇 高级应用

基础知识只能完成简单编程，只有掌握了高级应用，才能真正获得开发大型系统的能力。

### 第 14 章 进入数据仓库——数据库与 SQL 语言基础.....290



本章视频教学录像: 40 分钟

数据库使应用程序不再单调，你可以将数据随心所欲地展示出来，只需通过 SQL 就可以办到。

14.1 数据库基本概念 .....	291
14.2 SQL 应用 .....	292
14.2.1 SQL 语言的特点 .....	292
14.2.2 常用 SQL 语句简介 .....	292
14.3 Select 语句的使用——数据库的灵魂 .....	292
14.3.1 Select 子语句 .....	293
14.3.2 From 子语句 .....	295
14.3.3 As 子语句 .....	296
14.3.4 Where 子语句 .....	296
14.3.5 Order By 子语句 .....	299
14.3.6 Group By 子语句 .....	300
14.4 SQL 中的常用函数 .....	300
14.4.1 算术函数 .....	300
14.4.2 统计函数 .....	301
14.5 利用 SQL 语言修改表数据 .....	302
14.5.1 Insert 语句 .....	302
14.5.2 Update 语句 .....	303
14.5.3 Delete 语句 .....	304
14.6 高手点拨 .....	305
14.7 实战练习 .....	306

## 第15章 Visual Basic 与数据库的联合——Visual Basic 6.0 中的数据库编程 ..... 307



本章视频教学录像: 59 分钟

本章介绍如何将 Visual Basic 6.0 与数据库的联合应用。

15.1 英雄相惜——Visual Basic 6.0 与数据库.....	308
15.1.1 Visual Basic 支持的常用数据库.....	308
15.1.2 Visual Basic 中的数据库控件.....	308
15.2 数据库的建立、维护和查询.....	309
15.2.1 建立数据库.....	310
15.2.2 删除数据库中的表.....	314
15.2.3 修改数据表结构和数据.....	315
15.2.4 数据查询.....	317
15.2.5 数据窗体设计器.....	318
15.3 使用 Data 控件访问数据库.....	320
15.3.1 Data 控件的常用属性.....	320
15.3.2 Data 控件的常用方法.....	322
15.3.3 Data 控件的常用事件.....	323
15.3.4 Data 控件访问数据库实例.....	324
15.4 使用 ADO 控件访问数据库.....	329
15.4.1 添加 ADO 控件.....	329
15.4.2 ADO 控件的常用属性.....	330
15.4.3 ADO 控件的常用方法.....	331
15.4.4 ADO 控件的常用事件.....	331
15.4.5 ADO 控件访问数据库实例.....	332
15.5 高手点拨.....	336
15.6 实战练习.....	336

## 第16章 Visual Basic 6.0 生成的报表——数据报表 ..... 337



本章视频教学录像: 17 分钟

报表不再只是 Excel 的专利。

16.1 数据报表简介.....	338
16.2 数据报表的生成环境.....	338
16.3 数据报表的生成.....	344
16.4 高手点拨.....	347
16.5 实战练习.....	347

## 第 17 章 Visual Basic 编程的核心——API 编程.....349



本章视频教学录像: 33 分钟

本章将引领你步入 Visual Basic 编程的核心境界。

17.1 API 概述.....	350
17.1.1 API 基本数据类型.....	350
17.1.2 API 常见数据结构.....	351
17.1.3 API 浏览器.....	351
17.2 API 的函数分类.....	353
17.2.1 窗口管理类函数.....	353
17.2.2 图形设备接口类函数.....	354
17.2.3 系统服务类函数.....	354
17.2.4 国际特性类函数.....	355
17.2.5 网络服务函数.....	356
17.3 API 函数的应用.....	356
17.3.1 使用 Declare 语句手动声明 API 函数.....	357
17.3.2 使用 API 浏览器声明 API 函数.....	358
17.3.3 API 函数的调用.....	359
17.4 插上翅膀去飞翔——API 编程实例.....	359
17.5 高手点拨.....	361
17.6 实战练习.....	362

## 第 18 章 Visual Basic 中的网络世界——网络编程.....363



本章视频教学录像: 31 分钟

Visual Basic 中的网络世界同样精彩。

18.1 邮件应用编程.....	364
18.1.1 邮件程序接口控件的属性和方法.....	364
18.1.2 实现邮件发送.....	370
18.2 互联网传输应用编程.....	375
18.2.1 互联网传输控件的属性、事件和方法.....	375
18.2.2 实现互联网文件上传.....	380
18.3 网页浏览器应用编程.....	388
18.3.1 网页浏览器控件的属性、事件和方法.....	388
18.3.2 实现自定义网页浏览器应用.....	390
18.4 高手点拨.....	392
18.5 实战练习.....	392

**第19章 Visual Basic 中的视听——图形图像与多媒体编程.....383**

本章视频教学录像: 29 分钟

Visual Basic 可以让程序有声有色。通过本章的学习, 你可以在 VB 中画图、播放多媒体甚至制作动画。

19.1 图形应用编程 .....	394
19.1.1 坐标系 .....	394
19.1.2 颜色设置 .....	400
19.1.3 绘图方法 .....	402
19.2 多媒体应用编程 .....	408
19.2.1 多媒体控制接口控件基本概念 .....	408
19.2.2 多媒体控制接口控件的属性 .....	410
19.2.3 多媒体控制接口控件的事件 .....	411
19.2.4 多媒体控制接口控件应用实例 .....	413
19.3 让程序动起来——动画应用编程 .....	415
19.3.1 添加动画控件 .....	415
19.3.2 动画控件的属性 .....	416
19.3.3 动画控件的方法 .....	417
19.3.4 动画控件应用实例 .....	418
19.4 高手点拨 .....	420
19.5 实战练习 .....	420

**第20章 用 VB 操纵文件——文件系统编程 .....421**

本章视频教学录像: 28 分钟

Visual Basic 也可以当管家。学习了本章以后, 你也可以使用 Visual Basic 来操作文件。

20.1 文件的类型与结构 .....	422
20.1.1 文件结构 .....	422
20.1.2 文件类型 .....	422
20.2 文件操作语句 .....	422
20.3 操纵文件的魔法——文件操作函数 .....	426
20.4 顺序文件 .....	430
20.4.1 顺序文件的打开 .....	430
20.4.2 顺序文件的读取 .....	431
20.4.3 顺序文件的写入 .....	431
20.4.4 顺序文件的关闭 .....	432
20.4.5 顺序文件使用实例 .....	432
20.5 随机文件 .....	436

20.5.1 随机文件的打开 .....	436
20.5.2 随机文件的读取 .....	436
20.5.3 随机文件的写入 .....	437
20.5.4 随机文件的关闭 .....	437
20.5.5 随机文件使用实例 .....	437
20.6 二进制文件 .....	441
20.6.1 二进制文件的打开 .....	442
20.6.2 二进制文件的读取 .....	442
20.6.3 二进制文件的写入 .....	442
20.6.4 二进制文件的关闭 .....	442
20.6.5 二进制文件使用实例 .....	442
20.7 高手点拨 .....	443
20.8 实战练习 .....	444

## 第 21 章 让你的程序去旅行——应用程序打包 .....445



视频教学录像: 10 分钟

将你的应用程序打包, 别人不用安装 Visual Basic 6.0 也可以运行你的程序。

21.1 打包前的准备 .....	446
21.2 打包应用程序 .....	446
21.3 安装应用程序 .....	451
21.4 卸载应用程序 .....	452
21.5 打包应注意的问题 .....	453
21.6 高手点拨 .....	454
21.7 实战练习 .....	454

## 第 4 篇 应用开发

应用才是最终的目的。本篇将带领你开发一些实用的 Visual Basic 应用程序, 让你在应用中取得“真经”。

## 第 22 章 项目实战前的忠告——项目规划 .....456



本章视频教学录像: 28 分钟

在项目开发之前, 你知道要做什么吗?

22.1 项目开发流程 .....	457
-------------------	-----



22.1.1 项目策划阶段.....	458
22.1.2 需求分析阶段.....	458
22.1.3 项目开发阶段.....	458
22.1.4 项目测试阶段.....	459
22.1.5 项目后期维护.....	459
22.2 满足客户需求.....	459
22.3 项目开发团队.....	460
22.3.1 项目团队组成.....	460
22.3.2 项目团队特征.....	461
22.4 项目计划说明书.....	461
22.5 项目开发阶段的运作.....	462
22.5.1 初始阶段.....	462
22.5.2 细化阶段.....	463
22.5.3 构建阶段.....	463
22.5.4 交付阶段.....	463
22.5.5 维护阶段.....	463
22.6 高手点拨.....	463

## 第 23 章 网络通信应用开发——VB 实现远程控制 .....465



本章视频教学录像: 59 分钟

Visual Basic 在网络中的功能也是非常强大的。

23.1 系统分析.....	466
23.2 系统设计.....	467
23.3 运行系统.....	483
23.4 开发过程常见问题及解决方法.....	486
23.5 高手点拨.....	486

## 第 24 章 图形图像应用开发——仿 Windows 画图程序 .....487



本章视频教学录像: 8 分钟

用 Visual Basic 来制作简化版的画图程序。

24.1 系统分析.....	488
24.2 系统设计.....	488

24.3 运行系统 .....	503
24.4 高手点拨 .....	504

## 第 25 章 多媒体应用开发——开发自己的播放器 .....505



本章视频教学录像: 10 分钟

我自己的 MP3, 想怎么播就怎么播。

25.1 系统分析 .....	506
25.2 系统设计 .....	506
25.3 运行系统 .....	514
25.4 开发过程常见问题及解决方法 .....	515
25.5 高手点拨 .....	516

## 第 26 章 文件系统应用开发——文件分割与合并程序 .....517



本章视频教学录像: 9 分钟

文件太大被邮箱拒之门外了, 就用 Visual Basic 来分割与合并文件吧!

26.1 系统分析 .....	518
26.2 系统设计 .....	518
26.3 运行系统 .....	524
26.4 开发过程常见问题及解决方法 .....	529
26.5 高手点拨 .....	530

## 第 27 章 游戏开发——VB 连连看 .....531



本章视频教学录像: 8 分钟

超级流行的宠物连连看就是这样制作而成的。

27.1 系统分析 .....	532
27.2 系统设计与开发 .....	532
27.3 运行系统 .....	543
27.4 高手点拨 .....	546

## 第5篇 项目实战

本篇将带你步入真正的程序员行列,让你在实战中提高 Visual Basic 项目的开发能力,拥有“一览众山小”的感觉。

### 第28章 数据库应用开发——个人账目管理系统 .....548



本章视频教学录像: 17 分钟

管理自己的账目,用 Visual Basic 就可以。

28.1 系统分析 .....	549
28.1.1 系统需求分析 .....	549
28.1.2 系统功能模块设计 .....	549
28.2 数据库分析和设计 .....	550
28.2.1 数据库分析 .....	550
28.2.2 创建数据库 .....	550
28.2.3 创建表 .....	551
28.3 系统界面设计 .....	553
28.3.1 创建工程和数据库连接模块 .....	553
28.3.2 添加控件 .....	554
28.3.3 系统主界面设计 .....	555
28.3.4 系统功能实现的各界面设计 .....	557
28.4 系统代码设计 .....	560
28.4.1 主窗体代码设计 .....	560
28.4.2 【日常收入】窗体代码设计 .....	561
28.4.3 【日常支出】窗体代码设计 .....	563
28.4.4 【借入款项】窗体代码设计 .....	563
28.4.5 【借出款项】窗体代码设计 .....	564
28.4.6 【月度统计】窗体代码设计 .....	565
28.5 运行系统 .....	567
28.5.1 系统主界面操作 .....	567
28.5.2 项目管理操作 .....	567
28.5.3 日常收入、支出管理操作 .....	568
28.5.4 借入款项、借出款项管理操作 .....	568
28.5.5 月度统计管理操作 .....	569
28.6 高手点拨 .....	569

**第 29 章 打造你的小型超市——超市进销存管理系统 ..... 571**

本章视频教学录像: 8 分钟

用 Visual Basic 6.0 打造你的小型超市进销存管理系统。

29.1 需求及功能分析 .....	572
29.2 数据库设计 .....	572
29.2.1 创建数据库 .....	572
29.2.2 创建表 .....	572
29.3 系统界面设计 .....	575
29.3.1 【综合管理】窗体设计 .....	575
29.3.2 【员工管理】选项卡设计 .....	578
29.3.3 【供应管理】选项卡设计 .....	581
29.3.4 【客户管理】选项卡设计 .....	582
29.3.5 【货物分类】选项卡设计 .....	583
29.3.6 【货物管理】选项卡设计 .....	583
29.3.7 【进货记录操作】窗体设计 .....	584
29.3.8 【出货记录操作】窗体设计 .....	585
29.3.9 【VB 小型超市管理】主窗体设计 .....	587
29.4 系统代码编写 .....	590
29.4.1 添加【VB 小型超市管理】窗体代码 .....	590
29.4.2 添加【综合管理】窗体代码 .....	591
29.4.3 添加【进货记录操作】窗体代码 .....	602
29.4.4 添加【出货记录操作】窗体代码 .....	606
29.5 系统运行 .....	610
29.5.1 员工管理 .....	610
29.5.2 供应管理 .....	611
29.5.3 客户管理 .....	611
29.5.4 货物分类管理 .....	612
29.5.5 货物管理 .....	612
29.5.6 进货记录管理 .....	612
29.5.7 出货记录管理 .....	613
29.5.8 显示【关于】对话框 .....	613
29.6 高手点拨 .....	614



## 赠送资源 (光盘中)

▶ 1. 18 小时 Oracle 项目实战教学录像

▶ 2. 204 节 VB 二级等级考试教学录像

▶ 3. 167 页 VB 6.0 函数查询手册

▶ 4. 49 页 VB 6.0 控件查询手册

▶ 5. 10 套超值完整源代码

▶ 6. 全国计算机等级考试二级 VB 考试大纲及应试技巧

▶ 7. 160 个 VB 二级等考常见试题及解析

▶ 8. 50 个 VB 6.0 常见错误及解决方案电子书

▶ 9. 50 个 VB 高效编程技巧

▶ 10. VB 程序员职业规划

51. 18 小时 Oracle 项目实战教学录像  
52. 204 节 VB 二级等级考试教学录像  
53. 167 页 VB 6.0 函数查询手册  
54. 49 页 VB 6.0 控件查询手册  
55. 10 套超值完整源代码  
56. 全国计算机等级考试二级 VB 考试大纲及应试技巧  
57. 160 个 VB 二级等考常见试题及解析  
58. 50 个 VB 6.0 常见错误及解决方案电子书  
59. 50 个 VB 高效编程技巧  
60. VB 程序员职业规划



# 第0章



本章视频教学录像：10 分钟

## Visual Basic 学习指南

在学习 Visual Basic 6.0 之前应该对该语言的发展史以及发展方向加以了解，对其语言的优缺点也要有一定的认识，再系统地学习该语言。只有这样，才能更好地学习这门语言。

### 本章要点（已掌握的在方框中打钩）

- ☐ Visual Basic 的来源
- ☐ Visual Basic 的特点
- ☐ Visual Basic 的应用领域
- ☐ Visual Basic 的学习路线

## 0.1 Visual Basic 的来源

Visual Basic（简称 VB）是一种由微软公司开发的包含协助开发环境的事件驱动编程语言。从任何标准来说，VB 都是世界上使用人数最多的语言之一——无论是盛赞 VB 的开发者还是抱怨 VB 的开发者。它源自于 BASIC 编程语言。VB 拥有图形用户界面（GUI）和快速应用程序开发（RAD）系统，可以轻易地使用 DAO、RDO、ADO 连接数据库，或者轻松地创建 ActiveX 控件。程序员可以轻松地使用 VB 提供的组件快速建立一个应用程序。

1991 年，微软公司推出了 Visual Basic 1.0，当时引起了很大的轰动。这个连接编程语言和用户界面的进步被称为 Tripod（有些时候叫作 Ruby），最初的设计是由阿兰·库珀（Alan Cooper）完成的。许多专家把 VB 的出现当作是软件开发史上的一个具有划时代意义的事件。在当时，它是第一个“可视”的编程软件。这使得程序员欣喜之极，都尝试在 VB 的平台上进行软件创作。微软也不失时机地在 4 年内接连推出 2.0、3.0、4.0 三个版本。并且从 VB 3.0 开始，微软将 Access 的数据库驱动集成到了 VB 中，这使得 VB 的数据库编程能力大大提高。从 VB 4.0 开始，VB 也引入了面向对象的程序设计思想。VB 功能强大，学习简单。而且，VB 还引入了“控件”的概念，使得大量已经编好的 VB 程序可以被我们直接拿来使用。

2002 年开始，微软将 .NET Framework 与 Visual Basic 结合而成为 Visual Basic .NET (VB .NET)，重新打造 VB，新增许多特性及语法，又将 VB 推向一个新的高度。最新版本 Visual Basic 2012 也将带来许多令人期待的新功能。

通过几年的发展，它已成为一种专业化的开发语言和环境。用户可用 Visual Basic 快速创建 Windows 程序，并可编写企业水平的客户端 / 服务器程序及强大的数据库应用程序。Visual Basic 的发展简史如下表所示。

发布日期	名称	说明
1990-04	Visual Basic 1.0 Windows 版本	
1992-09	Visual Basic 1.0 DOS 版本	
1992-11	Visual Basic 2.0	对于上一个版本的界面和速度都有所改善
1993-06	Visual Basic 3.0	包含一个数据引擎，可以直接读取 Access 数据库
1995-08	Visual Basic 4.0	发布了 32 位和 16 位的版本，其中包含了对类的支持
1997-02	Visual Basic 5.0	包含了对用户自建控件的支持，且从这个版本开始 VB 可以支持中文
1998-10	Visual Basic 6.0	

## 0.2 Visual Basic 的特点

VB 的中心思想就是要便于程序员使用，无论是新手或者专家。VB 使用了可以简单建立应用程序的

窗体控件的增加和改变可以用拖放技术实现。一个排列满控件的工具箱用来显示可用控件（比如文本框或者按钮）。每个控件都有自己的属性和事件。默认的属性值会在控件创建的时候提供，但是程序员也可以进行更改。很多的属性值可以在运行时随着用户的动作和修改进行改动，这样就形成了一个动态的程序。举个例子来说：窗体的大小改变事件中加入了可以改变控件位置的代码，在运行时每当用户更改窗口大小，控件也会随之改变位置。在文本框中的文字改变事件中加入相应的代码，程序就能够在文字输入的时候自动翻译或者阻止某些字符的输入。

VB 的组件既可以拥有用户界面，也可以没有。这样一来服务器端程序就可以处理增加的模块。

VB 使得大量的外界控件有了自己的生存空间。大量的第三方控件针对 VB 提供。VB 也提供了建立、使用和重用这些控件的方法,但是由于语言问题,从一个应用程序创建另外一个并不简单。Visual Basic 语言具有不支持继承、无原生支持多线程、异常处理不完善等三项明显缺点,使其有局限性。

(2) 无原生支持多线程。Visual Basic 对于多线程无原生支持，只能通过 Windows API 的调用实现，且极其的不稳定。因为在 API 创建的线程中，并没有自动初始化运行时库，导致部分的函数无法使用。一般地，在早期的 VB 开发环境下，使用 API 创建线程的目的是完成容易使程序假死的大量数据或者逻辑的计算。

及数据库管理系统配合,还可以建立网站、设计 Web 网页、开发网络应用软件等,例如,网上银行服务系统、网上订票系统、网上电子商务系统、网上数据查询系统等。使用 Visual Basic 6.0 可以设计出来的系统或软件大到像 Windows 系统,小到像简单的计算器。

## 0.4 Visual Basic 学习经验

学习 Visual Basic 6.0,首先要充分了解该课程的作用和意义,只有这样才能激发起学习兴趣,其次要有学好该课程的决心,另外还要有好的方法指导、好的教材、必要的上机条件和充裕的上机时间,只有这样才能有好的学习效果。

无论学习哪一门科目,都需要兴趣和坚持。兴趣可以慢慢培养,坚持则表现了个人的承受能力。有时候会遇到难以解决的困难,有的人轻言放弃,而有的人却锲而不舍,直到问题的解决。世上成功者与失败者兼有,而那些成功的人大都是有恒心和毅力的。Visual Basic 6.0 是一种简单易学的语言,但想要学的精还需要各位读者有坚持到底的决心,不畏困难的勇气,勇于追求真理的信念。这样,任何一座高山都可以被我们所征服。

学习要踏踏实实,编程生活要耐得住寂寞。真正掌握了核心知识,才能海阔天空。古人有诗云:“路漫漫其修远兮,吾将上下而求索。”

## 0.5 Visual Basic 的学习路线

编程语言的学习就是坚持看、敲、写的过程。

(1) 要学好 Visual Basic 6.0,首先要买一本好的入门书籍,本书把 Visual Basic 6.0 所涉及的内容能够详细地讲解到,对于新手来说是个不错的选择。

(2) 先看书,看得差不多似乎明白的时候,一定要把程序敲出来自己运行一遍。否则,容易产生眼高手低的错误。

(3) 读程序。去论坛或者百度文库找一些 Visual Basic 程序的例子,试着去读懂。

(4) 自己改写程序。通过前面的学习,此时应该能掌握一些基本的编程的技巧。一定要有自己的想法,然后让自己的想法变成程序来实现。编程语言的学习就是坚持的过程,只要一门啃下来,再去学习其他的语言就很轻松了。

# 第1篇

## 基础知识

本篇通过创建第1个 Visual Basic 应用程序、Visual Basic 语言基础、算法和程序控制结构、数组以及内置函数与过程等内容，结合大量的实例讲解，使读者快速熟悉 Visual Basic 的开发过程，为接下来的学习打下良好的基础。

第1章 步入 VB 开发之门——初识 Visual Basic 6.0

第2章 Visual Basic 的入门钥匙——Visual Basic 语言基础

第3章 Visual Basic 的秘密——算法和程序控制结构

第4章 同类型批量数据管理的技巧——数组

第5章 应用程序提升的法宝——内置函数与过程



# 第1章



本章视频教学录像：48 分钟

## 步入 VB 开发之门 ——初识 Visual Basic 6.0

Visual Basic 一经推出就大受欢迎，成为 Windows 下使用最广泛的程序开发语言之一。Visual Basic 采用事件驱动的编程方式，控件编程的方法使开发人员能快速地建立起功能强大的应用程序。本章是初学者入门的第 1 章。在本章中，简要介绍了 Visual Basic 的发展、功能特点等。为了使读者快速入门，着重讲解了 Visual Basic 安装的软硬件环境、安装过程和启动过程等。此外，对 Visual Basic 6.0 的集成开发环境，包括其主要组成部分，都进行了详细介绍。最后通过一个具体的实例，介绍了如何通过 Visual Basic 开发应用程序，使读者能够迅速掌握使用 Visual Basic 6.0 开发应用程序的方法和步骤。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解 Visual Basic 6.0 的发展和特点
- ☐ 掌握 Visual Basic 6.0 的安装与启动
- ☐ 掌握如何定制开发环境
- ☐ 熟悉 Visual Basic 6.0 的开发环境
- ☐ 熟悉如何使用 Visual Basic 6.0 的帮助系统
- ☐ 熟悉使用 Visual Basic 6.0 管理工程
- ☐ 掌握第 1 个 VB 应用程序的创建

## 1.1 Visual Basic 简介

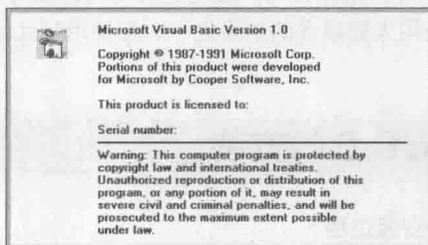


本节视频教学录像: 6 分钟

Visual Basic 6.0 是微软公司开发的一种通用的基于对象的程序设计语言, 也是 Visual Studio 6.0 中的重要成员。Visual Basic 6.0 主要用于开发基于 Windows 的应用程序, 在数据库、分布式处理、Internet 及多媒体等方面的应用十分广泛, 是一个成熟稳定的开发工具。

### 1.1.1 Visual Basic 的发展

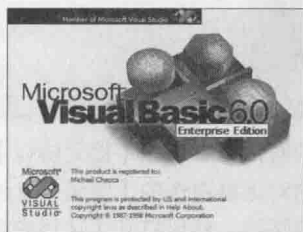
1991 年, 微软公司推出了 Windows 3.0, 其“图形化”和“易于使用”的特点受到了广大用户的热烈欢迎。为了方便 Windows 下的程序开发, 1991 年微软公司推出了颠覆传统开发方式的革命性的 Visual Basic 1.0。使用 Visual Basic 1.0 开发 Windows 应用程序, 不需要手动编写图形界面的代码, 只要通过鼠标的简单拖曳绘制出软件界面, Visual Basic 1.0 就会自动生成图形界面代码。这大大降低了 Windows 下软件开发的难度, 也极大地提高了开发的效率。



随着软件开发技术的发展, Visual Basic 不断加入新的功能, 对各种成熟的编程思想提供更多的支持, 随之在 1998 年推出了 Visual Basic 6.0。

在各种各样的开发环境中, Visual Basic 6.0 已经成了快速应用程序开发工具的一个代表。借助 Visual Basic 6.0 新加入的功能, 使用 Visual Basic 可以轻松地开发多层结构的分布式应用程序以及高效的 Web 应用程序。

Visual Basic 6.0 包含 3 个版本, 分别为学习版、专业版和企业版, 可以满足不同开发者的使用需要。



### 1.1.2 Visual Basic 6.0 的功能特点

Visual Basic 6.0 的发展与它本身具有的特点是分不开的, Visual Basic 6.0 的主要特点有以下几方面。

(1) 易于学习和使用。

Visual Basic 基于图形界面的开发环境使开发者能对各种功能一目了然、容易理解,用户仅仅通过鼠标的简单操作就可以构建出一个复杂的软件图形界面。

(2) 开发高效,功能强大。

程序员可以轻松使用 Visual Basic 提供的各种功能组件快速搭建一个应用程序。在数据库编程方面,使用 DAO、RDO、ADO 等控件可以直观、高效地完成各种数据库操作。

(3) 广泛的用户基础。

Visual Basic 是世界上使用人数最多的语言之一,更多的开发者会带来更多的思想,更多的交流和更多的使用机会

## 1.2 Visual Basic 6.0 的安装与启动



本节视频教学录像: 13 分钟

综上所述, Visual Basic 是一种由微软公司开发的包含协助开发环境的事件驱动编程语言。到目前为止, Visual Basic 仍是世界上使用人数最多的语言之一。在使用 Visual Basic 6.0 之前,需要先将其安装至本地计算机。

### 1.2.1 Visual Basic 6.0 的安装

下面介绍 Visual Basic 6.0 的安装过程。

(1) 将 VB 6.0 的安装光盘放入光驱,系统会自动执行安装程序。如果不能自动安装,可以双击安装光盘中的 SETUP.EXE 文件,执行安装程序,将弹出安装程序向导。



(2) 单击【下一步】按钮,在弹出的对话框中选中【接受协议】单选选项。

(3) 单击【下一步】按钮,在弹出的对话框中选中【安装 Visual Basic 6.0 中文企业版】单选选项。

(4) 单击【下一步】按钮,设置安装路径,然后打开【选择安装类型】对话框。

(5) 在【选择安装类型】对话框中,如果选择【典型安装】,系统会自动安装一些最常用的组件;如果选择【自定义安装】,用户则可以根据自己的实际需要有选择地安装组件。

(6) 单击【下一步】按钮,弹出版权警示与说明内容对话框。

(7) 单击【继续】按钮,选择安装路径与安装模式后,系统将开始自动安装 VB 6.0。安装完成后,系统将提示【重新启动计算机】,以便进行一系列的更新及配置工作。当 VB 6.0 安装完成后,将提示用

户是否安装 MSDN 帮助程序。

(8) 如果要安装 MSDN 帮助文件，应将 MSDN 帮助文件光盘放入光驱，按提示进行安装。完成安装 MSDN 程序后，在 VB 6.0 开发环境中按 F1 键，即可打开 MSDN 帮助程序。如果用户不想安装 MSDN，则只需在安装界面中取消 MSDN 安装选项即可。

(9) 安装 VB 6.0 的 SP6 补丁。

为了使安装的 VB 6.0 更加完整和全面，在安装完 VB 6.0 以后还需要安装补丁程序 SP6。SP6 补丁程序可以到微软的网站上自行下载，其下载后是一个可执行文件，双击图标即可安装，这里不再赘述。

## 1.2.2 Visual Basic 6.0 开发环境的定制

在 Visual Basic 6.0 的集成开发环境中可以根据自己的需要定制开发环境，如定制各子窗口、工具栏、通用选项等。

### 1. 在编辑器中设置要求强制变量声明

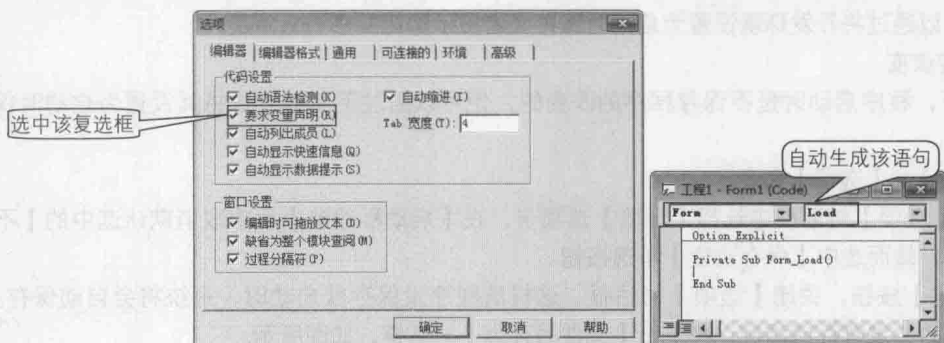
用户可以通过【选项】对话框设置在代码中要求强制的变量声明。具体操作步骤如下。

(1) 选择【工具】>【选项】命令，打开【选项】对话框。

(2) 选择【编辑器】选项卡，在【代码设置】栏中选中【要求变量声明】复选框。

(3) 单击【确定】按钮，完成设置。

这样，在代码的编辑区域中将自动添加 Option Explicit 语句。操作过程如图所示。



### 2. 设置网格大小和不对齐到网格

在窗体上有一些排列整齐的点，这些点构成了相互交错的网格，VB 利用这些网格可精确地确定控件的位置。这些网格的大小是可以调整的。有时出于实际的需要，也可以将控件设置为不对齐到网格，这样在调整控件位置时就可以利用 Ctrl 键加上 ↑、↓、←和→键来微调控件的位置。具体设置方法如下。

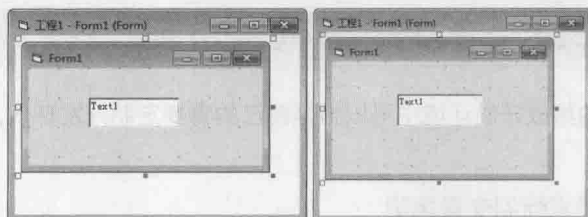
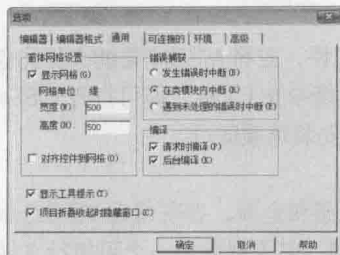
(1) 选择【工具】>【选项】命令，弹出【选项】对话框。

(2) 在【选项】对话框中选择【通用】选项卡。

(3) 在【窗体网格设置】栏中选中【显示网格】复选框。如果取消选中该复选框，在窗体上将不显示网格。在【宽度】和【高度】文本框中设置网格的大小，默认的大小为 120×120。这里为了突出显示效果，将其设置为 500×500。

(4) 取消选中【对齐控件到网格】复选框，单击【确定】按钮，完成设置。

具体操作过程如图所示。



网格大小为 500×500 时，      网格大小为 500×500 时，  
对齐到网格的效果                  不对齐到网格的效果

### 3. 设置启动时保存

工程的保存是程序设计中很重要的一个环节，在修改程序时若不及时保存，当程序出现错误自动关闭时会将之前编写的代码全部丢失，这样就需要重新编写代码，从而给程序的开发带来不必要的麻烦。在程序开发时，可以通过将开发环境设置为启动时保存或者提示保存的形式来解决。

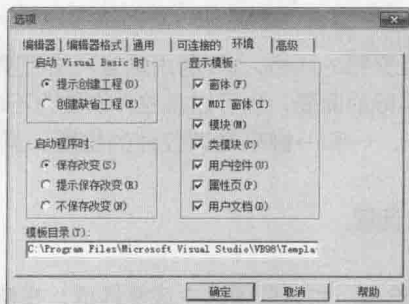
#### (1) 启动时保存改变

在默认情况下，程序启动时是不保存程序的改变的，但可以通过下面的方法将其设置为启动时保存。具体步骤如下。

① 选择【工具】>【选项】命令。

② 在弹出的【选项】对话框中选择【环境】选项卡，在【启动程序时】栏中取消默认选中的【不保存改变】复选框，转而选中【保存改变】单选按钮。

③ 单击【确定】按钮，关闭【选项】对话框。这样当程序没保存就启动时，系统将会自动保存。如果工程为新创建的，没有存储路径，将弹出【文件另存为】对话框，如图所示。



#### (2) 启动时提示保存

用户也可以将开发环境设置为在启动时提示是否保存。只需在【选项】对话框中【启动程序时】栏中选中【提示保存改变】单选按钮，即可在程序启动时，弹出【提示】对话框，询问是否保存，如图所示。



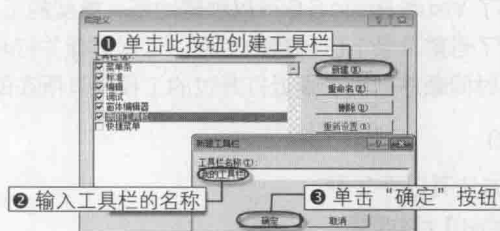
#### 4. 定制工具栏

工具栏是将一些菜单中经常使用的命令以按钮的形式组合在一起，使用起来更加方便、快捷。不过，有时系统提供的工具栏不能完全满足用户的需求，此时便可自定义工具栏。例如，如果有些菜单命令经常会用到，就可以将其添加到工具栏中。

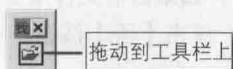
下面根据个人的需要创建一个工具栏，用于存放自己经常使用的菜单命令。具体操作方法如下。

(1) 选择【视图】>【工具栏】>【自定义】命令，在弹出的【自定义】对话框中选择【工具栏】选项卡，单击【新建】按钮，打开【新建工具栏】对话框。

(2) 在该对话框中输入要创建的工具栏的名称，在此输入【我的工具栏】，单击【确定】按钮，完成工具栏的添加。



(3) 再次启动【自定义】对话框，选择【命令】选项卡，拖动想要添加的命令到【我的工具栏】，例如拖动【打开工程】命令到【我的工具栏】。



(4) 重复步骤(3)，将需要的命令都添加到【我的工具栏】上，单击【关闭】按钮，关闭【自定义】对话框。



#### 5. 为代码编辑器设置鼠标滚动

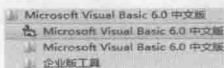
默认安装的 VB 6.0 在编辑代码时，并不支持鼠标滚动，给用户的程序开发带来很大的不便。微软提供了支持鼠标滚动的 DLL，用户可以到微软网站下载这个 DLL 文件，从而使自己的开发环境具有支持鼠标滚动的功能。



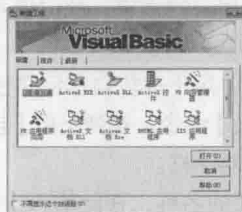
## 1.2.3 启动与退出

### 1. 启动 Visual Basic 6.0

选择【开始】>【所有程序】>【Microsoft Visual Basic 6.0 中文版】>【Microsoft Visual Basic 6.0 中文版】菜单命令，即可启动 Visual Basic 6.0。



Visual Basic 6.0 的起始界面如图所示。




在起始界面中，包含以下 3 个选项卡。

- (1)【新建】选项卡。列出了 Visual Basic 6.0 可以创建的所有类型的工程。
- (2)【现存】选项卡。列出了当前目录下已创建的工程，可以选择并打开它们。
- (3)【最新】选项卡。按照时间顺序列出了最近打开过的工程及其所在的文件夹。

### 2. 退出 Visual Basic 6.0

退出 Visual Basic 6.0 的方法有以下 3 种。

- (1) 单击窗口右上角的【关闭】按钮 .
- (2) 选择【文件】>【退出】菜单命令。
- (3) 使用快捷键【Alt+Q】。

使用以上方法退出 Visual Basic 6.0 时，如果当前文件未保存，系统会弹出是否保存文件的对话框。单击【是】按钮，保存文件的更改后退出；单击【否】按钮，不保存文件的更改并退出；单击【取消】按钮，取消退出。

## 1.3 Visual Basic 6.0 的集成开发环境



本节视频教学录像：12 分钟

集成开发环境是 Visual Basic 6.0 的开发环境，所有开发活动都在该环境下进行。Visual Basic 6.0 的集成开发环境由 9 部分组成：主窗口，控件工具箱，窗体编辑器，属性窗口，代码编辑器，工程资源管理器，布局窗口，对象浏览窗口，立即、本地和监视窗口。

### 1.3.1 认识 Visual Basic 6.0 的工作界面

启动 Visual Basic 6.0，弹出【新建工程】对话框。



部分选项对应的含义如下。

(1)【标准 EXE】选项：建立一个标准的 EXE 工程。

(2)【ActiveX EXE】选项和【ActiveX DLL】选项：这两种应用程序只能在专业版和企业版中建立。两种程序在功能上是一致的，只是包装不同，前者包装成 EXE（可执行）文件，后者包装成 DLL（动态链接库）文件。



**提示**

DLL 的全称是 Dynamic Link Library，中文叫作“动态链接文件”，是一种可执行文件，它允许程序共享执行特殊任务所必需的代码和其他资源。DLL 文件不是独立运行的程序，它是某个程序的一部分，只能由所属的程序调用，用户不需要打开它。

(3)【ActiveX 控件】选项：只能在专业版或企业版中建立，主要用于开发用户自己定义的 ActiveX 控件。

(4)【VB 应用程序向导】选项：该向导用于在开发环境中直接建立新的应用程序框架。

(5)【数据工程】选项：主要提供开发数据报表应用程序的框架。

(6)【IIS 应用程序】选项：用 Visual Basic 代码编写服务器端的 Internet 应用程序。

(7)【外接程序】选项：用于建立用户自己的 Visual Basic 外接程序，并在开发环境中自动打开连接设计器。

选择【新建】选项卡中的【标准 EXE】图标，单击【打开】按钮。

此时将弹出 Visual Basic 6.0 集成开发环境的主界面。Visual Basic 6.0 集成开发环境提供了设计、运行和调试应用程序所需的各种工具，用户在其中无需打开额外的程序就可以设计、运行和调试程序。

Visual Basic 6.0 集成开发环境包含标题栏、菜单栏、工具栏、工具箱、窗体设计器窗口、工程资源管理器窗口、工程窗口、窗体布局窗口和属性窗口等。



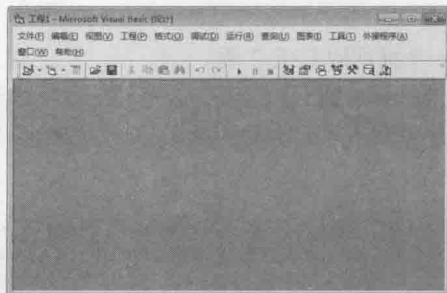


提示

Windows 下的应用程序的界面和菜单有很多相通的地方, 不同软件的命令和操作有很多是可以通用的。不妨在 Visual Basic 6.0 的界面中找一找哪些地方与你用过的软件有相似之处, 这样会为学习节省不少时间。

## 1.3.2 主窗口

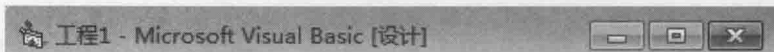
主窗口也称设计窗口。主窗口位于集成环境的顶部, 该窗口由标题栏、菜单栏和工具栏组成。



### 1. 标题栏

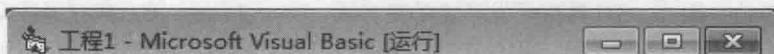
标题栏是 Visual Basic 6.0 集成开发环境屏幕最上面的水平条, 用于显示应用程序的名称和系统的工作状态。Visual Basic 6.0 有以下 3 种工作状态。

#### (1) 设计状态



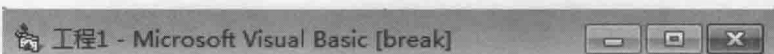
可进行用户界面的设计和代码的编辑, 以完成应用程序的开发。

#### (2) 运行状态



运行应用程序。此时不可编辑界面和代码。

#### (3) 中断状态



暂时中断应用程序的运行。此时可以编辑代码, 但是不能编辑界面。

### 2. 菜单栏

菜单栏位于标题栏的下方, 有【文件】【编辑】【视图】【工程】【格式】【调试】【运行】【查询】【图表】【工具】【外接程序】【窗口】和【帮助】等菜单。Visual Basic 6.0 的所有功能都可以通过菜单中的菜单项来完成。

文件(F) 编辑(E) 视图(V) 工程(E) 格式(O) 调试(D) 运行(R) 查询(Q) 图表(C) 工具(T) 外接程序(A) 窗口(W) 帮助(H)

下表列出了各菜单的主要功能。

图标	功能
文件(F)	工程的建立和管理
编辑(E)	对项目中的源代码以及对象的属性进行编辑操作
视图(V)	根据需要显示或者隐藏相应的窗口、工具栏及其他部分
工程(P)	对各种资源和属性进行管理、设置
格式(O)	管理、调整、设置控件的尺寸和位置等
调试(D)	调试程序，如设置断点和逐句调试等常用的调试功能
运行(R)	启动程序或全编译执行
查询(U)	运行结果查询、数据库 SQL 查询和语言语法查询等查询操作
图表(I)	新建、设置、添加、显示和修改图表等
工具(T)	添加、管理过程，菜单编辑器，选项设置等
外接程序(A)	在当前工程中增加或删除外接程序
窗口(W)	各个窗口间的切换，调整窗口的排列方式
帮助(H)	启动帮助系统。要使用帮助系统需要事先安装 MSDN

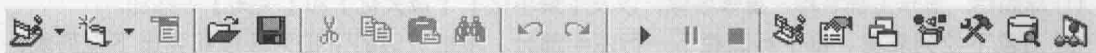
### 3. 工具栏

Visual Basic 6.0 中的工具栏将一些常用功能以图标的形式列出来，只要单击某个图标，就可以执行该图标对应的命令，从而可以提高开发的效率。Visual Basic 6.0 提供有 5 个工具栏，分别是【编辑】工具栏、【标准】工具栏、【窗体编辑器】工具栏、【调试】工具栏和【自定义】工具栏。

默认情况下，集成环境中只显示【标准工具栏】。

工具栏有固定和浮动两种形式。把鼠标指针移到固定形式工具栏中没有图标的地方，按住鼠标，向下拖动，或者双击工具栏左端的两条浅色竖线，即可把工具栏变为浮动的；如果双击浮动工具栏的标题条，则可变为固定工具栏。

固定形式的【标准】工具栏位于菜单栏的下面，它以图标的形式提供常用的菜单命令。



在工具栏的右侧还有两个栏，分别用来显示窗体或控件的当前位置和大小。

其中左边一栏显示的是窗体或控件左上角的坐标，右边一栏显示的是窗体或控件的长 × 宽，其单位为缇 (twip)。缇是一种与屏幕分辨率无关的计量单位，这种计量单位可以确保在不同的屏幕上都能保持正确的相对位置或比例关系。在 Visual Basic 中，缇是默认单位。





提示

缇是一种计量单位, 1 缇等于 1 磅的 1/20, 1 英寸的 1/1440。1cm 有 567 缇, Visual Basic 使用缇做单位。可以确保应用程序的各个控件在不同的分辨率的屏幕上能保持设计时的大小及位置。

除上面几部分外, 在主窗口标题栏的左上角和右上角还有几个控制框, 其作用与 Windows 普通窗口中的控制框相同。

### 1.3.3 窗体设计 / 代码设计窗口

#### 1. 窗体设计窗口

窗体设计窗口简称窗体 (Form), 是最终用户看到的软件界面, 应用程序的运行结果, 各种图形、图像、数据等, 都是通过窗体或窗体中的控件显示出来的。

当打开一个新的工程文件时, Visual Basic 建立一个空的窗体, 并命名为 FormX 这里的 X 为 1, 2, 3……)。



提示

为了在程序设计时便于理解, 最好将 Visual Basic 自动命名的窗体更改为自己定义的名称, 比如将主窗口命名为 FormMain, 子窗口命名为 FormChild 等。

#### 2. 代码设计窗口

代码设计窗口又称代码编辑器, 用来编写或者修改过程或事件过程的代码。双击控件或者窗体空白处, 以及在【工程资源管理器】中双击模块都可以打开代码窗口。



代码设计窗口由下面几部分组成。

(1) 标题栏。显示工程名称、窗体名称, 以及【最小化】、【最大化】和【关闭】等按钮。

(2) 对象下拉列表框。位于标题栏下一行左半部分。单击右边的下拉列表按钮弹出下拉列表, 其中列出了当前窗体及所包含的所有对象名。其中, 无论窗体的名称如何改变, 作为窗体的对象名总是 Form。

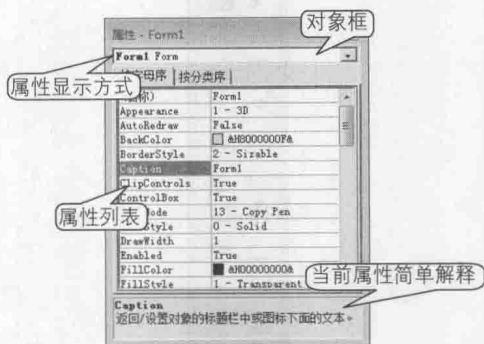
(3) 过程下拉列表框。位于标题栏下一行右半部分。单击右边的下拉列表按钮弹出下拉列表, 其中列出了所选对象的所有事件名。

(4) 代码区。窗口中的空白区域即为代码区, 在其上可编辑程序代码, 使用方法与通常的字处理软件相似。

(5) 【过程查看】按钮 和 【全模块查看】按钮 .

这两个按钮位于代码窗口的左下角，用于切换代码窗口的两种查看视图。单击【过程查看】按钮，一次只查看一个过程；单击【全模块查看】按钮，可查看程序中的所有过程。

### 1.3.4 属性窗口



属性窗口主要是针对窗体和控件设置的，在 Visual Basic 中，窗体和控件被称为对象。每个对象的特征都可以用一组属性来描述，属性窗口就是用来设置窗体或窗体中控件属性的。

属性窗口分为 4 部分，分别为对象框、属性显示方式、属性列表和当前属性简单解释。

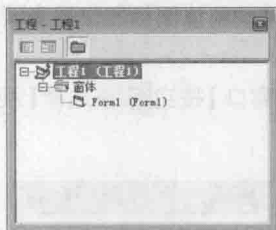
(1) 对象框。位于属性窗口的顶部，可以通过单击其右端向下的箭头显示下拉列表，其内容为应用程序中每个对象的名字及对象的类型。




(2) 属性显示方式。分为两种，即“按字母序”和“按分类序”，可通过单击相应的按钮实现。

(3) 属性列表。显示当前活动对象的所有属性，以便观察或设置每项属性的当前值。属性的变化将改变相应对象的特征。

(4) 当前属性简单解释。每选择一种属性（条形光标位于该属性上），在属性解释部分都会显示该属性名称和功能说明。

### 1.3.5 工程资源管理器窗口



在工程资源管理器窗口的顶部有 3 个按钮，分别为【查看代码】按钮 、【查看对象】按钮  和【切换文件夹】按钮 。

如果单击工程资源管理器窗口中的【查看代码】按钮，相应文件的代码将在代码窗口中显示出来。单击【查看对象】按钮，Visual Basic 将显示相应的窗体。

在工程资源管理器窗口中，含有建立一个应用程序所需要的文件清单。工程资源管理器窗口中的文件有窗体文件（.frm）、程序模块文件（.bas）、类模块文件（.cls）、工程文件（.vbp）、工程组文件（.vbproj）和资源文件（.res）等。



## 1.3.6 工具箱窗口



工具箱窗口由工具图标组成。这些图标是 Visual Basic 应用程序的构件，称为图形对象或控件 (Control)，每个控件由工具箱中的一个工具图标来表示。


工具箱中的工具分为两类，一类称为内部控件或标准控件，另一类称为 ActiveX 控件。启动 Visual Basic 后，工具箱中只有内部控件。

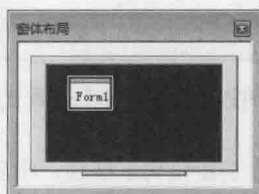
工具箱主要用于应用程序的界面设计。在设计阶段，首先用工具箱中的工具 (即控件) 在窗体上建立用户界面，然后编写程序代码。界面的设计完全通过控件来实现，可任意改变其大小，并可移动到窗体的任何位置。这些控件将在第 4 章和第 5 章中介绍。

## 1.3.7 其他窗口

### 1. 窗体布局窗口

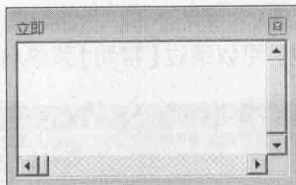
窗体布局窗口用来布置应用程序中各窗体的位置。可将窗体布局窗口看作是一个缩小的屏幕，其中显示出窗体在屏幕上的位置。可以通过拖动窗体图标的位置来调整程序运行时窗体显示的位置。窗体布局窗口主要用来定位窗体的位置。

单击【标准】工具栏中的【窗体布局窗口】按钮或选择【视图】菜单中的【窗体布局窗口】菜单项，都可以打开窗体布局窗口。



### 2. 立即窗口

选择【视图】菜单中的【立即窗口】菜单项，可以打开立即窗口。在中断模式时会自动打开立即窗口。



在中断模式下，可以检查、调试、重置、单步执行或继续执行程序，但是立即窗口中的代码是不能存储到工程中的。




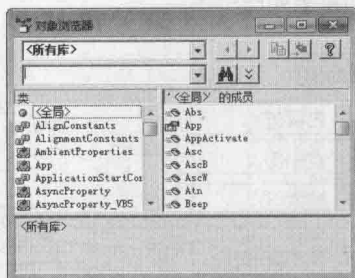
**提示**

双击立即窗口的标题栏，可以把它从 Visual Basic 6.0 的窗口底部单独分离成一个浮动的小窗口，再次双击标题栏可以将其恢复初始的状态。

### 3. 对象浏览器

对象浏览器 (Object Browser) 列出了工程中有效的对象。对象浏览器主要用于 Visual Basic 的对象和应用程序，查看对象的方法和属性，也可将代码粘贴到自己的应用程序中。

单击工具栏中的【对象浏览器】按钮 ，选择【视图】菜单中的【对象浏览器】选项或者直接按快捷键【F2】，即可打开对象浏览器窗口。



## 1.3.8 Visual Basic 帮助系统的使用

本小节主要介绍 MSDN Library 的安装与使用以及使用 VB 的帮助菜单。

### 1. MSDN Library 的安装与使用

MSDN 即 Microsoft Developer Network，是微软公司面向软件开发者提供了一种信息服务。

#### (1) 安装 MSDN Library

在安装 VB 6.0 时，将弹出“安装向导”对话框，在该对话框中选中 MSDN 单选按钮，单击“下一步”按钮，即可安装 MSDN。MSDN 的安装非常简单，在此不再赘述。

#### (2) 启动 MSDN Library

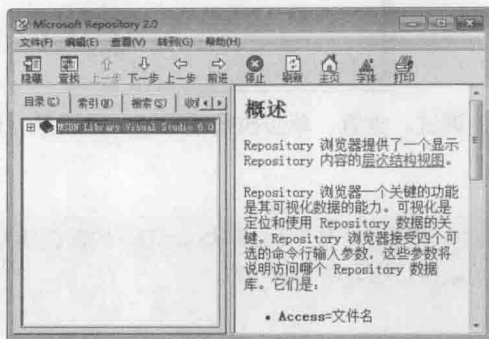
安装完成以后，用户可以通过下面两种方法启动 MSDN。

##### ① 通过【开始】菜单启动

通过在【开始】菜单中，选择【程序】/Microsoft Developer Network/MSDN Library Visual Studio 6.0 (CHS) 命令，启动 MSDN。

## ② 在集成开发环境中启动

如果启动了 VB 6.0 的集成开发环境, 可以通过【帮助】菜单启动 MSDN, 启动后的 MSDN 如图所示。

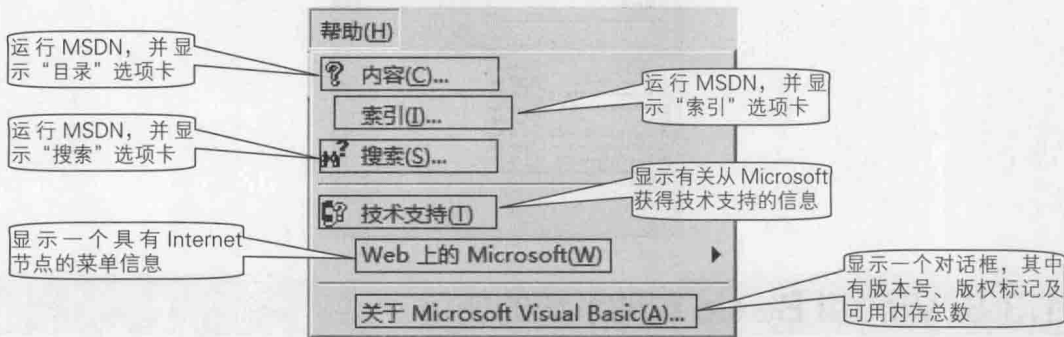


## (3) 使用 MSDN Library

在程序开发过程中, MSDN 可以帮助用户解决程序开发中遇到的相关问题, 用户只需选定需要帮助的相关对象, 然后按【F1】键, 即可获取相关的 MSDN 帮助信息。

## 2. 使用 VB 的帮助菜单

在程序开发过程中, 用户会遇到很多难题或者疑问, 此时 VB 的帮助系统就派上了用场。下面首先介绍一下 VB 的帮助菜单。VB 的帮助菜单如图所示。



## 1.4 用 Visual Basic 6.0 管理工程



本节视频教学录像: 9 分钟

工程管理主要包括工程的保存、关闭、打开和重命名等, 主要通过【文件】菜单完成操作。

### 1.4.1 工程介绍

Visual Basic 的应用与工程有着密切的关系。在 Visual Basic 中无论应用程序的规模是大还是小, 其总对应着一个或几个工程。使用 Visual Basic 进行程序设计就要深刻理解工程的含义。

当使用 Visual Basic 创建应用程序时, 所有文件的有关信息就保存在称为“工程”的文件中。在 Visual Basic 中, 使用工程来管理构成应用程序的所有不同的文件。可以说, 工程文件是与工程相关联

的所有文件和对象以及所设置的环境信息的一个简单列表。

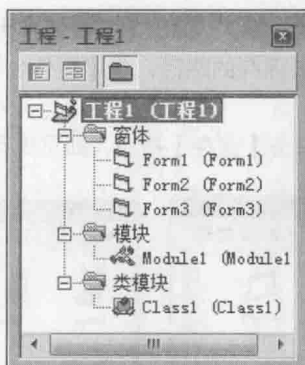
当完成工程的全部文件之后，可将此工程转换成可执行文件（.exe），直接在 Windows 下运行。



#### 提示

使用 Visual Basic 的专业版和企业版，还可以创建其他类型的可执行文件，例如 .ocx 和 .dll 文件（DLL 文件和 OCX 文件只能在专业版和企业版中创建）。

当创建、添加或从工程中删除可编辑文件时，Visual Basic 会在工程资源管理器窗口中反映出所发生的变化，工程窗口包含此工程的当前文件的列表，如图所示。



工程文件就是与该工程有关的全部文件和对象的清单，也包括所设置的环境选项方面的信息。

一个工程包括以下各项。

(1) 跟踪所有部件的工程文件（.vbp）。

(2) 每个窗体的文件（.frm）。

(3) 每个窗体的二进制数据文件（.frx），它含有窗体上控件的属性数据。含有二进制属性（例如图片或图标）的任何 .frx 文件都是不可编辑的，这些文件都是自动产生的。

(4) 1 个或多个类模块文件（.cls），该文件是可选项。

(5) 1 个或多个标准模块文件（.bas），该文件是可选项。

(6) 1 个或多个包含 ActiveX 控件的文件（.ocx），该文件是可选项。

(7) 单个资源文件（.res），该文件是可选项。

每次保存工程时，这些信息都要被更新。所有的这些文件和对象也可供其他工程共享。

工程文件的扩展名是“.vbp”，可以将几个工程“.vbp”文件组成一个工程组“.vbg”文件。



#### 提示

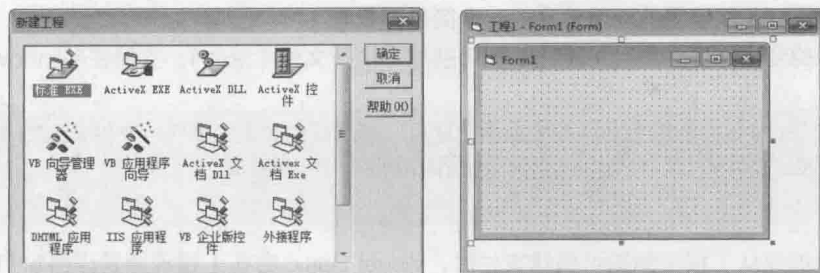
工程文件中并不保存上述文件，只是保存了指向这些文件的指针。

## 1.4.2 新建、保存工程

### 1. 新建工程

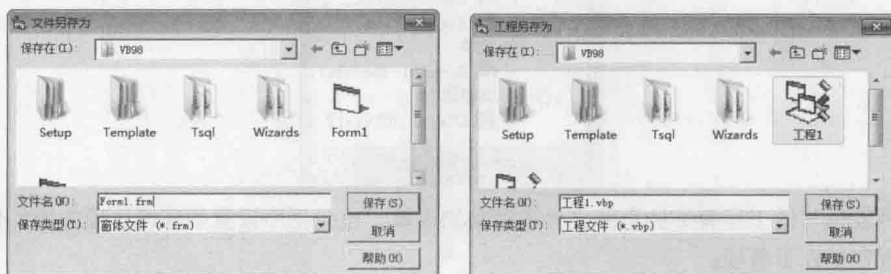
(1) 选择【文件】>【新建工程】菜单命令，弹出【新建工程】窗口。

(2) 选择需要创建的文件类型，单击【确定】按钮。



## 2. 保存工程


- (1) 选择【文件】>【保存工程】菜单命令，弹出【文件另存为】对话框。
- (2) 在【保存在】下拉列表中选择要保存的路径，在【文件名】文本框中输入文件名，单击【保存】按钮。
- (3) 弹出【工程另存为】对话框，单击【保存】按钮，即可保存后缀为“.vbp”的工程文件。

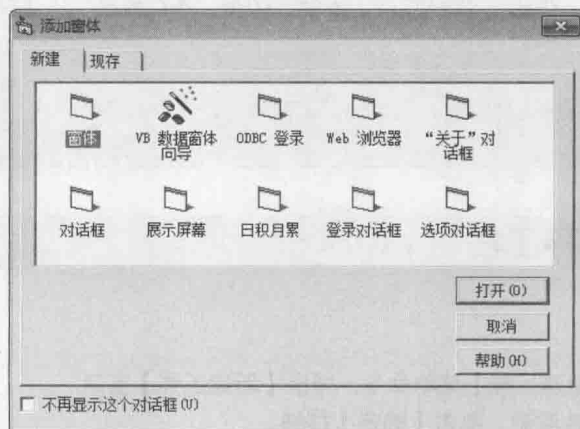


## 1.4.3 向工程中添加窗体和模块

一个工程可以包含一个或多个窗体，在窗体中不仅包含了窗体对象的设计，也包含了窗体代码设计。


### 1. 向工程中添加窗体

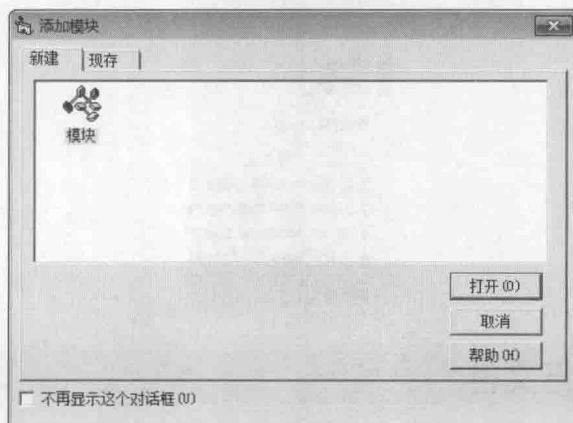
选择【工程】>【添加窗体】菜单命令或者单击工具栏中的“添加窗体”按钮，这时将会出现添加窗体对话框。



在该对话框中有两个页面，如果想要添加新的窗体，就先在“新建”页面中选择要添加的窗体类型，然后单击“打开”即可；如果想要添加已经存在的窗体，则需先选择“现存”页面，再从目录列表中找到相应的窗体文件，最后单击“打开”按钮。

## 2. 向工程中添加模块


选择【工程】>【添加模块】菜单命令或者单击工具栏中的“添加模块”按钮，这时将会出现添加模块对话框。



在该对话框中有两个页面，如果想要添加新的模块，就先在“新建”页面中选择要添加的模块类型，然后单击“打开”即可；如果想要添加已经存在的模块，则需先选择“现存”页面，再从目录列表中找到相应的模块文件，最后单击“打开”按钮。


## 1.4.4 运行和关闭工程

### 1. 打开工程

选择【运行】>【启动】菜单命令，单击工具栏中的按钮或按快捷键【F5】，都可以运行工程。

### 2. 关闭工程

使用以下3种方法均可将工程关闭。

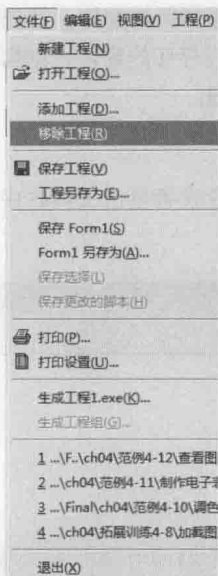
- (1) 选择【文件】>【退出】菜单命令或按快捷键【Alt+Q】。
- (2) 右击 Visual Basic 6.0 开发环境的标题栏，选择【文件】>【关闭】菜单命令或按快捷键【Alt + F4】。
- (3) 单击标题栏右端的按钮。

使用以上方法后，系统提示是否保存修改后的工程，单击【是】按钮，保存工程后退出；单击【否】按钮，不保存工程并退出。

## 1.4.5 删除工程

选择【文件】>【移除工程】菜单命令，可以删除工程。





### 1.4.6 生成可执行文件

选择【文件】►【生成工程 1.exe】菜单命令，弹出【生成工程】对话框。在【保存在】列表框中选择要生成可执行文件的路径，在【文件名】输入框中输入文件名，然后单击【确定】按钮，将生成以【.exe】为后缀的可执行文件。



## 1.5 来自 VB 世界的第一声问候——第 1 个应用程序



本节视频教学录像：9 分钟


在初步了解了 Visual Basic 6.0 的基础知识后，就可以开始设计一个小应用程序来熟悉 Visual Basic 6.0 的操作。该实例是在 Visual Basic 6.0 中编写一个应用程序，要求在窗口中单击【显示】按钮，标签控件中会显示“欢迎来到精彩的 VB 世界”。

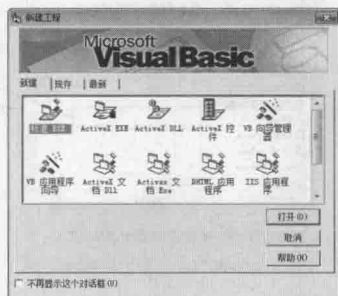
## 1.5.1 VB 程序设计的一般步骤

在进行具体的程序设计前，应先了解使用 Visual Basic 6.0 进行应用程序开发的步骤。其一般性步骤如下。

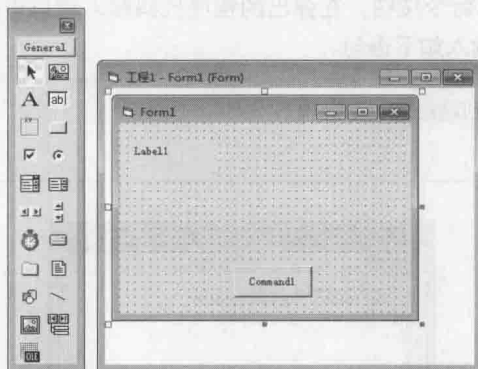
- (1) 创建应用程序的界面。
- (2) 设置控件属性。
- (3) 编写代码。
- (4) 调试、运行程序。

## 1.5.2 创建应用程序的界面

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标，然后单击【打开】按钮。

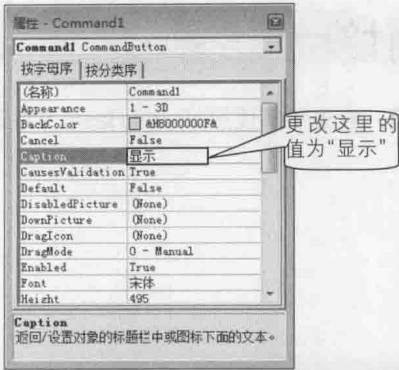


(2) 分别双击工具箱面板上的 Command Button 控件和 Label 控件，并将添加到窗体中的控件摆放如下。

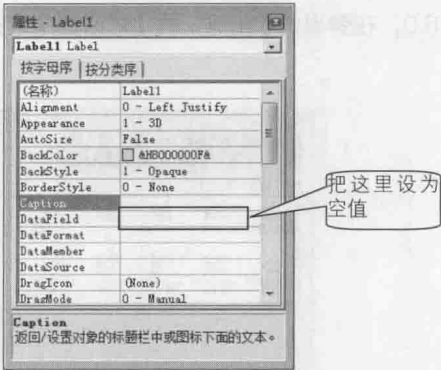


## 1.5.3 设置控件属性

- (1) 单击窗体设计器窗口中的命令按钮，在属性窗口中将显示命令按钮的属性。
- (2) Caption 属性是命令按钮的显示名称，将其改为“显示”。



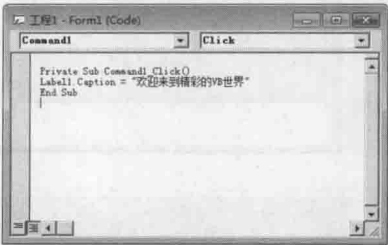
(3) 用同样的方法，将 Label1 控件的 Caption 属性改为空值，也就是不显示任何文字。




### 1.5.4 编写代码

双击窗体设计器窗口上的命令按钮，在弹出的程序代码输入窗口中的“Private Sub Command1\_Click()”和“End Sub”之中输入如下语句。

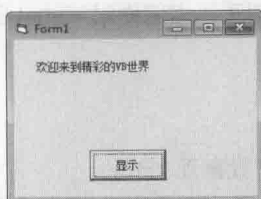
```
Label1.Caption = "欢迎来到精彩的 VB 世界"  
' 设置 Caption 属性值
```



### 1.5.5 调试、运行程序

(1) 输入完毕，关闭代码输入窗口，单击工具栏中的【启动】按钮  (或者直接按快捷键【F5】)，

可以看到运行后的程序界面。单击【显示】按钮，在标签控件中就会显示“欢迎来到精彩的 VB 世界”字样。



(2) 选择【文件】>【保存工程】菜单命令，在弹出的【文件另存为】对话框中，将窗体保存为“显示内容.frm”，单击【保存】按钮，在弹出的【工程另存为】对话框中，将工程保存为“显示内容.vbp”，然后单击【保存】按钮。



(3) 选择【文件】>【生成工程 1.exe】菜单命令，将工程生成成为“显示内容.exe”。



输入代码的时候，引号要用半角的英文引号。如果输入全角的引号，则会出错。

**注意**

## 【运行结果】

双击生成的“显示内容.exe”应用程序，在窗口中单击【显示】按钮，标签控件中会显示“欢迎来到精彩的 VB 世界”。



## 【范例分析】

通过这个简单的例子,我们掌握了在 Visual Basic 6.0 中设计程序的一般步骤,即添加控件—调整界面—编写代码—调试运行。通过这个例子,相信你对 Visual Basic 6.0 的图形化程序设计的高速快捷有了一定的亲身体会。

## 【拓展训练】

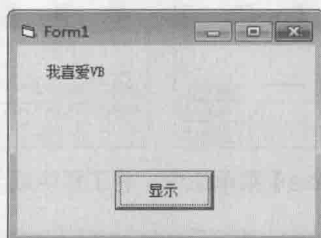
单击【显示】按钮,在标签中输出“我喜爱 VB”。

如果想按下【显示】按钮,在标签中输出“我喜爱 VB”这几个字,该怎么办呢?

只需把 1.5.4 中的代码里引号中间的内容修改为需要显示的内容即可。引号仍然要用半角的符号。

```
Label1.Caption = "我喜爱 VB"
```

运行后单击【显示】按钮,标签中将显示“我喜爱 VB”。



## 1.6 实战练习

在 Visual Basic 6.0 中编写一个应用程序,实现以下功能。

- (1) 程序界面上包含 1 个命令按钮和 1 个标签,命令按钮文字为“显示”。
- (2) 标签内容初始为空。
- (3) 单击按钮,标签内容则变为“这是我的 VB 程序”。

# 第2章



本章视频教学录像：1 小时 5 分钟

## Visual Basic 的入门钥匙 ——Visual Basic 语言基础

对任何新知识的学习都是开始时最难，虽然 Visual Basic 一向以易学易用而著称，但是对那些从未接触过 Visual Basic 的初学者来说，由于编码需要遵循一定的规则，刚开始难免会因各种各样的语言规则，不知从何入手。为了学好如何应用 Visual Basic 程序进行编码，就必须要把基本的语言使用规则基础打好。本章将带领读者从 Visual Basic 的语言基础开始循序渐进地学习。介绍 Visual Basic 语言的基本元素和程序控制语句，主要包括标识符、数据类型、运算符、表达式、数组、程序控制结构和代码编写规则等。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解标识符和常用数据类型
- ☐ 了解常量和变量
- ☐ 掌握各种运算符
- ☐ 熟悉表达式的用法
- ☐ 了解代码编写规范



## 2.1 标识符和数据类型



本节视频教学录像：22 分钟

标识符和数据类型是 Visual Basic 程序的基本构成要素，本节对这两个概念进行阐述。

### 2.1.1 标识符

标识符是用户在编程时定义的名称，包括常量、变量、过程、函数和类等，在实际操作这些变量的时候我们需要为每个变量起一个名字以示区别。这就像在现实生活中每一个人、每一个商店都会有一个名字一样。

在 Visual Basic 中有两种标识符，分别是系统关键字和用户自定义标识符。

系统关键字是 Visual Basic 中拥有固定含义，不能被重新定义的标识符。其实在现实生活中也有这样的关键词，如计算机、手机、卫生间，这些词已经被明确赋予了它的意义，我们起名字的时候就不会再使用这些词。

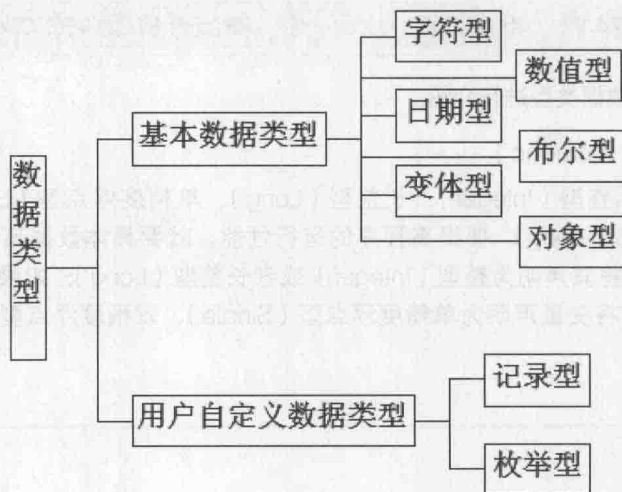
Visual Basic 中常见的关键字如表所示。

关键字名称	作用
As	定义一个变量
Binary	二进制方式读写文件
ByRef	按地址传递参数
Byval	按值传递参数
Date	日期函数
Else	条件语句中的“否则”
Empty	表示未初始化的变量值
Error	错误对象
Input	读写文件的方式
Len	获得字符串的长度
Let	属性定义的关键字
Me	当前对象，通常表示 form 或者 ActiveX 的 usercontrol
Mid	从一个字符串中提取子字符串
New	实例化一个对象变量
Null	表示变量不包含有效数据
Nothing	将对象变量从实际对象中分离出来
Option	模块全局申明，比如 Option Explicit 表示强制定义变量

关键字名称	作用
Print	打印函数
Private	定义函数、过程或者变量的作用域
Public	定义函数、过程或者变量的作用域
Property	定义属性
Resume	再继续, 重新开始, 重新占用, 再用, 恢复
Seek	移动文件指针
Static	定义静态变量
String	字符数据类型

## 2.1.2 数据类型

数据类型是用抽象的方式来描述客观事物的一种定义, 其不同的数据类型定义了不同的数据存储方式。计算机语言正是通过选择不同的数据类型来准确地对客观事物进行描述。数据类型的整体分类如图所示。



图中对 Visual Basic 6.0 中数据类型的整体情况进行了描述。在细节方面, 考虑到数据的运算效率和精度要求, 又将数值型分为整型、长整型、单精度型、双精度型、字节型和货币型等。各类型的字节取值和所占空间范围如表所示。

数据类型	所占字节	范围
布尔型 (Boolean)	2	True 或 False
字节型 (Byte)	1	0 ~ 255
整型 (Integer)	2	-32 768 ~ 32 767
长整型 (Long)	4	-2 147 483 648 ~ 2 147 483 647
单精度浮点型 (Single)	4	-3.402 823E+38 ~ -1.401 298E-45 1.401 298E-45 ~ 3.402 823E+38
字符串 (String) 变长	10+ 字符串长	0 ~ 20 亿个字符左右
字符串 (String) 定长	字符串长	1 ~ 65 400 个字符左右
双精度浮点型 (Double)	8	-1.797 693 134 862 315 E+ 308 ~ -4.940 66E - 324 4.940 66E-324 ~ 1.797 693 134 862 315E +308
日期型 (Date)	8	100 年 1 月 1 日 ~ 9 999 年 12 月 31 日
货币型 (Currency)	8	-922 337 203 685 477.5808 ~ 922 337 203 685 477.580 7
对象型 (Object)	4	任何对象引用
变体 (Variant) 字符	22+ 字符串长	0~20 亿个字符左右
变体 (Variant) 数值	16	-1.797 693 134 862 315 E+308 ~ -4.940 66E-324 4.940 66E-324 ~ 1.797 693 134 862 315E+308
自定义型 (User-defined)	申请的长度	每个元素的范围同它的数据类型的范围

下面对表中的各种数据类型进行介绍。

### 1. 数值数据类型 (Numeric)

数值数据类型包括整型 (Integer)、长整型 (Long)、单精度浮点型 (Single)、双精度浮点型 (Double) 和货币型 (Currency)。要提高程序的运行性能,就要具体数据具体对待。如果变量中存放的数据是整数,就要将其声明为整型 (Integer) 或者长整型 (Long); 如果变量中存放的数据是带小数的数字,就应该将变量声明为单精度浮点型 (Single)、双精度浮点型 (Double) 或者货币型 (Currency)。

例如:

```
Dim s As Integer
s=3.65
print s
```

运行程序后,会在窗体中直接输出一个数值 4。这里需要注意的是,将整型或货币型的数值赋给整型变量后,VB 会自动对数值的小数部分进行四舍五入。

### 2. 字节数据类型 (Byte)

如果变量中存放的是二进制数,则要将变量声明为字节数据类型 (Byte)。特别是对变量中存储的

二进制数据需要进行类型转换。如果不用字节数据类型存储这些二进制数，可能会导致二进制数据的丢失或损毁。



**提示**

在计算机中通常把用来存储一个英文字母的 8 个二进制位叫作一个字节。一个英文字母（不分大小写）占一个字节的空间，一个中文汉字占两个字节的空间。符号中英文标点占一个字节，中文标点占两个字节。

### 3. 字符串数据类型 (String)

字符串是指除了双引号和回车符号之外的所有可以输出的字符。字符串包括在双引号中，如果变量中保存的数据都是字符串（例如人名），就应该把变量声明为字符串类型。在 Visual Basic 中，String 变量或参数默认是一个可变长度的字符串，随着对字符串赋予新数据，它的长度可增可减。当然，也可以声明字符串具有固定长度，例如：

```
Dim Name As String * 20
```

这条语句声明变量 Name 是一个长度为 20 个字符的定长字符串常量。如果你声明了一个定长字符串变量的长度后实际存储的字符长度小于声明长度，Visual Basic 会自动将剩余的部分填充空格；如果赋予字符串的长度超过了声明长度，Visual Basic 会自动将超出长度部分的字符截去。

### 4. 布尔数据类型 (Boolean)

布尔数据类型又叫作逻辑型数据。如果变量的值只有两个对立的状态，如“True”或“False”、“Yes”或“No”，就应该将变量声明为布尔数据类型。在 Visual Basic 中布尔数据类型的默认值为 False。



**提示**

布尔是英国的数学家和逻辑学家，1815 年 11 月 2 日生于林肯，1864 年 12 月 8 日卒于爱尔兰的科克。为了纪念他在逻辑学方面的卓越贡献，人们将表示逻辑变量的值以他的名字命名。

### 5. 日期型数据类型 (Date)

如果变量中要保存的数据是用来表示日期和时间的，就应该将变量声明为日期数据类型。日期数据类型在代码中进行赋值的时候，必须用两个英文“#”号括起来。例如：

```
Dim s As Date
s = #11/23/2009#
Print s
```

最终的结果将会在窗体上显示“2009-11-23”字样。



**提示**

当其他数值类型的数据转换为日期数据类型的时候，小数点左边的值表示日期，小数点右边的值表示时间。如果是负数，则表示公元 1899 年 12 月 31 日前的日期。

### 6. 对象数据类型 (Object)

对象数据类型 (Object) 的变量作为 32 位 (4 个字节) 地址来存储，通过这个地址可以引用当前

应用程序或者其他应用程序中的对象。

### 7. 变体型数据类型 (Variant)

变体型数据类型 (Variant) 变量能够存储所有系统定义类型的数据。如果将其赋予 Variant 变量, 则不必在这些数据的类型间进行转换, Visual Basic 会自动完成任何必要的转换。



**提示**

在 Visual Basic 6.0 中可以将所有的变量都设置为变体类型, 同时这似乎看起来也很方便, 不用再麻烦地定义其他的数据类型。然而, 在程序运行的时候, 因为程序变量的空间分配、数据类型的转换等都会占用大量的系统空间, 因此不建议读者在编写代码的时候使用变体型数据类型, 而应根据实际需求, 合理地使用变量类型。

### 8. 自定义数据类型

上面介绍的都是 Visual Basic 中的基本数据类型, 但是在实际编程中这些基本数据类型并不能满足所有的需求。例如, 我们设计了一个员工薪水计算的程序, 在这个程序中, 需要用一种数据类型来描述员工的个人信息, 而员工的个人信息是由姓名、性别和职位等很多项组成的。没有一种基本数据类型可以直接描述这个数据类型。这个时候可以通过将基本数据类型予以组合来创建一个符合我们自身需求的新数据类型, 这种数据类型就是自定义数据类型。自定义数据类型通过 Type 语句来实现。

语法规则如下所示。

Type 自定义类型名

元素名 1 As 类型名


元素名 2 As 类型名

...

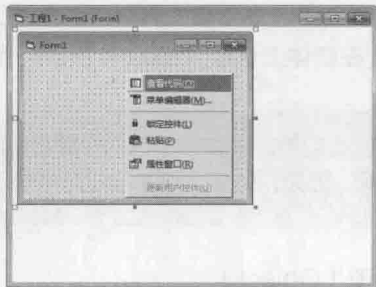
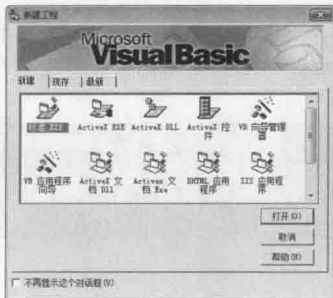
元素名 n As 类型名

End Type

## 【范例 2-1】根据自身需求, 自定义一个 employee 变体类型, 包含员工的姓名、性别、年龄和职位等字段。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标, 然后单击【打开】按钮。

(2) 在【Form1】窗体上单击右键, 在弹出的快捷菜单中选择【查看代码】菜单项, 进入代码窗口。





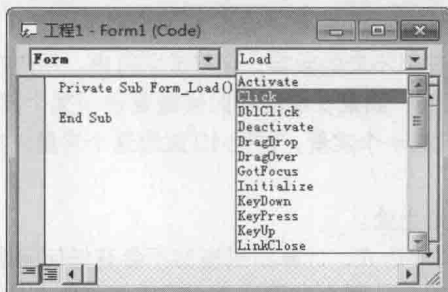
#### 技巧

用鼠标双击窗体或者选择【运行】菜单的【启动】命令，也可以进入代码窗口。

(3) 在代码窗口的顶端输入以下代码（代码 2-1-1.txt）。

```
01 Private Type employee '自定义员工的数据类型
02   name As String '员工姓名
03   sex As String '员工性别
04   age As Integer '员工年龄
05   position As String '员工职位
06 End Type '结束定义
07 Dim employee1 As employee '声明一个自定义类型的变量
```


(4) 在代码窗口中选择【Form】窗体的【Click】事件。



(5) 输入以下代码（代码 2-1-2.txt）。

```
01 Private Sub Form_Click()
02   employee1.name = "王小强" '为 employee1 对象的 name 成员赋值
03   employee1.sex = "男" '为 employee1 对象的 sex 成员赋值
04   employee1.age = "25" '为 employee1 对象的 age 成员赋值
05   employee1.position = "社长" '为 employee1 对象的 position 成员赋值
06   Print "员工信息:" '输出标题
07   Print '输出空行
08   Print '输出空行
09   Print "姓名:" & employee1.name '输出姓名
10   Print "性别:" & employee1.sex '输出性别
11   Print "年龄:" & employee1.age '输出年龄
12   Print "职位:" & employee1.position '输出职位
13 End Sub
```

### 【运行结果】

保存程序，单击【启动】按钮, 运行程序。用鼠标单击窗体，结果如图所示。





## 2.2 常量和变量



本节视频教学录像: 17 分钟

在 Visual Basic 中对数据的操作离不开常量和变量。那什么是变量, 什么是常量, 常量和变量各有什么特点, 在操作常量和变量的时候需要注意些什么? 本节将介绍相关内容。

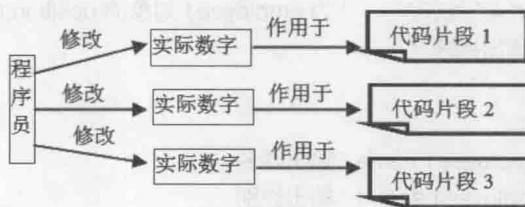
### 2.2.1 常量

常量是指在程序运行过程中其值不变的数字、字符或字符串。用户在编程时使用的常数就是一种常量。有的读者可能会有这样的疑问, 如果在编程的时候需要使用某个具体数字, 比如 10, 那么直接写上 10 不就好了, 为什么还要先定义一个常量, 再把 10 赋给这个常量, 然后使用这个常量, 这不是多此一举吗?

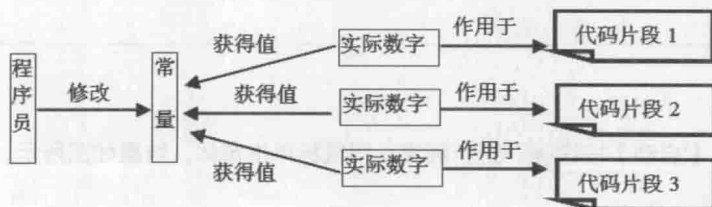
不, 常量的用途自有它的微妙之处。

(1) 如果直接使用数值来编写表达式, 计算结果当然不会有任何问题。但是如果使用的这个数值变了, 比如以前是 10, 现在变成 20 了, 那么就要在这个程序中将所有使用到这个数值的地方都修改掉。如果我们事先定义一个常量, 然后在所有使用这个数值的地方都使用这个常量, 那么只需要把这个常量的值从 10 改成 20 就可以了。

(2) 使用常量能够增强程序的可读性。如果不用一些有实际意义的名字来代替那些数字, 很难说清楚在程序中某个地方的数字 10 和另外一个地方的数字 10 有什么区别, 究竟各是什么意思。二者的不同之处如下面两个图所示。



不定义常量的修改方法



## 定义常量的修改方法

Visual Basic 中的常量有两种形式：直接常量和符号常量。

### 1. 直接常量

直接常量实际上就是所赋的值为基本类型的常量，主要有字符串常量、数值常量、布尔常量，以及日期常量等。

#### (1) 字符串常量

字符串常量是用双引号括起来的，除了双引号、回车和换行符之外的所有字符定义的常量。如果双引号内没有任何字符，并不表示这个常量不存在，只是表明这是一个内容为空的字符串，我们把它叫作空字符串。

#### (2) 数值常量

数值常量就是值为数值的常量。常见的 5 种数值常量有整数常量、长整型常量、定点数常量、浮点数常量和字节数常量。

#### (3) 布尔常量

布尔常量就是取值为布尔值的常量，即只有 True 和 False 两个取值的常量。

#### (4) 日期常量

日期常量是所取值为表示日期时间的值的常量。

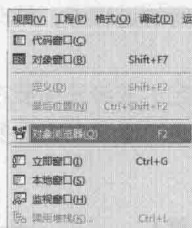
### 2. 符号常量

符号常量分为系统内部定义常量和用户自定义常量两种。

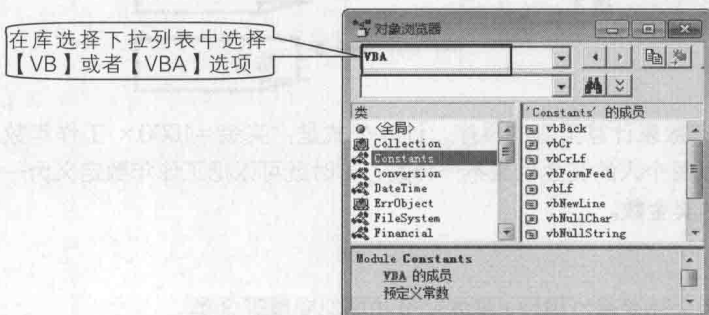
#### (1) 系统内部定义常量

在 Visual Basic 中内置了一些经常使用的常量，通常这些常量以字母“vb”作为前缀，如 vbCalendar、vbCallType 等。

① 打开 Visual Basic 开发环境，按【F2】快捷键或者选择【视图】菜单中的【对象浏览器】菜单项，打开【对象浏览器】窗口。



② 在库选择下拉列表中选择【VB】或者【VBA】选项，就可以看到很多的系统内部定义常量。



## (2) 用户自定义常量

如果用户觉得系统内部定义常量不能满足自己的需求, 可以通过一定的规则创建属于自己的符号常量。在 Visual Basic 中, 可使用 Const 语句创建一个符号变量。

语法:

```
[Private|Public] Const <常量名> [As 类型] = <表达式>
```

语句前面的 Private 或 Public 是定义的这个常量的作用域, 其中 Private 表示定义的常量只能在该常量所在模块内使用, 而 Public 则表示定义的常量可以被该常量所在模块之外的其他模块使用。

常量名: 指定常量名称

Const: 常量定义关键字。

As: 指定常量的类型。

表达式: 指定常量定义的数值常数、字符串常数或者由运算符组成的表达式。如果是数值常数, 不仅可以使十进制, 还可以使用十六进制 (数值前加 “&H”) 和八进制 (数值前加 “&O”)。

例如:

```
Const s As String=" 木木 " '定义字符型常量
```

```
Const s As Integer=100 '定义整型常量
```

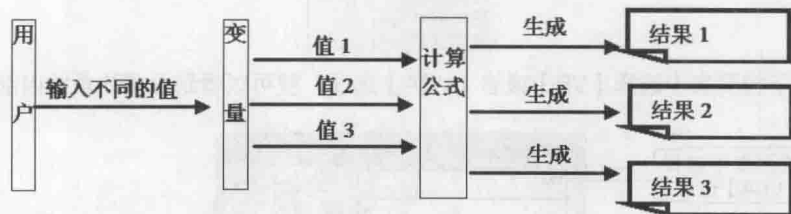
## 【语法详解】

常量不同于变量, 一旦定义了一个常量, 这个常量就会一直占据一定的内存空间。从程序执行效率的角度考虑, 不要在程序中定义不需要使用的常量。

## 2.2.2 变量

在 Visual Basic 中, 变量是指在程序运行的过程中, 具有特定类型的可以改变的数字、字符或字符串。

在我们编写程序时, 有时某个值是不确定的或者是在程序运行中由用户指定的, 这时我们可以通过定义一个变量来写出运算表达式, 而不必知道这个值是多少。



例如要编写一个根据员工工作年数来计算奖金的程序, 计算公式是: 奖金 = 1000 × 工作年数。因为每个人的工作年数是不同的, 所以每个人的奖金也是不一样的, 这时就可以把工作年数定义为一个变量, 不同的工作年数会计算出不同的奖金数。

### 1. 变量的类型

根据变量的作用范围, 可以分为全局变量、模块 / 窗体变量和局部变量等 3 类。

### (1) 全局变量

在整个程序中都可以使用的变量。定义一个全局变量的语法为：

---

Public 变量名 [As 类型]

---

### (2) 模块 / 窗体变量

作用范围比全局变量要小一些。模块 / 窗体变量可以在变量所在的模块或窗口中使用。在程序的“通用声明”部分使用 Dim 或者 Private 语句就可以定义一个模块 / 窗体变量。定义一个模块 / 窗体变量的语法为：

---

Private 变量名 [As 类型]

---

### (3) 局部变量

局部变量的作用范围最小，只能在变量所在的过程中使用。在程序的某个过程中使用 Dim 或者 Static 语句就可以定义一个局部变量。局部变量虽然作用范围比较小，但是在自己的作用范围内绝对是“地头蛇”。如果在某个局部变量外部有一个和它同名的变量，在该局部变量的作用范围内起作用的将是这个局部变量。

## 2. 如何声明一个变量

使用 Dim 或 Static 语句进行显式声明。

### (1) 使用 Dim 语句声明

语法：

---

Dim 变量名 [As 类型]

---

使用 Dim 语句声明的变量作用范围取决于 Dim 语句所在的位置。如果 Dim 语句是在某个过程内，那么声明的变量就是该过程中的局部变量；如果 Dim 语句是在某个窗体或模块的“通用声明”部分，那么声明的变量就是一个模块 / 窗体变量。

### (2) 使用 Static 语句声明

语法：

---

Static 变量名 [As 类型]

---

使用 Static 定义的变量叫作静态变量。一个变量除了有它的作用范围之外，还有一个属性就是它的生命周期或者叫作存活时间。一旦一个变量的生命周期结束，那么这个变量就不存在了，它所占据的内存空间也会被释放。一个局部变量如果不是静态的，那么它所在过程的代码执行完毕这个变量的生命周期就结束了，再次执行这段代码的时候，这个变量就会重新开始一个新的生命周期。静态变量所在过程的代码执行完毕生命周期并不会结束。静态变量和非静态变量的区别就好比买房和租房，如果房子是自己买的，那么即使外出度假时房子空闲也不会有人占有这个房子；而租房，租期满了如果不续租，房子就会被房东收回，如果续租就会重新开始一个新的租期。下面通过一段代码来看看静态变量和非静态变量的区别。

---

```
Private Sub Command1_Click()
    Static m As Integer ' 定义一个静态变量
    Dim n As Integer    ' 定义一个内部变量
```

---

```

m = m + 1    ' 让静态变量值加 1
n = n + 1    ' 让内部变量值加 1
End Sub

```

此段代码在第 1 次执行的时候, 因为变量  $m$  和  $n$  都没有被赋值, 所以  $m$  和  $n$  都是它们的默认初始值 0。运行结束后  $m$  和  $n$  的值都是 1。

当此段代码第 2 次被执行的时候, 因为  $m$  是静态变量, 所以  $m$  的生命周期并没有结束, 它的值还是第 1 次执行后的 1。因为  $n$  不是静态变量, 所以第 1 次执行结束它的生命周期就结束了, 当第 2 次执行的时候, 它会重新开始一个新的生命周期,  $n$  会被再次赋予默认初始值 0。所以第 2 次执行后,  $m$  为 2,  $n$  为 1。

隐式声明就是不经声明直接使用, 此时 Visual Basic 将会根据所赋予的值自动设置它的类型。例如:

```

x = 1        'x 为整型
Name = "cat" 'Name 为字符串型

```

隐式声明看起来似乎更加方便, 但隐式声明出现的问题往往是最棘手的, 因为它产生的问题很难被发现, 而且编译器也不会报错。隐式声明最大的问题恰恰源自它的优点: 不用经过声明可以在代码中直接使用。我们来看下面的代码。

```

Private Sub Command1_Click()
cat = 6      ' 为没有声明的变量 cat 赋值
n = car      ' 为没有声明的变量 n 赋值 car
End Sub

```

首先, 我们不经声明直接使用了变量  $cat$  并将它赋值为 6, 然后想把  $cat$  的值也就是 6 赋给变量  $n$ , 可是不小心把  $cat$  写成了  $car$ , 程序就会认为我们不经声明又直接使用了一个变量  $car$ , 于是程序就新建了一个变量  $car$ , 并用默认值给这个新建的变量赋值。

这时候变量  $n$  的值已经不是 6 了, 如果再使用变量  $n$  进行各种计算, 就会出现很多出乎意料的结果, 而程序依然能够运行, 并且不会报错。



#### 注意

在编程时应尽量少用隐式声明, 以免产生程序在结果有误时仍可以运行的情况。

使用类型说明符直接声明一个变量的语法为:

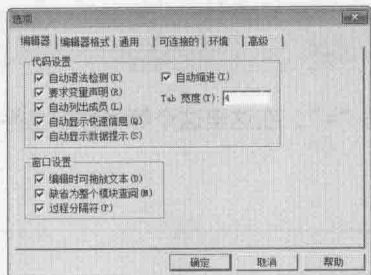
< 变量名 > 类型符

在使用一个变量之前并不必先声明这个变量, Visual Basic 会用变量的名字自动创建一个变量。

使用隐式声明的变量很危险, 我们可以使用强制显式声明来解决这个问题。一旦开启了强制显式声明功能, 如果没有声明一个变量就使用, 编译器就会报错。可以通过以下两种方式开启强制显式声明。

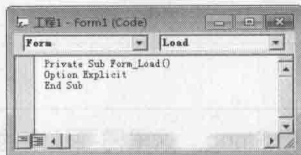
(1) 通过【选项】对话框设置强制显式声明

选择【工具】菜单下的【选项】菜单项, 弹出【选项】对话框, 选择【编辑器】选项卡, 选中【要求变量声明】复选框, 即可打开强制显式声明功能。



## (2) 代码声明

在代码窗口中的声明部分手动输入强制显式声明变量语句 (Option Explicit)。



## 2.3 运算符



本节视频教学录像：12 分钟

运算符就是在 Visual Basic 中执行某种运算功能的符号。在 Visual Basic 中包括算术运算符、赋值运算符、关系运算符、逻辑运算符、连接运算符和特殊运算符等。

### 2.3.1 算术运算符

Visual Basic 提供有非常丰富的算术运算符，使用这些运算符能够实现一些非常复杂的数学运算。下表按照运算优先级高低顺序列出了 Visual Basic 中的算术运算符及其功能。

运算符	含义	优先级	示例	结果
+	加法运算符	6	6+1	7
-	减法运算符	6	x =6 ; x-1	5
Mod	求模运算符，所得结果为两数相除的余数	5	x =4 ; 5 Mod x	1
\	整除运算符，所得结果为实际结果的整数部分	4	x =3 ; 5\ x	1
*	乘法运算符	3	x =4 ; x *2	8
/	除法运算符	3	x =4 ; 8/ x	2
-	负号运算符	2	x =6 ; -x	-6
^	指数运算符	1	x=3 ; x^2	9



### 2.3.2 赋值运算符

Visual Basic 中的赋值运算符是“=”，在这里这个等号并不是平时数学运算中的等号，它表示将等号右边的值赋给等号左边的变量。

赋值运算表达式是：

< 变量名 > = < 要赋的值 >

变量名：可以是变量、数组或者对象的某个属性。

要赋的值：可以是常数、变量、表达式、函数或者对象的属性等，但是它必须有确定的值。

例如：

String = " 我爱 VB"

表示将字符串“我爱 VB”赋给变量 String。这里需要注意的是，在赋值的时候，等号两边的数据类型要匹配。

### 2.3.3 关系运算符

关系运算符又叫作比较运算符。关系运算是通过比较两个表达式之间的关系，最终返回一个布尔值类型的运算结果。关系运算符之间的优先级是相同的。下表列出了 Visual Basic 中的关系运算符。

运算符	名称	示例	说明
=	等于	1 = 2	值为 False
<>	不等于	"cat" <> "DOG"	值为 True
>	大于	7 > 8	值为 False
>=	大于或等于	"cat" >= "bird"	值为 False
<	小于	9 < 12	值为 True
<=	小于或等于	6 <= 6	值为 True



提示

如果参与比较的两个变量或表达式都是数值的话，就直接比较它们的大小；如果参与比较的两个变量或表达式都是字符的话，则将字符转换为 ASCII 码并比较这两个 ASCII 码值。

### 2.3.4 逻辑运算符

在 Visual Basic 中提供有进行逻辑运算的运算符。逻辑运算符的优先级比算术运算符和关系运算符

要低。下表列出了 Visual Basic 中的逻辑运算符。

运算符	含义
And	与运算符。两个操作数同时为真时，结果为真
Or	或运算符。两个操作数中有一个为真时，结果为真
Not	非运算符。操作数为真时，结果为假；反之，结果为真
Xor	异或运算符。两个操作数为一真一假时，结果为真
Eqv	等价运算符。两个操作数相等时，结果为真
Imp	蕴涵运算符。两个操作数中，第 1 个为真，第 2 个为假，结果为假，其他各情况均为真

例如：逻辑值的运算如下表所示。

X 值	Y 值	X And Y	X Or Y	X Xor Y	X Eqv Y	X Imp Y	Not X
True	True	True	True	False	True	True	False
True	False	False	True	True	False	False	False
False	False	False	False	False	True	True	True
False	True	False	True	True	False	True	True

### 2.3.5 连接运算符

字符串连接运算符的作用是把两个字符串合并连接为一个字符串。Visual Basic 有两个字符串连接运算符，分别是“&”和“+”，它们的优先级是相同的。使用“&”运算符时，如果两个字符都是字符串，则直接将两个字符串连接成一个新的字符串；如果两个字符中有一个是数值，Visual Basic 自动将这个数值转换成为字符串，然后再连接成一个新字符串。

“+”运算符在参与计算的两个表达式都是字符串的时候和“&”运算符的功能是一样的，如果参与计算的另一个表达式中有一个是数值或者两个都是数值，将进行加法操作。因此，如果要做两个字符串的连接操作，最好使用“&”运算符，这样就不会为到底进行的是加法操作还是字符串连接操作而费脑筋了。

### 2.3.6 特殊运算符

除了以上运算符外，在 Visual Basic 中还有两个特殊运算符：Is 和 Like。

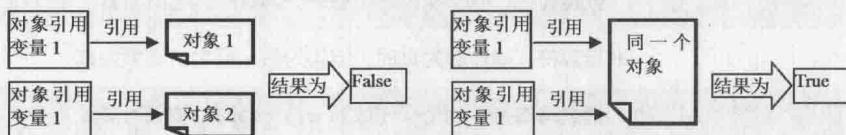
(1) Is 运算符

比较两个对象的引用变量。

语法:

结果 = 对象引用变量 1 Is 对象引用变量 2

如果对象引用变量 1 和对象引用变量 2 引用的是同一个对象, 运算结果为 True, 否则运算结果为 False, 如图所示。



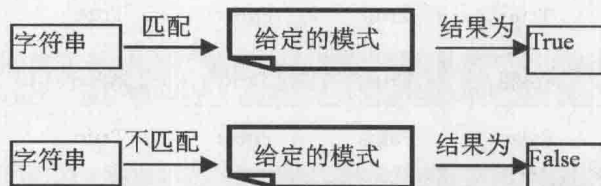
(2) Like 运算符

比较给定字符串是否和指定的模式相匹配。

语法:

结果 = 字符串 Like 模式

如果字符串匹配给定的模式, 结果返回 True, 否则返回 False, 如图所示。



下面列出了模式中的各种匹配条件。

- (1) ? 表示任意一个字符。
- (2) \* 表示任意多个字符。
- (3) # 表示任意一个数字 (0~9)。
- (4) [chars] 表示 chars 中出现的任意一个字符。
- (5) [!chars] 表示要匹配的字符串中不能出现 chars 中的任意一个字符。

例如:

result = "a" Like "[a-z]"

运算结果为 True, 因为字符 "a" 是从 a 到 z 之间的一个字符。

### 2.3.7 运算符的优先级

运算符的优先级是指在计算表达式时运算符执行的先后顺序。具有较高优先级的运算符先于较低优先级的运算符执行。然而在实际开发中, 我们要尽量避免使用运算符的优先级来指定运算的顺序, 更好的方法是把要优先计算的部分用圆括号括起来。例如表达式  $a + b * c$ , 按照优先级顺序是要先计算  $b * c$  的值, 然后再和  $a$  相加。如果要执行  $a + b$  的计算, 只需将表达式改为  $(a + b) * c$  就可以了。使用圆括号来指定优先级有以下两个优点。

(1) 不再需要记忆复杂的优先级顺序，也不用担心由于优先级顺序记忆错误导致的计算错误。

(2) 表达式更容易被人阅读。在实际开发过程中，一个软件往往需要多人共同完成，即使你能很准确地记忆和使用运算符的优先级，并不能保证别人也能很准确地记忆和使用运算符的优先级，所以使用圆括号指定程序运算表达式的优先级无疑是一种更加聪明和高效的方式。

例如：

```
S = (30-5<(8+6)And 9*8=72)
```

一个表达式中如果包含了多种运算符，那么该运算符应该按以下次序进行：算术表达式 > 字符串表达式 > 关系表达式 > 逻辑表达式。因此，上例中的运算结果为 Flase。

## 2.4 表达式



本节视频教学录像：3 分钟

表达式是由常量、变量和函数等用运算符及圆括号连接而成的式子。表达式是构成程序代码的最基本要素之一。根据表达式的运算结果，可将其分为算术表达式、字符串表达式和日期表达式等 3 种。

### 2.4.1 算术表达式

用算术运算符和括号将常量、变量或函数等运算对象连接起来的式子称为算术表达式。Visual Basic 中常用的算数运算符有 + (加)、- (减)、\* (乘)、/ (除)、\ (整除)、mod (模运算，求余数) 和 ^ (幂) 等。

### 2.4.2 字符串表达式

字符串表达式是用字符串运算符将字符串连接起来的表达式。在 Visual Basic 中有两个字符串运算符：“&” 和 “+”。使用 “&” 运算符时，如果两个字符都是字符串，则直接将两个字符串连接成一个新的字符串；如果两个字符中有一个是数值，Visual Basic 会自动将这个数值转换为字符串，然后再连接成一个新的字符串。“+” 运算符在参与计算的两个表达式都是字符串的时候，和 “&” 运算符的功能是一样的；但如果参与计算的另一个表达式中有一个是数值或者两个都是数值，则进行加法操作。

### 2.4.3 日期表达式

当我们需要计算或者使用时间和日期相关的数据时（例如，编写一个程序需要每天在固定的时间备份数据），就要用到日期表达式。日期表达式主要是对 Date (日期型) 数据类型进行计算。

## 2.5 代码编写规范



本节视频教学录像：9 分钟

代码编写规范在应用程序的开发过程中有着极为重要的作用，有利于程序员对程序进行读写，以及

方便后期维护,同时也是养成良好编程习惯的基础。

## 2.5.1 Visual Basic 6.0 标识符的定义规则

在 Visual Basic 中有两种标识符,分别是系统关键字和用户自定义标识符。系统关键字是 Visual Basic 中拥有固定含义、不能被重新定义的标识符。用户自定义标识符是用户自己为变量通过一定的规则所定义的名称。这就像我们中国人在起名字的时候要把姓放在前面,名字放在后面一样,是一个命名规则。在 Visual Basic 中用户自定义标识符的命名规则如下。

- (1) 用户自定义标识符不能和系统关键字相同。
- (2) 在同一个作用域内不允许出现相同名称的用户自定义标识符。
- (3) 用户自定义标识符不区分字母大小写。
- (4) 用户自定义标识符必须以字母开头,后跟数字、英文大小写字母、下划线或美元符号“\$”。
- (5) 长度不能超过 255 个字符,而控件、窗体、类和模块等的名字不能超过 40 个字符。

## 2.5.2 Visual Basic 6.0 中变量及控件的命名规则

变量必须要命名后才能赋值。在过程内部声明的变量,在该过程执行时才存在。当运行过程结束时,该变量的值自动清除。变量命名必须符合以下规定。

- (1) 变量名必须以字母开头,不能以数字或下划线开始。
- (2) 变量不能包含嵌入的句号、空格或者其他停顿符号。
- (3) 变量长度不得超过 255 个字符。
- (4) 变量名不能是 Visual Basic 的保留字,如 Name、For 等。
- (5) 在编码过程中,定义变量时选择的变量名一定要遵循上述规则,否则系统会给出错误提示并以红色字体显示错误行。

在 Visual Basic 中,控件以图标形式放在“工具箱”中,每种控件都有与之对应的图标。启动 Visual Basic,工具箱位于窗体的左侧。

### 1. 标准控件(内部控件) Visual Basic 6.0 的控件分为以下 3 类

- (1) 标准控件(也称内部控件)。
- (2) ActiveX 控件。
- (3) 可插入对象。

### 2. 控件的命名和控件值

(1) 在一般情况下,窗体和控件的命名都有默认值,如 Form1、Command1、Text1 等。在应用程序中使用约定的前缀,可以提高程序的可读性。

(2) 为了方便使用,Visual Basic 为每个控件规定了一个默认属性,在设置这样的属性时,不必给出属性名,通常把该属性称为控件的值。

## 2.5.3 程序书写规则

按照程序书写规则写出的代码,不仅清晰明了,更为重要的是具有很好的可读性。因此,在编写代码的时候要遵循一定的书写规则,如下所述。

- (1) 每条语句写一行，一行最多允许有 255 个字符。
- (2) 如果需要在同一行中书写多条语句，应用英文状态下的冒号“:”隔开。
- (3) 需要对较长的语句进行换行，可在该语句行的末尾加入一个空格和一个下划线。

例如：

```
If a > 0 and b < 0 And c <> 0 And _
d = 0 Then
```

如果进行换行时没有输入空格，系统会提示语法错误。例如在上面的语句中，如果没有输入空格，就会弹出一个警告对话框，提示编译错误。



- (4) 为代码添加注释，以方便后期检查或别人阅读。

(5) 在 Visual Basic 中不区分字母大小写，所以 cat、CAT 和 Cat 都是一样的，因此在定义变量的时候需要注意。

## 2.5.4 添加注释

优秀的程序员都具有在关键步骤下添加注释的好习惯。添加注释只是为了方便用户看懂代码而写的，目的是提高代码的可读性，其本身对程序并没影响。添加注释需要在 Visual Basic 一条语句的最后添加注释符（'）或关键字 Rem。注释符在程序运行时通知 Visual Basic 自动忽略注释符之后的语句。这样便于日后检查修改程序。

## 2.5.5 格式化缩排程序语句

为了进一步增加程序的可读性，建议程序员尽量养成程序缩排的好习惯。在书写程序时，可以采用 Tab 键手工对程序进行缩排，也可以通过编辑工具栏中的“凸出”和“缩进”命令进行编排。例如：没有采用缩排的代码。

```
01 Private Sub Command1_Click()
02 Dim i, j As Integer '定义两个整型变量
03 For i = 1 To 9      '用 For 循环打印 9 行内容
04 j = 1              '为变量 j 赋初值
05 Do While j <= 1    '用 Do 循环，打印每一列中的内容
06 Print i; " "; j; " = "; i * j,      '输出每一列的内容
07 j = j + 1          '增加行数
08 Loop              '满足条件，继续内部循环
09 Print              '打印空行，起换行作用
10 Next i            '满足条件，继续外部循环
```



```
11 End Sub
```

采用缩排的代码。

```
01 Private Sub Command1_Click()  
02   Dim i, j As Integer      '定义两个整型变量  
03   For i = 1 To 9          '用 For 循环打印 9 行内容  
04     j = 1 '为变量 j 赋初值  
05     Do While j <= 10      '用 Do 循环, 打印每一列中的内容  
06       Print i; "**"; j; "="; i * j, '输出每一列的内容  
07       j = j + 1          '增加行数  
08     Loop                  '满足条件, 继续内部循环  
09   Print '打印空行, 起换行作用  
10 Next i '满足条件, 继续外部循环  
11 End Sub
```

对没有采用缩排格式的代码来说, 读起来有点找不到“北”的感觉, 在没有添加注释的情况下, 只有一句一句进行分析才明白代码的最终目的是什么。而对采用缩排格式的代码来说, 很清晰地就能分析出来, 代码中主要分为内部循环和外部循环, 其中内循环的作用和外循环的作用也一眼即可明了。

## 2.6 高手点拨



本节视频教学录像: 2 分钟

对于从未接触过 Visual Basic 的初学者来说, 要想快速学好 Visual Basic, 就要从 Visual Basic 的语言基础开始循序渐进地学习。在 Visual Basic 的语言基础学习过程中要注意以下内容。

- (1) 标识符的定义规则。
- (2) 各种数据类型特点和作用。
- (3) 变量和常量的应用。
- (4) 运算符的作用和优先级。

作为初学者, 只有对上述几点做到了然于胸, 才能读懂和编写基本的程序。

## 2.7 实战练习

### 一、思考题

1. 用户自定义标识符的命名规则是什么?
2. 使用圆括号来指定优先级时有哪些优点?

### 二、上机题

交换两个变量的值, 写出相应的语句。

# 第3章



本章视频教学录像：1 小时 2 分钟

## Visual Basic 的秘密 ——算法和程序控制结构

算法是问题求解过程的精确描述，因此掌握算法是学习程序设计的核心，它可以帮助用户更好、更快地掌握编程思想及编程方法。程序的执行就像水在管道中流动，如果不加以控制，只能从上流到下。任何一种程序设计语言都有自己的算法和程序控制结构，但大致结构都相似。在 Visual Basic 中，控制程序执行的基本结构有 3 种：顺序结构、选择结构和循环结构。本章将详细介绍这 3 种结构的程序控制语句。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解算法基本概念、特性及算法的几种描述方法
- ☐ 了解结构化程序设计
- ☐ 掌握程序的顺序结构
- ☐ 掌握程序的选择结构
- ☐ 掌握程序的循环结构
- ☐ 了解其他辅助控制语句

## 3.1 算法



本节视频教学录像: 3 分钟

算法是学习程序设计的基础,也可以说是程序设计的入门知识,掌握算法可以帮助读者快速理清程序设计的思路,找出多种解决问题的方法,从而选择最合适的解决方案。

### 1. 什么是算法

“算法”这个术语听起来可能很陌生,其实大多数人每天都会用到许多算法。

我们早晨坐车上上班,一般情况下会坐公交车上班,但如果时间来不及或遇到其他特殊情况,可能会打车上班,这就是一个“算法”。因此,广义地讲,“算法”就是解决某个问题或处理某件事的方法和步骤。

### 2. 算法的特性

一个算法应该具有以下 5 个主要特性。

- (1) 有穷性: 一个算法(对任何合法的输入)在执行有穷步后能够结束,并且在有限的时间内完成。
- (2) 确定性: 算法中的每一步都有确切的含义。
- (3) 可行性: 算法中的操作能够用已经实现的基本运算执行有限次来实现。
- (4) 输入: 一个算法有零个或者多个输入,零个输入就是算法本身确定了初始条件。
- (5) 输出: 一个算法有一个或多个输出,以反映出数据加工的结果,没有输出的算法是没有意义的。

### 3. 算法的描述方法

为了让算法清晰易懂,需要选择一个好的描述方法。算法的描述方法有很多,有自然语言、伪代码、传统流程图等。

#### (1) 自然语言

自然语言就是用人们日常使用的语言描述解决问题的方法和步骤,这种描述方法通俗易懂,即使是不熟悉计算机语言的人也很容易理解程序。但是,自然语言在语法和语义上往往具有多义性,并且比较繁琐,对程序流向等描述不明了、不直观。

#### (2) 伪代码

伪代码是介于自然语言和计算机语言之间的文字和符号,它与一些高级编程语言(如 Visual C++) 类似,但是没有真正编写程序时所要遵循的严格规则。伪代码用一种从顶到底、易于阅读的方式表示算法。在程序开发期间,伪代码经常用于“规划”一个程序,然后再转换成 VB 程序。

#### (3) 传统流程图

传统流程图使用不同的几何图形来表示不同性质的操作,使用流程线来表示算法的执行方向。比起前两种描述方式,它具有直观形象、逻辑清晰、易于理解等特点,但占用篇幅较大,流程随意转向,较大的流程图不易读懂。

## 3.2 结构化程序设计



本节视频教学录像: 4 分钟

结构化程序设计(Structured Programming)是进行以模块功能和处理过程设计为主的详细设计的基本原则。

## 1. 概念

其概念最早由 E.W.Dijkstra 在 1965 年提出的,是软件发展的一个重要的里程碑。它的主要观点是采用自顶向下、逐步求精及模块化的程序设计方法。使用三种基本控制结构构造程序,任何程序都可由顺序、选择、循环三种基本控制结构构造。结构化程序设计主要强调的是程序的易读性。

## 2. 内容

结构化程序设计曾被称为软件发展中的第三个里程碑。该方法的要点是:

(1) 主张使用顺序、选择、循环三种基本结构来嵌套连结成具有复杂层次的“结构化程序”,严格控制 GOTO 语句的使用。用这样的方法编出的程序在结构上具有以下效果。

① 以控制结构为单位,只有一个入口,一个出口,所以能独立地理解这一部分。

② 能够以控制结构为单位,从上到下顺序地阅读程序文本。

③ 由于程序的静态描述与执行时的控制流程容易对应,所以能够方便正确地理解程序的动作。

(2) “自顶而下,逐步求精”的设计思想,其出发点是从问题的总体目标开始,抽象低层的细节,先专心构造高层的结构,然后再一层一层地分解和细化。这使设计者能把握主题,高屋建瓴,避免一开始就陷入复杂的细节中,使复杂的设计过程变得简单明了,过程的结果也容易做到正确可靠。

(3) “独立功能,单出、入口”的模块结构,减少模块的相互联系使模块可作为插件或积木使用,降低程序的复杂性,提高可靠性。程序编写时,所有模块的功能通过相应的子程序(函数或过程)的代码来实现。程序的主体是子程序层次库,它与功能模块的抽象层次相对应,编码原则使得程序流程简洁、清晰,增强可读性。

## 3. 基本结构

结构化程序设计的三种基本结构是:顺序结构、选择结构和循环结构。

### (1) 顺序结构

顺序结构表示程序中的各操作是按照它们出现的先后顺序执行的。

### (2) 选择结构

选择结构表示程序的处理步骤出现了分支,它需要根据某一特定的条件选择其中的一个分支执行。选择结构有单选择、双选择和多选择三种形式。

### (3) 循环结构

循环结构表示程序反复执行某个或某些操作,直到某条件为假(或为真)时才可终止循环。在循环结构中最主要的是:什么情况下执行循环,哪些操作需要循环执行。循环结构的基本形式有两种:当型循环和直到型循环。

当型循环:表示先判断条件,当满足给定的条件时执行循环体,并且在循环终端处流程自动返回到循环入口;如果条件不满足,则退出循环体直接到达流程出口处。因为是“当条件满足时执行循环”,即先判断后执行,所以称为当型循环。

直到型循环:表示从结构入口处直接执行循环体,在循环终端处判断条件,如果条件不满足,返回入口处继续执行循环体,直到条件为真时再退出循环到达流程出口处,是先执行后判断。因为是“直到条件为真时为止”,所以称为直到型循环。

## 4. 设计方法

### (1) 自顶向下

程序设计时,应先考虑总体,后考虑细节;先考虑全局目标,后考虑局部目标。不要一开始就过多追求众多的细节,先从最上层总目标开始设计,逐步使问题具体化。

## (2) 逐步细化

对复杂问题, 应设计一些子目标作为过渡, 逐步细化。

## (3) 模块化

一个复杂问题, 肯定是由若干稍简单的问题构成。模块化是把程序要解决的总目标分解为子目标, 再进一步分解为具体的小目标, 把每一个小目标称为一个模块。

## 5. 优缺点

### (1) 优点

由于模块相互独立, 因此在设计其中一个模块时, 不会受到其他模块的牵连, 因而可将原来较为复杂的问题化简为一系列简单模块的设计。模块的独立性还为扩充已有的系统、建立新系统带来了不少的方便, 因为我们可以充分利用现有的模块做积木式的扩展。

按照结构化程序设计的观点, 任何算法功能都可以通过由程序模块组成的三种基本程序结构的组合: 顺序结构、选择结构和循环结构来实现。

结构化程序设计的基本思想是采用“自顶向下, 逐步求精”的程序设计方法和“单入口单出口”的控制结构。自顶向下、逐步求精的程序设计方法从问题本身开始, 经过逐步细化, 将解决问题的步骤分解为由基本程序结构模块组成的结构化程序框图; “单入口单出口”的思想认为一个复杂的程序, 如果它仅是由顺序、选择和循环三种基本程序结构通过组合、嵌套构成, 那么这个新构造的程序一定是一个单入口单出口的程序。据此就很容易编写出结构良好、易于调试的程序来。

① 整体思路清楚, 目标明确。

② 设计工作中阶段性非常强, 有利于系统开发的总体管理和控制。

③ 在系统分析时可以诊断出原系统中存在的问题和结构上的缺陷。

### (2) 缺点

① 用户要求难以在系统分析阶段准确定义, 致使系统在交付使用时产生许多问题。

② 用系统开发每个阶段的成果来进行控制, 不能适应事物变化的要求。

③ 系统的开发周期长。

## 3.3 顺序结构



本节视频教学录像: 1 分钟

顺序结构是指程序按照语句出现的先后次序执行。可以把顺序结构想像成一个没有分支的管道, 把数据想像成水流, 数据从入口进入后, 依次执行每一条语句直到结束。流程如图所示。



在 Visual Basic 中赋值语句、注释语句、输入语句、输出语句、变量定义语句等都属于顺序结构功能的语句。这些语句本身没有控制和改变程序结构的能力, 它们在语句中出现的顺序就是执行的顺序。

### 3.3.1 赋值运算符

赋值运算符用来为变量或事件的属性赋值。程序的计算实质上是数据的计算，而数据依存于各种变量、属性值等里面，我们通过赋值运算符把数据和变量、属性值等关联起来从而进行实际计算。赋值运算符的常见语法形式是：

变量名 = 表达式

对象名.属性 = 表达式

赋值运算符中的等号并不是表示等号两边的值是相等的，而是将等号右边的值赋给等号左边的变量或属性。如果等号右边是一个表达式，则会先执行这个表达式，执行完毕将执行结果赋给等号左边的变量或属性。

下面两个是正确的赋值运算符表达式。

`a = a + 1` '把数值变量 a 加 1 后赋给变量 a

`Command1.Caption = "确定"` '把字符串“确定”赋给 Command1 的 Caption 属性

下面两个是错误的赋值运算符表达式。

`6 = a + b` '错误原因：赋值符号左边不能是常量

`a + b = 6` '错误原因：赋值符号左边不能是表达式

### 3.3.2 数据的输入与输出

#### 1. 数据的输入

Visual Basic 中的数据输入主要通过文本框控件、列表框、组合框、复选框等控件以及 InputBox 函数来实现。控件的使用在后续篇章中将详细介绍，这里主要介绍 InputBox 函数的用法。InputBox 函数是用来接收用户通过键盘输入的数据。调用 InputBox 函数的格式如下。

`Var=InputBox (<提示>[,对话框标题][,编辑框中默认值][,x坐标][,y坐标][,帮助文件名,帮助主题号])`

功能：打开一个对话框，等待用户输入内容，当用户单击【确定】按钮或按【Enter】键，函数返回输入的值，其值的类型为字符串。参数说明如下。

(1) 提示：不能省略该项。字符串表达式，在对话框中作为信息显示，可为汉字；若要多行显示，必须在每行行末加回车 Chr(13) 和换行 Chr(10) 控制符或 vbCrLf 符号常数。

(2) 对话框标题：字符串表达式，在对话框的标题区显示；若省略，则把应用程序名放入标题栏中。

(3) 编辑框中默认值：字符串表达式，当输入对话框中无输入时，则该默认值作为输入的内容。

(4) x 坐标、y 坐标：整型表达式，坐标确定对话框左上角在屏幕上的位置，屏幕左上角为坐标原点。

例如：下面程序实现从键盘接收一个字符串，并将其显示在窗体上。

```
Private Sub Form_Click()
```

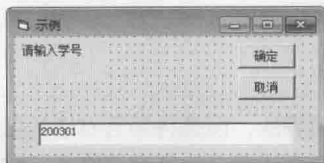


```

Dim num As String
num = InputBox("请输入学号", "示例")
Print num
End Sub

```

上述程序将用户的输入显示在窗体上，运行效果如图所示。



InputBox 函数接收输入



InputBox 函数显示输入

当用户输入完成后单击左图中的【确定】按钮，表示输入成功，用户输入的数据将被赋值到变量 num 中，使用 Print 函数可以显示在窗体上。如果单击【取消】按钮，则输入失败。

## 2. 数据的输出

Visual Basic 中，数据的输出主要有 Print 方法、MsgBox 函数和控件输出 3 种方式。一般来说，Print 方法主要输出在特定控件上，例如 Form.Print，即输出在窗体上；MsgBox 函数则以对话框的形式输出信息；而控件输出主要是指输出在数据控件上，例如 Label1.Caption="Visual Basic"。下面通过示例看一下这 3 种输出的形式。例如，用 Print 方法、MsgBox 函数和控件输出 3 种方式输出 a 中的数据。

在窗体上使用 Print 方法输出数据的输出格式为：

Print 变量

使用 MsgBox 函数的简单输出格式为：

MsgBox 变量

使用 Label 控件的输出即使用其 Caption 标题属性，使用格式为：

Label1.Caption = 变量

这 3 种方式在后续的章节中都将详细介绍。

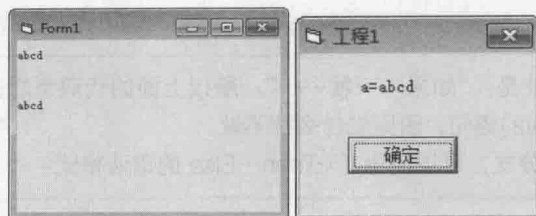
该示例的实现代码如下。

```

Private Sub Form_Click()
Dim a As String
a = InputBox("请输入 a 的值", "示例")
Print a 'Print 方法输出
MsgBox "a=" & a 'MsgBox 函数输出
Label1.Caption = a '标签控件输出
End Sub

```

该程序执行效果如图所示。



Print 方法输出

标签控件输出

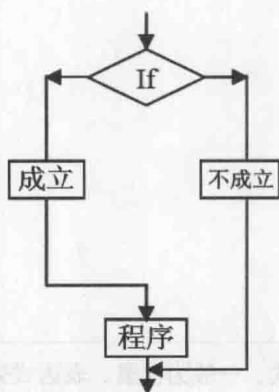
程序执行时，先由用户在输入框 InputBox 中输入 a 的值，再分别用 Print 方法、MsgBox 输出和 Label 控件输出在窗体上。控件的输出将结合具体的控件，除了 Label 标签控件外，还有许多如 TextBox 文本框输出、Image 图片框输出等，这些将在以后的章节中详细讲解，此处只介绍前两种方法。

## 3.4 选择结构

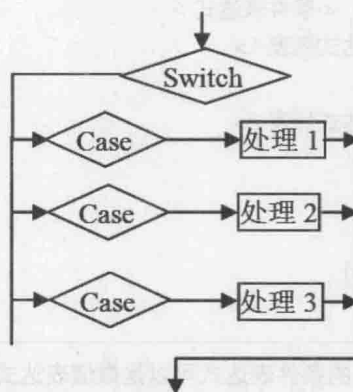


本节视频教学录像：16 分钟

选择结构是通过对给定的条件进行判断，然后根据判断结果执行不同任务的一种程序结构。Visual Basic 中的 If 条件语句和多分支条件语句（Select）是最为常用的两种条件语句。条件语句和多分支条件语句的流程如图所示。



条件语句流程图



多分支条件语句流程图

### 3.4.1 If 条件语句

If 语句的常见语法格式是：

If 条件表达式 Then

语句 1

语句 2

...

End If

If...Then...翻译成中文就是：“如果……就……”。所以上面的代码意思是如果 If 后面的条件表达式成立的话，就执行 Then 下面的语句，否则就什么都不做。

如果希望再加一个选择分支，可以使用 If...Then...Else 的语法格式：

If 条件表达式 Then

语句 1

Else

语句 2

End If

加上 Else 以后的代码意思是如果 If 后面的条件表达式成立的话，就执行 Then 下面的语句，否则执行 Else 后面的语句。

### 3.4.2 Select case 语句

If 语句主要用于单分支或者双分支的判断性选择结构，虽然也可以用形如 If...Then...Elseif...Elseif...Else 这样的语法来实现多分支的选择结构，但是这显然太麻烦了，Select 语句是更为方便和有效的一种办法。Select 语句的语法是：

Select Case < 条件表达式 >

Case < 表达式列表 1>

< 语句块 1>

Case < 表达式列表 2>

< 语句块 2>

...

[Case Else

< 语句块 n>]

End Select

上述语句中的条件表达式可以是数值表达式或字符串表达式，一般为变量。表达式列表则是表明条件表达式的可能取值，表达式列表有以下 3 种形式。

(1) 逗号分隔的多个表达式。例如：

Case 1,2,3 ' 表达式的值是 1、2 或者 3


(2) < 表达式 1> To < 表达式 2>。例如：



Case 1 to 10 ' 表达式的值介于 1 到 10 之间

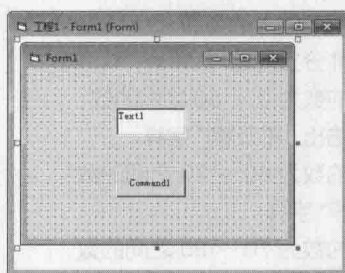
(3) Is < 关系操作符 > < 表达式 1>。例如：

Is <10 ' 表达式的值小于 10

### 【范例 3-1】在文本框中输入一个数值型数字，单击【Command】按钮后，判断该数字的奇偶性。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 用鼠标双击工具箱中的【TextBox】按钮 ，在 Form1 窗体上添加一个文本框控件，并用鼠标双击工具箱中的【CommandButton】按钮 ，添加一个按钮控件。



(3) 双击 Command1 控件，进入代码窗口，输入以下代码（代码 3-1.txt）。


```
01 Private Sub Command1_Click()  
02 Dim a As Integer ' 定义一个整型变量  
03 a = Val(Text1.Text) ' 得到输入的整数  
04 If a Mod 2 = 0 Then ' 判断能否被 2 整除  
05     Print a & " 是偶数 " ' 可以被 2 整除是偶数  
06 Else  
07     Print a & " 是奇数 " ' 不可以被 2 整除是奇数  
08 End If ' 退出条件判断  
09 End Sub
```

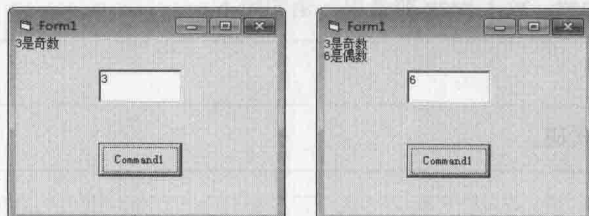


第 3 行使用了一个 Val 函数，Val 函数是将文本框中的字符串型的数字转换为整型。

**注意**

### 【运行结果】

保存程序，单击【启动】按钮 ，运行程序。清空文本框中的内容，输入数字“3”，单击【Command1】按钮，窗体上显示“3 是奇数”。清空文本框中的内容，输入数字“6”，单击【Command1】按钮，窗体上显示“6 是偶数”。



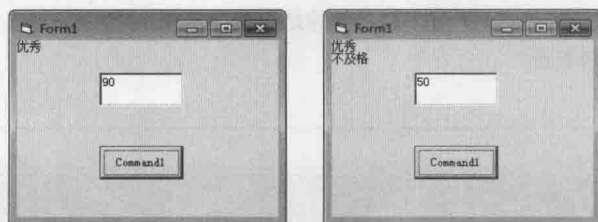
### 【拓展训练 3-1】

在文本框中输入学生的考试成绩,采用多分支条件语句对学生分数进行评定,判断其优良性。

学会了使用 If 条件语句的判断方法,可以尝试使用多分支条件语句来控制程序的运行。可以将【范例 3-1】中步骤(3)的代码修改如下(拓展代码 3-1.txt)。

```
01 Private Sub Command1_Click()
02   Dim x As Single, y As Single      ' 定义两个单精度类型的变量
03   x = Val(Text1.Text)              ' 获取输入的整数
04   Select Case x                     ' 进行多条件分支判断
05     Case 0 To 59                   ' 如果输入的数为 0 ~ 50 之间的数
06       Print "不及格"              ' 输出 "不及格" 字样
07     Case 60 To 70                  ' 如果输入的数为 60 ~ 70 之间的数
08       Print "中等"                ' 输出 "中等" 字样
09     Case 70 To 80                  ' 如果输入的数为 70 ~ 80 之间的数
10       Print "良好"                ' 输出 "良好" 字样
11     Case 81 To 100                 ' 如果输入的数为 81 ~ 100 之间的数
12       Print "优秀"                ' 输出 "优秀" 字样
13   End Select                       ' 结束判断
14 End Sub
```

运行后,清空文本框内容,输入“90”,单击【Command1】按钮,窗体上显示“优秀”。再次清空文本框内容,输入“50”,单击【Command1】按钮,窗体上显示“不及格”。



### 3.4.3 条件函数

在 Visual Basic 中,除了上述这几种分支语句外,还有两个条件函数值得读者注意: IIf 函数和 Choose 函数。

IIf 函数用于替代 If 函数,适用于简单的判断场合。其调用格式为: IIf (表达式, 当表达式的值为 True 时的值, 当表达式的值为 False 时的值)。其功能与 If...Then...Else 选择结构相同。

例如,求 x, y 中大的数,放入 max 变量中,语句如下。

```
max = IIf(x > y, x, y)
```

上述代码等同于下列代码。

```
If x > y Then
```

```
max=x
Else
max=y
End If
```

因此，IIF 函数可以作为 If...Then...Else 结构的简介形式来书写。

Choose 函数用于替代 Select Case 语句，也适用于简单的判断场合。其调用格式为：

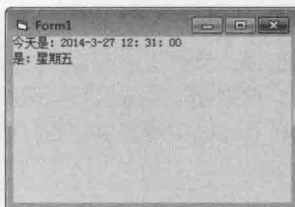
Choose( 整数表达式, 选项列表 )

功能为：根据整数表达式的值来决定返回选项列表中的某个值。如果整数表达式值是 1，则 Choose 会返回列表中的第 1 个选项；如果整数表达式值是 2，则会返回列表中的第 2 个选项，依此类推。若整数表达式的值小于 1 或大于列出的选项数目时，Choose 函数返回 Null。

例如，根据当前日期函数 Now、WeekDay，利用 Choose 函数显示今日是星期几的形式。

```
Private Sub Form_Click( )
Print "今天是: "; Now
t =Choose(Weekday(Now), "星期日", "星期一", "星期二", "星期三", "星期四", "星期五", "星期六")
Print" 是: "; t
End Sub
```

上述程序中，函数 Weekday ( Now ) 获取当前是星期几的一个数字，根据 Choose 函数返回具体的中文星期几显示。其执行效果如图所示。



上述代码等同于如下程序。

```
Private Sub Form_Click( )
Print" 今天是: "; Now
t =Weekday(Now)
Print t
Select Case t
Case 1
t = " 星期日 "
Case 2
t = " 星期一 "
Case 3
t = " 星期二 "
Case 4
```



```
t = "星期三"
```

```
Case 5
```

```
t = "星期四"
```

```
Case 6
```

```
t = "星期五"
```

```
Case 7
```

```
t = "星期六"
```

```
End Select
```

```
Print "是: "; t
```

```
End Sub
```

同样地, Choose 函数使 Select Case 语句的应用简单化了。在实际的使用中, 读者可根据情况自行选择使用哪种方式。

## 3.5 循环结构



本节视频教学录像: 15 分钟

当程序需要重复执行一些任务时, 就可以考虑采用循环结构。例如重复 100 次加减乘除, 数组中的每个整数加 1 等。当然可以为每次的任务都编写一条语句, 但是这样就违背了使用计算机的初衷: “使用计算机的目的就是为了减轻工作量, 而不是增加工作量。” 对于这些重复的任务, 可以使用 Visual Basic 中的循环结构很轻松地完成。



**提示**

Visual Basic 中的循环结构是指通过计算机来实现重复执行某任务的程序结构。循环结构可以减少源程序重复书写的工作量, 用来描述重复执行某段算法的问题, 这是程序设计中最能发挥计算机特长的程序结构, Visual Basic 中所提供的各种循环可以用来处理同一问题, 一般情况下它们可以互相替换。

### 3.5.1 For 循环语句

For 循环又叫作计数循环, 如果能够预知循环的次数, 就可以预先设定计数循环的所计数值。For 循环的语法是:

```
For 循环变量 = 初值 To 终值 [Step 步长]
```

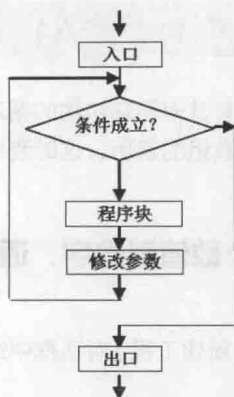
```
[ 语句块 ]
```

```
[Exit For]
```

```
[ 语句块 ]
```

```
Next [ 循环变量 ]
```

在 For 循环语句中, 循环变量、初值、终值和步长等都是数值型数据, 如果步长没有指定, 则默认值为 1。步长可以为正, 也可以为负。若为正, 则初值应小于或等于终值; 若为负, 则初值应大于或等于终值, 这样才能保证执行循环体内的语句; 若为 0, 则循环永不结束 (死循环)。For 循环语句的流程如图所示。



### 3.5.2 Do...Loop 循环语句

Do 循环分为先判断条件的 Do...Loop 循环和后判断条件的 Do...Loop 循环两种。先判断条件的 Do...Loop 循环的语法是：

Do While 条件表达式  
语句块  
Loop

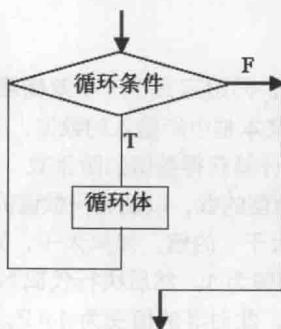
或者

Do Until 条件表达式  
语句块  
Loop

后判断条件的 Do...Loop 循环的语法是：

Do  
语句块  
Loop While 条件表达式


条件循环结构的流程如图所示。





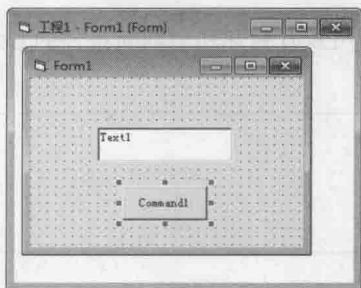
### 3.5.3 循环的嵌套

循环的嵌套,就是在一个循环体内含有其他循环结构的循环体。在实际问题中,有些单靠一个循环是无法很好解决的,例如对二维甚至多维数组的遍历,这时我们使用在循环中再嵌套循环的方式来实现一些复杂的逻辑。

#### 【范例 3-2】在文本框中输入一个数值型数字,通过使用计数循环结构方法,来计算所输入数值型数字的阶乘。

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标,然后单击【打开】按钮。

(2) 用鼠标双击工具箱中的【TextBox】按钮,在 Form1 窗体上添加一个文本框控件,并用鼠标双击工具箱中的【CommandButton】按钮,添加一个按钮控件。



(3) 双击 Command1 控件,进入代码窗口,输入以下代码(代码 3-2.txt)。

```
01 Private Sub Command1_Click()  
02   Dim i, x As Integer, s As Double ' 定义两个整型变量和一个双精度型变量  
03   x = Val(Text1.Text) ' 获取输入的数值  
04   s = 1 ' 给变量 s 赋初值 1  
05   For i = 1 To x ' 条件符合,利用循环结构进行计算  
06     s = s * i ' 累积  
07   Next i ' 进行下一轮循环  
08   Print x; " 的阶乘为 "; s; ' 输入最终结果  
09 End Sub
```

#### 【代码详解】

第 02 行定义了两个整型变量  $i$  和  $x$ ,同时定义了一个双精度型的变量  $s$ 。其中,变量  $i$  主要用于循环语句的初始值,变量  $x$  主要用于获得文本框中所输入的数值,变量  $s$  主要用于每一次计算后的结果。

第 05 ~ 07 行主要进行 For 循环来计算获得数值的阶乘数。例如在文本框中输入一个数值 3,那么代码执行到第 03 行时,将其转换为数值型的数,以便进行数值的计算。当代码执行到 For  $i=1$  To  $x$  时,此时  $x$  为 3,首先判断当前  $i$  的值是否大于  $x$  的值,如果大于,则停止循环,如果小于,则开始进行第 1 次循环,计算  $s=1\times 1$ ,此时  $s$  的结果值为 1。然后执行代码 Next  $i$  语句,将  $i$  的值加 1,并判断是否大于 3,如果不大于,执行第 2 次循环,此时  $s$  的值变为  $1\times 2$ ,得到最终的结果值为 2。然后再次将  $i$


的值加 1, i 的值变为了 3, 并判断当前 i 的值是否大于 3, 不大于继续执行循环体, s 的值变为了  $2 \times 3$ 。然后再次将 i 的值加 1, 此时 i 的值为 4, 已经大于 3, 所以退出循环体。执行最后一行代码语句, 将结果输出出来。



提示

在使用 For 循环时, 如果需要在代码执行过程中停止当前循环, 可以使用 Exit For 语句退出当前循环。

## 【运行结果】

保存程序, 单击【启动】按钮 , 运行程序。清空文本框中的内容, 输入数字“6”, 单击【Command1】按钮, 窗体上显示“6 的阶乘为 720”。



## 【拓展训练 3-2】

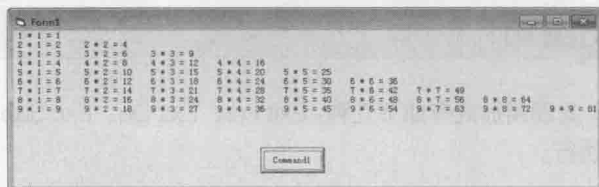
利用嵌套循环方法编写代码, 运行程序后, 单击窗体显示出九九乘法表。

条件循环结构与计数循环结构的用法类似, 下面来尝试一下使用嵌套循环显示出九九乘法表。

首先新建一个工程, 在窗体上添加一个按钮控件, 双击按钮控件, 进入代码窗口, 输入以下代码 (拓展代码 3-2.txt)。

```
01 Private Sub Command1_Click()
02   Dim i, j As Integer      ' 定义两个整型变量
03   For i = 1 To 9          ' 用 For 循环, 打印 9 行内容
04     j = 1 ' 为变量 j 赋初值
05     Do While j <= i        ' 用 Do 循环, 打印每一列中的内容
06       Print i; " * "; j; " = "; i * j, ' 输出每一列的内容
07       j = j + 1            ' 增加行数
08     Loop                  ' 满足条件, 继续内部循环
09     Print ' 打印空行, 起换行作用
10   Next i ' 满足条件, 继续外部循环
11 End Sub
```

运行后, 单击【Command1】按钮, 即可显示出九九乘法表。



## 3.6 其他辅助控制语句



本节视频教学录像：11 分钟

除了上面提到的几种语句外，Visual Basic 还提供了其他一些辅助控制语句。下面简单介绍其用法。

### 3.6.1 End 结束语句

End 语句可以结束一个过程或块，独立的 End 语句用于结束一个程序的运行，可以放在任何事件过程中。该语句使用形式如下。

End

此外，在 Visual Basic 6.0 中，还有多种形式的 End 语句，用于结束一个过程或块，主要形式包括 End If、End Select、End With、End Type、End Function、End Sub 等，与对应的语句配对使用。End 语句在各种代码块中的格式及作用如下表所示。

语句类型	作用
End	停止执行程序，在 Visual Basic 6.0 中常用来退出程序
End Function	表示函数体结束
End If	表示 If 结构结束
End Select	表示 Select 结构结束
End Sub	表示 Sub 过程结束
End Type	表示用户自定义类型结束
End With	表示 With 语句结束
End Property	表示 Property 语句结束

End 语句用于停止执行过程或块，可以在过程中的任何位置关闭代码的执行。在执行时，End 语句会重置所有模块级别变量和所有模块的静态局部变量。若要保留这些变量的值，改为使用 Stop 语句，则可以在保留这些变量值的基础上恢复执行。



注意

End 语句不调用 Unload、QueryUnload 或 Terminate 事件或任何其他 Visual Basic 代码，只是生硬地终止代码的执行，并且释放程序所占用的内存资源。

### 3.6.2 Exit 退出语句

Exit 语句为退出语句。其调用形式有如下几种：Exit For、Exit Do、Exit Sub、Exit Function，其功能都为退出某种控制结构的执行。

(1) Exit For：退出 For 循环结构。用于中途跳出 For 循环，可以直接使用，也可以用条件判断语句加以限制，在满足某个条件时才能执行此语句，跳出 For 循环。例如，在 For 循环内部添加语句“If 条件 Then Exit For”。

(2) Exit Do：退出 Do Loop 循环结构。用于中途跳出 Do 循环，与上述类似，既可以直接使用，也可以用条件判断语句限制使用。

(3) Exit Sub：退出 Sub 过程。用于中途跳出 Sub 过程，既可以直接使用，也可以用条件判断语句限制使用。

(4) Exit Function：退出 Function 函数。用于中途跳出 Function 过程，可以直接使用，也可以用条件判断语句限制使用。

### 3.6.3 GoTo 跳转语句

GoTo 语句是无条件程序跳转语句，其功能是无条件地转移到标号或行号指定的一行语句。该语句使用形式如下。

GoTo { 标号 | 行号 }

在使用上述形式的 GoTo 语句跳转到某一个标号的代码时，应注意如下问题。

(1) GoTo 语句只能转移到同一过程的标号或行号处。标号是一个字符序列，首字符必须为字母，与大小写无关，任何转移到的标号后应有冒号。

(2) 行号是一个数字序列。

(3) 结构化程序设计中要求尽量少用或不用 GoTo 语句，用选择结构或循环结构来代替。



**提示**

Goto 语句在使用上争议很大。因为 Goto 语句使用起来比较灵活，而且有些情形能够提高程序的效率，使得很多开发人员喜欢使用该语句。但是 Goto 语句使程序的静态结构和程序的动态执行之间差别很大，造成程序难以阅读，难以查错。

### 3.6.4 On Error 语句

On Error 语句用于启动一个错误处理程序并指定该子程序在一个过程中的位置，也可用来禁止一个错误处理程序。

语法有 On Error GoTo line、On Error Resume Next、On Error GoTo 0 等。

On Error 语句的语法可以具有以下任何一种形式的语句描述。

#### 1. On Error GoTo line

启动错误处理程序，且该历程从必要的 line 参数中指定的 line 开始。line 参数可以是任何行标签或行号。如果发生一个运行时错误，则控件会跳到 line，激活错误处理程序。指定的 line 必须在一个过程中，这个过程与 On Error 语句相同，否则会发生编译时间错误。

#### 2. On Error Resume Next

说明当一个程序运行时发生错误时，控件转到紧接着发生错误的语句之后的语句，并在此继续运行。访问对象时要使用这种形式而不使用 On Error GoTo。



### 3. On Error GoTo 0

禁止当前过程中任何已启动的错误处理程序。

说明：如果不使用 On Error 语句，则任何运行时错误都是致命的。也就是说，结果会导致显示错误信息并中止运行。一个“允许的”错误处理程序是由 On Error 语句打开的一个处理程序，一个“活动的”错误处理程序是处理错误的过程中允许的错误的处理程序。如果在错误处理程序处于活动状态时（在发生错误和执行 Resume、Exit Sub、Exit Function 或 Exit Property 语句之间这段时间）又发生错误，则当前过程的错误处理程序将无法处理这个错误。控件返回调用的过程。如果调用过程有一个已启动的错误处理程序，则激活错误处理程序来处理该错误。如果调用过程的错误处理程序也是活动的，则控件将再往回传到前面的调用过程，这样一直进行下去，直到找到一个被允许的但不是活动的错误处理程序为止。如果没有找到被允许而且不活动的错误处理程序，那么在错误实际发生的地方，错误本身是严重的。错误处理程序每次将控件返回调用过程时，该过程就成为当前过程。在任何过程中，一旦错误处理程序处理了错误，在当前过程中就会从 Resume 语句指定的位置恢复运行。注意一个错误处理程序不是 Sub 过程或 Function 过程。它是一段用行标签或行号标记的代码。

## 3.6.5 复用语句 With...End With

With 语句用于定制一个对象或用户自定义类型。With 语句不仅方便用户定制对象，也可增强代码的可读性使程序结构更加清晰。With 语句用于在一个单一对象或一个用户定义类型上执行一系列的语句。该语句的使用形式如下。

```
With object
    [statements]
End with
```

With 语句的语法具有以下两个参数。

- (1) object：必要参数，一个对象或用户自定义类型的名称。
- (2) Statements：可选参数，对 object 执行一条或多条语句。

With 语句可以对某个对象执行一系列的语句，而不用重复指出对象的名称。例如，要改变一个对象的多个属性，可以在 With 控制结构中加入属性的赋值语句，这时只是引用对象一次而不是在每个属性赋值时都要引用。例如，下列语句使用 With 语句执行一系列对标签控件的参数设置。

```
With MyLabel
    .Height = 2000
    .Width = 2000
    .Caption = "Visual Basic"
End With
```



**技巧**

当程序一旦进入 With 块，object 就不能改变，因此不能用一个 With 语句来设置多个不同的对象。

## 3.7 高手点拨



本节视频教学录像: 12 分钟

上面只是对几种程序控制结构的简单介绍, 在实际应用中, 很多复杂的程序都是由这些基本结构组成, 下面介绍一些程序中常用的算法。

### 1. 累加、累乘

累加、累乘是较常见的数值问题。累加(累乘)是将多个数相加(乘), 所以一般采用循环结构来实现。在循环体中应有表示累加(如  $\text{sum}=\text{sum}+\text{x}$ )或累乘(如  $\text{t}=\text{t}*\text{i}$ )的赋值语句。需要注意的是, 累加中用于存放和的变量一般赋初值为 0, 而累乘中用于存放积的变量赋初值为 1。

下面以例子说明。

求  $1! + 2! + \dots + n!$  的值,  $n$  由键盘输入。

```
Private Sub Form_Click()
    Dim s as double,t as double,n as integer
    s=0
    t=1
    n=InputBox(" 请输入 n 的值 ")
    For i=1 to n
        t=t*i      ' 求 i! 并赋给变量 t
        s=s+t
    Next i
    Print "1!+2!+...+";n;"!=";s
End Sub
```

### 2. 求最大值和最小值

求若干数的最大值(最小值), 其算法思想是, 先取第一个数作为最大值(最小值)的初值, 然后依次将下一个数和它比较, 若比它大(小), 将该数替换为最大值(最小值)。

下面以例子说明。

产生 10 个 1~100 之间的随机整数, 输出它们的最大值。

```
Private Sub Form_Click()
    Dim i%,x%,max%
    Randomize
    Print "10 个随机整数:"
    x=int(Rnd*100)+1
    Print x;
    max=x      ' 将第一个数作为初值赋给 max
    For i=2 to 10
        x=int(Rnd*100)+1
        if x>max Then max=x
    Next i
    Print x;
```

```

Next i
Print
Print "最大值为: ";max
End Sub

```

### 3. 迭代法

“迭代法”是指重复执行一组工作，每次执行这组工作时，都从旧值递推出新值，并用新值代替旧值。

下面以例子说明。

已知某球从 100 米高度自由落下，落地后反弹，每次弹起的高度都是上次高度的一半。求小球第 10 次落地后反弹的高度和球所经过的路程。

```

Private Sub Form_Click()
Dim h as single,r as single,s as single
h=100
s=0
For i=1 to 10
    r=h/2
    s=s+r+h
    h=r
Next i
Print "h=";h,"s=";s
End Sub

```

## 3.8 实战练习

### 上机题

运用计数循环结构计算出  $1 + 2 + 3 + 4 + \cdots + 100$  的结果。

# 第4章



本章视频教学录像：1小时2分钟

## 同类型批量数据管理的技巧——数组

在实际应用过程中，将一批相互有联系、有一定顺序、同一类型和具有相同性质的数据采用集合进行定义和存储，这样的集合就是数组。数组在同类型批量数据管理中有着很广泛的应用。本章将带领读者从数组的基础开始循序渐进地学习。本章介绍数组的基本概念、基本操作和数组相关函数及语句，主要包括定长和动态数组和数组元素的输入、输出、插入、查找、删除等。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解数组的基本概念和分类
- ☐ 了解数组的引用和初始化
- ☐ 掌握数组元素的插入、删除和查找
- ☐ 熟悉数组元素的应用及排序
- ☐ 了解数组相关的函数
- ☐ 了解数组相关的语句

## 4.1 数组的概念



本节视频教学录像: 10 分钟

数组的英文名是“Array”。Array 在英文中是“阵列”的意思。所谓“阵列”，就是一组集中的、排列有序的集合。数组并不是一种数据类型，而是一组相同类型的变量的集合，可用于存储和处理成组的有序数据，数组必须先声明后使用。每一个数组给定一个名称，叫作数组名；数组中的每一个数或变量被称为数组元素；一个数组中各个数组元素之间的区别用数组的下标来表示，放在数组名后面的括号内，因此，数组元素又被称为下标变量。

总体上数组可分为两类：定长（静态）数组和动态（可变量）数组。



提示

数组和简单变量相比有以下几个特点。

- (1) 数组的命名规则和简单变量的命名规则相同。
- (2) 数组的数据类型和数组中所存储的数组元素的类型必须相同。
- (3) 由于受到计算机硬件和软件条件的限制，数组中的元素个数必须是有限的。

### 4.1.1 定长数组及声明

定长数组是一组具有相同类型的变量的有序集合，集合中的变量称为数组元素。数组元素共用一个名字，通过下标来唯一标识自己。

在使用某个数组前，需要先声明该数组。对数组进行声明，应该包括数组名、维数、大小、类型以及作用域等。

数组声明语句的具体语法是：

```
Dim | Private | Public | Static 数组名 ([下界 1 To] 上界 1, <[下界 2 To] 上界 2, ……>) [As 数据类型]
```

对数组声明语句的说明如下。

Dim：声明普通局部数组。

Private：声明模块级数组。

Public：声明可在工程任何模块中使用的数组。

Static：声明静态数组。

As：用来说明数组元素的类型。声明时若没有指定数组的数据类型，则默认是变体型。

例如：

```
Dim Private A(6 To 12) As Integer '数组 A 为私有整型一维数组，共 7 个元素，下标从 6 到 12。
```

```
Dim Public N(-3 to 6,9) '声明数组 N 为公用二维数组，共有 10*10=100 个元素。
```



注意

在声明数组时，也可以使用类型说明符代替“[As 数据类型]”，例如：

```
Dim a$(10) '字符串类型数组
```

```
Dim b%(5, 5) '整型数组
```

## 4.1.2 动态数组及声明

有时我们并不能在编写程序的时候就确定数组中到底会存储多少元素，对于这种情况，一般的解决方法是对所有的数组都按照它可能的数组元素的最大值创建数组，但是这样会浪费很多存储空间。动态数组就是为了解决这个问题而产生的。动态数组中的数组元素个数不定，并且可以根据需要动态改变数组元素的个数。

可以按照下面的步骤创建一个动态数组。

(1) 首先像前面定长数组的声明一样，先声明一个数组，但是不说明维数和界限。

```
Dim | Private | Public | Static 数组名 () [As 数据类型]
```

(2) 然后在实际使用的时候用 ReDim 语句定制数组的维数和上下界，为数组分配实际的内存空间。

```
ReDim [Preserve] 数组名 (< 维数说明 >) [As 类型]
```

使用 ReDim 语句时，需要遵循下面一些规则。

(1) ReDim 语句中的维界定义中的上下界可以是常量，也可以是有了确定值的变量。

(2) ReDim 语句只能出现在过程体内，为数组临时分配存储空间，当所在过程结束时，分配的存储空间就会释放。

(3) 使用 Redim 语句时，如果不使用 Preserve 选项，原来数组中的值会丢失，即数组中的内容全部被重新初始化，当前存储在数组中的值会全部丢失。Visual Basic 会重新将数组元素的值置为 Empty（对 Variant 数组）、0（对 Numeric 数组）、零长度字符串（对 String 数组）或者 Nothing（对于对象的数组）。如果希望改变数组大小又不丢失数组中的数据，使用具有 Preserve 关键字的 ReDim 语句就可以做到这点。

使用 ReDim 语句时，如果使用 Preserve 选项，则在对数组重新说明时，将会保留数组中原来的数据，但是不能改变维数，并且只能改变最后一维的大小，前面维的大小不能改变。



**提示**

(1) 在动态数组 ReDim 语句中的下标可以是常量，也可以是确定值的变量。

(2) 在过程中可以多次使用 ReDim 来改变数组的大小，也可改变数组的维数。

(3) 每次使用 ReDim 语句都会使原来数组中的值丢失，可以在 ReDim 语句后加 Preserve 参数来保留数组中的数据，但只能改变最后一维的大小，前面几维大小不变。

## 4.2 数组基本操作



本节视频教学录像：34 分钟

在数组的学习中，数组的应用是非常重要的一项内容，而这就需要我们了解数组的一些基本功能，熟练掌握一些数组的基本操作和实现。本小节就让我们来一起学习一些数组的基本操作。

### 4.2.1 数组的引用

数组元素的引用——数组名（下标）。

例如：求数组中的最大元素及所在下标。

```
Dim Max As Integer, iMax As Integer
```



```
Dim ia(1 to 10) as integer
```

```
Max=ia(1) : iMax=1
```

```
For i = 2 To 10
```

```
    If ia(i)>Max Then
```

```
        Max=ia(i)
```

```
        iMax=i
```

```
    End If
```

```
Next i
```

## 4.2.2 数组的初始化

当数组被声明后, 根据数组被声明的数据类型, 数组中的元素将被赋予不同的默认值。

数组类型	所有数组元素将初始化为
数值数组	0
变体字符串数组	空字符串
定长字符串数组	给定长度的空格
逻辑型数组	False



**提示**

(1) 初始值与相应数组元素一一对应, 初始值相互之间用逗号隔开。

(2) 初始化前对数组元素的定义不能是具体的数据类型, 只能是 Variant 或默认类型。

(3) 对于该数组可以不定义而直接由 Array 函数来确定。

(4) 此种方法只能初始化一维数组。

## 4.2.3 数组元素的输入、输出

### 1. 数组元素的输入

对于数组元素较少的数组, 可通过单个赋值语句进行输入操作; 对于数组元素较多的数组, 一般通过 For 语句、InputBox 函数和文本框输入。

(1) 通过 InputBox 函数输入适合输入少量数据

例如:

```
01 Dim SB ( 3,4 ) As singer
```

```
02 For i=0 To 3
```

```
03 For j=0 To 4
```

```
04     SB(i,j)=InputBox(" 输入 "&i&j&" 的值 ")
```

```
05 Next j
```

```
06 Next i
```

(2) 通过 For 语句输入

例如:

```
01 Option Base 1 '默认下标下界为 1
02 DIM b(2,3) as single
03 For i=1 to 2
04     For i=1 to 3
05         b(i,j)=inputbox("输入 b(" & i & "," & j & ") 的值")
06     Next
07 Next
```

(3) 通过文本框控件输入

对大批量的数据输入, 采用文本框和函数 split()\join() 进行处理, 效率更高。

## 2. 数组的输出

数组的输出同样是以数组元素为操作对象, 可通过 Print 方法来实现, 也可用 For...Next 循环语句输出。

(1) 通过 For...Next 循环语句实现输出

例如:

```
01 For i = 1 To 10
02     Print A(i), '在一行上显示
03 Next i
04 For i = 1 To 100
05     Print B(i),
06     If i Mod 10 = 0 Then Print '每 10 个一行显示
07 Next i
```

(2) 通过 print 方法实现输出

例如: 输出方阵中下三角元素。

```
01 Dim sc(5, 5) As Integer, i As Integer, j As Integer
02 For i = 1 To 5
03     For j = 1 To 5
04         sc(i, j) = i * j; '数组赋初值
05     Next
06 Next
07 For i = 1 To 5
08     For j = 1 To i '内循环控制每行输出的元素个数
09         Print sc(i, j); '以紧凑格式输出数组元素
10     Next
11     Print
12 Next
```

## 4.2.4 数组元素的插入、删除和查找

数组元素的插入、删除操作一般是在已排序好的数组中插入或删除一个元素, 使得插入或删除以后的数

组还是有序的;这涉及到查找问题,在数组中先找到插入的位置或删除的元素,然后进行相应的操作。

### 1. 数组中元素的插入

例如:设有一按升序排列的有  $n$  个元素的数组  $a$  (数组元素为整型),现将一个数“11”插入后,仍保持其有序。

解题思路:

- (1) 查找插入位置  $k(1 \leq k \leq n-1)$ 。
- (2) 从  $n-1$  到  $k$  逐一往后移动一个位置,将第  $k$  个元素的位置腾出。
- (3) 将数据插入。

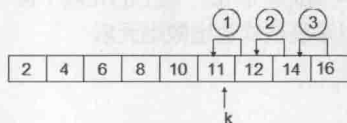
```

01 Dim a(), i, k As Integer
02 Print "插入前: ";
03 Redim a(1 To 9) ' 给数组元素赋值
04 a = Array(2, 4, 6, 8, 10, 12, 14, 16)
05 For i = 1 To 8
06   Print a(i);
07 Next
08 For k = 1 To 9 ' 查找插入的位置
09   If 10 = a(k) Then Exit For
10 Next
11 For i = 9 To k Step -1 ' 从 k+1 元素开始往前移动
12   a(i + 1) = a(i)
13 Next
14 a(k) = 11
15 Print "插入后: ";
16 For i = 1 To 9
17   Print a(i);
18 Next
  
```

### 2. 数组中元素的删除

解题思路:

- (1) 查找欲删除的元素所在的位置  $k$ 。
- (2) 从  $k+1$  到  $n$  个位置逐个往前移动。
- (3) 将数组个数减 1。



例如:删除数组一个元素。设数组的元素为 (2,4,6,8,10,11, 12,14,16), 删除其中的元素“11”。

```

01 Dim a(), i, k As Integer
02 Print "插入前: ";
  
```

```

03 Redim a(1 To 9) '给数组元素赋值
04 a = Array(2, 4, 6, 8, 10, 11, 12, 14, 16)
05 For i = 1 To 9
06   Print a(i);
07 Next
08 For k = 1 To 9 '查找插入的位置
09   If 11 = a(k) Then Exit For
10 Next
11 For i = k + 1 To 9 '从 k+1 元素开始往前移动
12   a(i - 1) = a(i)
13 Next
14 Redim Preserve a(1 To 8) '利用数组重新声明减少一个元素, 但保持数组原来的值
15 Print: Print '显示插入后的数组元素
16 Print "插入后: ";
17 For i = 1 To 8
18   Print a(i);
19 Next
  
```

## 4.2.5 数组元素的应用及排序

排序就是将一组数按递增或递减的次序重新排列。常用的方法有选择法、冒泡法、插入法。

### 1. 选择法排序

算法描述:

设有  $n$  个数的数组  $a(1), a(2), \dots, a(n)$ , 要求按递增的次序排列。

首先设变量  $k$  用于存放当前最小数的下标, 然后按下列步骤进行。

(1) 从  $n$  个数中选出最小元素的下标, 然后将最小数与第一个数交换位置, 即  $a(k)$  与  $a(1)$  互换。先将  $a(k)$  与  $a(2)$  进行比较,  $k$  的初值为 1, 若  $a(2) < a(k)$ , 则将变量  $k$  指向 2 ( $k$  总是为较小元素的下标值); 再将  $a(k)$  与  $a(3) \dots a(n-1)$ 、 $a(n)$  比较, 并依次做出同样的处理, 最终  $k$  为  $n$  个数中最小元素的下标,  $a(k)$  为最小的元素, 此时将  $a(k)$  与  $a(1)$  互换。

(2) 从除第一个数外, 其余  $n - 1$  个数按步骤(1)的方法选出次小的数, 使  $a(k)$  与  $a(2)$  交换位置。先将  $k$  置为 2, 将  $a(k)$  与  $a(3)$ 、 $a(4)$ 、 $\dots$ 、 $a(n-1)$ 、 $a(n)$  比较, 每当  $a(i) < a(k)$  时, 置  $k$  为  $i$ , 最后,  $k$  为第一轮余下的  $n - 1$  个数中的最小数, 将  $a(k)$  与  $a(2)$  互换。

(3) 继续进行第三轮, 第四轮...直到第  $n - 1$  轮, 余下的  $a(n)$  就是  $n$  个数中的最大值。

至此,  $n$  个数已从小到大顺序存放到数组  $a$  中。

### 2. 冒泡法排序

冒泡法在每一轮排序时将相邻的数比较, 当次序不对就交换位置, 冒泡法若有  $n$  个元素, 则需要比较  $n - 1$  轮, 比较方向可以从右向左, 也可以从左向右, 而且每一轮都是从头比较到尾。

例如: 将数组中的 6 个数, 用冒泡排序法排序。

```

01 Dim a(6) As Integer
  
```

```

02 Dim k, n, i, j, t As Integer
03 Randomize
04 n = 6
05 For i = 1 To n
06   a(i) = Int(Rnd * 9) + 1      ' 给数组 6 个元素赋值, 0 ~ 9 中的随机整数
07   Label3.Caption = Label3.Caption + Str(a(i))
08 Next
09 ' 冒泡排序法
10 For i = 1 To n - 1
11   For j = n To i + 1 Step -1
12     If a(j - 1) > a(j) Then    ' 相邻元素比较
13       t = a(j)
14       a(j) = a(j - 1)
15       a(j - 1) = t
16     End If
17   Next
18   Label4.Caption = Label4.Caption + Str(a(i)) ' 将排序结果显示在 Label 上
19 Next
20 Label4.Caption = Label4.Caption + Str(a(i))

```

## 4.3 数组相关函数及语句



本节视频教学录像: 14 分钟

数组相关函数及语句是数组的比较常用的实现方式, 本小节让我们来认识和学习一些在数组中比较常用的数组相关函数及语句。

### 4.3.1 Array 函数

Array 函数格式: 数组变量名 = Array ( 数组元素值表 )。

功能: 把一组数据赋给数组中的每个元素。

说明:

- (1) 数组元素值表是一个用逗号分隔的值表, 这些值构成数组的各元素值。
- (2) Array 函数仅适用于一维数组, 只能给 Variant 类型的变量赋值, 赋值后的数组大小由参数的个数决定。
- (3) 使用 array 函数创建的数组, 其下界受 Option base 语句指定的下界的限制。
- (4) 若不提供参数, 则创建一个长度为 0 的数组。



**提示**

数组变量名是预先定义的数组名。在数组变量名后没有括号。采用变量定义形式, 作为数组使用, 类型为 variant。

例如: 要将 1、2、3、4、5、6、7、8、9、10 这些值赋给数组 A, 可以使用下面的方法。

Dim A

A = Array(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)

## 4.3.2 UBound 函数和 LBound 函数

函数格式:

LBound( 数组名 [,N])

UBound( 数组名 [,N])

功能:

LBound 函数返回“数组名”指定的数组的第 N 维的下界。

UBound 函数返回“数组名”指定的数组的第 N 维的上界。

说明: N 为 1 表示第一维, N 为 2 表示第二维, 等等。如果省略 N, 则默认为 1。

例如: 要打印一维数组 A 的各个值, 可以通过下面的代码实现。

```
For I = LBound(A) To UBound(A)
```

```
    Print A(I);
```

```
Next I
```

例如: 要打印二维数组 B 的各个值, 可以通过下面的代码实现。

```
01 For I = LBound(B, 1) To UBound(B, 1)
```

```
02     For J = LBound(B, 2) To UBound(B, 2)
```

```
03         Print B(I, J);
```

```
04     Next J
```

```
05     Print
```

```
06 Next I
```

## 4.3.3 Split 函数

Split 函数格式: Split( 字符串表达式 [, 分隔符])

功能: 以某个指定符号作为分隔符, 将“字符串表达式”指定的字符串分离为若干个子字符串, 以这些子字符串为元素构成一个下标从零开始的一维数组。

说明: “字符串表达式”用于指定要被分隔的字符串, “分隔符”是可选的, 如果忽略, 则使用空格作为分隔符。例如:

Dim A

A = Split("how are you", " ")



执行以上赋值之后:

```
A(0)="how", A(1)="are", A(2)="you".
```

也可以用 Split 函数给一个动态数组赋值。例如:

```
Dim A() As String
A = Split("how are you", " ")
```

### 4.3.4 Option Base 语句

在模块级别中使用, 用来声明数组下标的缺省下界。


语法: Option Base {0 | 1}

说明: 由于下界的缺省设置是 0, 因此无需使用 Option Base 语句。如果使用该语句, 则必须写在模块的所有过程之前。一个模块中只能出现一次 Option Base, 且必须位于带维数的数组声明之前。

注意: Dim、Private、Public、ReDim 以及 Static 语句中的 To 子句提供了一种更灵活的方式来控制数组的下标。不过, 如果没有使用 To 子句显式地指定下界, 则可以使用 Option Base 将缺省下界设为 1。使用 Array 函数或 ParamArray 关键字创建的数组的下界为 0, Option Base 对 Array 或 ParamArray 不起作用。Option Base 语句只影响位于包含该语句的模块中的数组下界。例如:

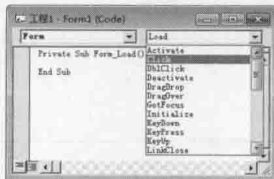
```
01 Option base 1 '将缺省的数组下标设为 1
02 Dim LowerDim MyArray(20)
03 TwoDArray(3, 4) '声明数组变量
04 Dim ZeroArray(0 To 5) '取代缺省的下标
05 LBound '使用函数来测试数组的下界
06 Lower = LBound(MyArray) '返回 1
07 Lower = LBound(TwoDArray, 2) '返回 1
08 Lower=LBound(ZeroArray)
```

#### 【范例 4-1】编写代码以实现运行程序后单击窗体, 在窗体上垂直输出 1、2、3、4、5。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在【Form1】窗体上单击鼠标右键, 在弹出的快捷菜单中选择【查看代码】菜单项, 进入代码窗口。

(3) 在代码窗口中选择【Form】窗体的【Click】事件。






#### 技巧

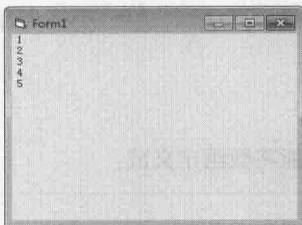
用鼠标双击窗体或者选择【视图】菜单下的【代码窗口】命令，也可以进入代码窗口。

(4) 输入以下代码。

```
01 Private Sub Form_Click()
02   Dim I As Integer ' 定义一个整型变量
03   Dim a%(5)        ' 定义一个整型数组，并为数组设置元素个数
04   For I = 1 To 5    ' 从 1 至 5 开始循环运算
05     a(I) = I        ' 为数组赋值
06     Print a(I)      ' 输出数组中的元素值
07   Next I ' 循环不结束，继续下一次循环
08 End Sub
```

### 【运行结果】

保存程序，单击【启动】按钮 ，运行程序。用鼠标单击窗体，将会在窗体中垂直显示 1、2、3、4、5 等数字。



### 【拓展训练】

既然能输出“1、2、3、4、5”这几个数字，那么要想输出 100 ~ 110 这一系列数字也就不是一件很难的事了。只需要对步骤(4)中的代码进行相关的修改就可以了（拓展代码 2-1.txt）。

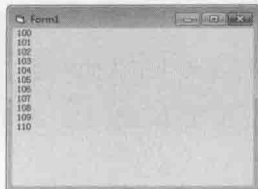
```
01 Private Sub Form_Click()
02   Dim I As Integer ' 定义整型变量
03   Dim a%(100 To 110) ' 定义整型数组，并为其设置坐标上限和下限
04   For I = 100 To 110 ' 利用循环语句开始循环
05     a(I) = I        ' 为数组赋值
06     Print a(I)      ' 输入数组元素
07   Next I ' 循环条件为真，进行下一次循环
08 End Sub
```



#### 提示

在修改程序时，主要是对数组所定义的上界和下界进行了修改。如果对程序限界定义的过大，则会造成系统资源的浪费；如果对程序上界和下界定义过少，则会出现下标越界的严重错误。

运行后，单击窗体就会看到所需的结果。



## 4.4 高手点拨



本节视频教学录像：4 分钟

在平时使用数组进行赋值时，以下一些常见的问题需要注意。

### 1. 只能将数据类型定义为可变类型

赋值的数据虽然可以有多种类型，但定义时必须将数据定义成可变型。若定义成其他类型，比如定义为整型：

```
Dim a() As Integer
A()=Array(1,3,5,10)
```

则系统会报告类型错误。就是说在利用 Dim 等进行定义时，除了 Variant 类型外，Integer 等其他类型都不适合这种形式。

### 2. 数组定义时不要指定上、下界

比如要对 10 个数据进行赋值，不能将数组定义成：

```
Dim s(10) As Variant
```

这样当利用 Array 函数赋值时系统将会报错。也就是说数组 s() 应定义成动态数组，不指定上下界，其上下界在用 Array 函数赋值时自动指定。

### 3. 有关数组的隐性声明

当使用数组的隐性声明形式时，即不对数组进行提前声明而直接使用 Array 函数赋值时，要注意此时数组名要省略其后面的括号，数组 s() 应写成 s，即有：

```
s=Array(...)
```

而如果写成 s() = Array(...)，系统将会报错。只要注意了以上几点，对数组的大量数据初始化问题就能得到顺利的解决。

## 4.5 实战练习

查找下标为 0，上标为 9 的数组中值为“5”的元素索引值，若没有，则返回“找不到”。

# 第5章



本章视频教学录像：1 小时 13 分钟

## 应用程序提升的法宝——内置函数与过程

使用 Visual Basic 中的内置函数，程序员在实现许多功能的同时可以减少大量代码的编写，而巧妙地使用过程则可使计算机完成特定的功能。本章主要介绍 Visual Basic 中常用的内置函数、Sub 过程和 Function 过程。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解数学函数
- ☐ 了解字符串函数
- ☐ 了解转换函数
- ☐ 了解时间日期函数
- ☐ 了解随机函数
- ☐ 了解判断函数
- ☐ 了解格式化函数
- ☐ 了解 Shell 函数
- ☐ 掌握过程及过程的嵌套与递归调用
- ☐ 了解参数传递

## 5.1 秘密武器——常用的内置函数



本节视频教学录像：38 分钟

巧妙地使用内置函数，程序员可以减少代码的编写量，轻松实现许多复杂的功能。本节介绍一些常用的内置函数。

### 5.1.1 数学函数

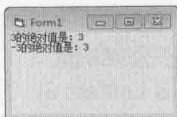
数学函数就是实现各种数学计算的函数，在 Visual Basic 中包含有丰富的数学函数供用户使用。常用的数学函数如表所示。

函数名	功能	参数单位	用法	说明
Sin()	求正弦值	度	Sin(number)	number 参数是 Double 或任何有效的数值表达式，表示一个以弧度为单位的角。结果的取值范围在 -1 到 1 之间
Cos()	求余弦值	弧度	Cos(number)	number 参数是 Double 或任何有效的数值表达式，表示一个以弧度为单位的角。结果的取值范围在 -1 到 1 之间
Tan()	求正切值	弧度	Tan(number)	number 参数是 Double 或任何有效的数值表达式，表示一个以弧度为单位的角
Atn()	求反正切值	弧度	Atn(number)	number 参数是 Double 或任何有效的数值表达式，表示一个以弧度为单位的角。值的范围在 -pi/2 和 pi/2 弧度之间
Log()	求自然对数值	数值	Log(number)	number 参数是 Double 或任何有效的大于 0 的数值表达式。自然对数是以 e 为底的对数，常数 e 的值大约是 2.718 282
Exp()	求以 e 为底的指数值	数值	Exp(number)	number 参数是 Double 或任何有效的数值表达式。如果 number 的值超过 709.782 712 893，则会导致错误发生
Sqr()	求平方根值	数值	Sqr(number)	number 参数是 Double 或任何有效的大于 0 的数值表达式

例如，使用 Abs 函数来计算“3”和“-3”的绝对值。

```
01 Private Sub Form_Activate()  
02   Print "3 的绝对值是：" & Abs(3)      ' 输入正数 3 的绝对值  
03   Print "-3 的绝对值是：" & Abs(-3)    ' 输出负数 3 的绝对值  
04 End Sub
```

运行窗体后，结果如图所示。



## 5.1.2 字符串函数

在程序设计中，将能够对字符串处理的函数称为字符串函数。常用的字符串函数如表所示。

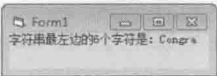
函数名	功能	用法	说明
Ltrim()	删除 string 字符串左边的空白字符	LTrim(string)	string 参数可以是任何有效的字符串表达式。如果 string 参数包含 Null，将返回 Null
Rtrim()	删除 string 字符串右边的空白字符	Rtrim(string)	string 参数可以是任何有效的字符串表达式。如果 string 参数包含 Null，将返回 Null
Left()	返回字符串 string 中左起指定长度为 length 的字符串	Left(string, length)	函数执行成功时返回 string 字符串左边长度为 length 的子串。发生错误时返回空字符串("")。如果任何参数为 NULL，将返回 NULL
Right()	返回字符串 string 中右起指定长度为 length 的字符串	Right(string, length)	函数执行成功时返回 string 字符串右边长度为 length 的子串。发生错误时返回空字符串("")。如果任何参数为 NULL，将返回 NULL
Mid()	返回 string 字符串中长度为 length 的字符	Mid(string, start[, length])	函数执行成功时返回 string 字符串从 start 开始的长度为 length 的子串
Len()	返回 string 字符串内字符的数量	Len(string   varname)	例如：Len(" 我爱 VB") 将返回值 4
LenB()	返回 string 字符串所占的字节数	LenB(string   varname)	例如：LenB(" 我爱 VB")；将返回值 6，这是因为一个中文占两个字节
Ucase()	将 string 字符串中的小写字母转换为大写字母	UCase(string)	string 为任何有效的字符串表达式。如果 string 包含 Null，将返回 Null 例如：Ucase("cat")；返回值："CAT"
Space()	产生指定长度为 number 的空字符串	Space(number)	无
String()	返回指定长度重复字符的字符串	String(number, character)	例如：String(3, "XYZ")；返回值："XXX"
InStrRev()	返回从字符串的末尾算起，一个字符串在另一个字符串中出现的位置	InStrRev(string 1, string2[, start [, compare]])	Start 可选，设置每次搜索的开始位置，默认为 -1；Compare 可选，在计算字符串时，指示要使用的比较类型的字符，默认为二进制

例如，使用 Left() 函数获取某字符串中最左边的 6 个字符。



```
01 Private Sub Form_Activate()  
02   Dim str As String      ' 定义字符串变量  
03   str = "Congratulations" ' 为字符串变量赋初值  
04   Print "字符串最左边的 6 个字符是：" & Left(str, 6) ' 输出字符串最左边 6 位  
05 End Sub
```

运行窗体后，结果如图所示。



### 5.1.3 转换函数

在编程处理数据的时候，有时需要在各种数据类型间进行转换，Visual Basic 中提供有很多数据类型转换的函数，特别是各种进制数据的转换以及字符串和数字的转换函数。

#### 1. 进制转换函数

常见数据进制有十进制、二进制、八进制、十六进制等。十进制是我们最为常见和常用的数据进制，二进制是计算机的基础进制，八进制和十六进制因为和二进制有着天然的亲密关系，并且表示比二进制更方便，因此也成为计算机的常用表示方法。

在 Visual Basic 中进行数据进制转换的函数中，常用的有以下几种。

##### (1) Hex() 函数

求出参数的十六进制形式。

语法：

Hex(number)

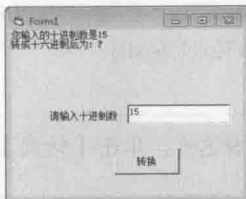
number 参数为任何有效的数值表达式或字符串表达式，不同取值情况下的函数返回结果如表所示。

number 参数值	函数结果
Null	Null
Empty	0
任何其他数字	最多 8 个十六进制字符

例如输入一个十进制数，将其转换成十六进制，并将转换后的结果输出在窗体上。

```
01 Private Sub Command1_Click()  
02   Print "您输入的十进制数是 " & Text1.Text ' 输出获得的十进制数  
03   Print "转成十六进制后为：" & Hex(Val(Text1.Text)) ' 输出转换后的结果  
04 End Sub
```

输入一个十进制数“15”，单击【转换】按钮，结果如图所示。



## (2) Oct() 函数

求出参数的八进制形式。

语法：

Oct(number)

number 参数为任何有效的数值表达式或字符串表达式，不同取值情况下的函数返回结果如表所示。

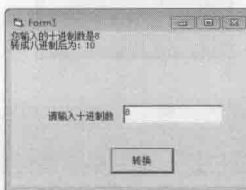
number 参数值	函数结果
Null	Null
Empty	0
任何其他的数字	最多 11 个十六进制字符

例如输入一个十进制数，将其转换成八进制，并将转换后的结果输出在窗体上。

```

01 Private Sub Command1_Click()
02   Print " 您输入的十进制数是 " & Text1.Text    ' 输出获得的十进制数
03   Print " 转成八进制后为: " & Oct(Val(Text1.Text))    ' 输出转换后的结果
04 End Sub
    
```

输入一个十进制数“8”，单击【转换】按钮，结果如图所示。



## (3) Cint() 函数

用于强制将一个表达式转换成 -32 768~ 327 67 的整型。

语法：

Cint(number)

number 参数为任何有效的数值表达式或返回值为数值的字符串表达式。

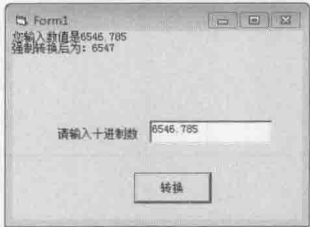
例如输入一个带小数的数值，将其强制转换成整型，并将转换后的结果输出在窗体上。

```

01 Private Sub Command1_Click()
    
```

```
02 Print " 您输入数值是 " & Text1.Text           ' 输出获得的十进制数
03 Print " 强制转换后为: " & CInt(Val(Text1.Text)) ' 输出转换后的结果
04 End Sub
```

输入任意有效的数值表达式或字符串表达式，单击【转换】按钮，结果如图所示。



2. 字符串和数字转换函数

(1) Str() 函数

将数值型数据转换为字符型数值。

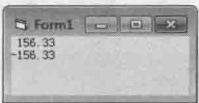
语法：

Str(number)

number 参数为长整型，其中可包含任何有效的数值表达式。

例如，将一个数字转换成字符串。

```
01 Private Sub Form_Activate()
02 Print str(156.33) ' 返回 " 156.33"
03 Print str(-156.33) ' 返回 " -156.33"
04 End Sub
```



提示

当数字转换成字符时，字符串的首位是一个空格或是正负号。

(2) Val() 函数

返回包含于字符串内的数字。

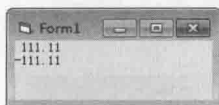
语法：

Val(string)

string 参数可以是任何有效的字符串表达式。

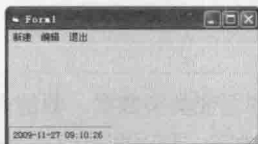
例如，返回字符串中包含的数值。

```
01 Private Sub Form_Activate()
02   Print Val("111.11")      ' 返回 " 111.11"
03   Print Val("- 111 .11")    ' 字符中的空格被除去了, 返回 "-111.11"
04 End Sub
```



## 5.1.4 日期时间函数

在编写程序的时候, 往往会遇到各种样式的日期和时间, 例如在状态栏中显示当前的系统日期等。



下面介绍几个常用的日期时间函数和日期时间转换函数。

### 1. Date ( ) 函数

用于返回系统的当前日期。

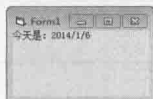
语法:

Date

例如, 在窗体上显示系统的当前日期。

```
01 Private Sub Form_Activate()
02   Print " 今天是: " & Date      ' 返回系统当前日期
03 End Sub
```

结果如图所示。



### 2. Day(date) 函数

用于表示一个月中的某一日, 返回值为 1~31 之间的整数。

语法:

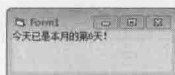
Day(date)

date 参数可以是任何能够表示日期的可变类型数据、数值表达式、字符串表达式或它们的组合。如果 date 包含 Null, 则返回 Null; 如果不为空, 则返回当前指定的日期的某一日。

例如, 在窗体上显示系统当前是本月的第几天。

```
01 Private Sub Form_Activate()  
02 Print "今天已是本月的第" & Day(Date) & "天!" ' 返回系统当前是本月第几天  
03 End Sub
```

结果如图所示。



### 3. Month() 函数

用于返回 1 到 12 之间的整数, 表示一年中的某月。

语法:

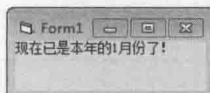
```
Month(date)
```

date 参数可以是任何能够表示日期的可变类型数据、数值表达式、字符串表达式或它们的组合。如果 date 包含 Null, 则返回 Null; 如果不为空, 则返回当前指定的日期的某一月。

例如, 在窗体上显示系统当前本年的月份。

```
01 Private Sub Form_Activate()  
02 Print "现在已是本年的" & Month(Date) & "月份了!" ' 返回系统当前月份  
03 End Sub
```

结果如图所示。



### 4. Time() 函数

用于返回当前系统时间 (不包含日期)。

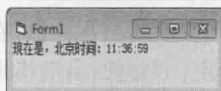
语法:

```
Time
```

例如, 在窗体上显示当前的时间。

```
01 Private Sub Form_Activate()  
02 Print "现在是, 北京时间:" & Time ' 返回系统当前时间  
03 End Sub
```

结果如图所示。



## 5. Weekday() 函数

返回某个日期是星期几。

语法：

Weekday(date,[firstdayofweek])

Weekday 函数语法中参数的含义如表所示。

参数	描述
date	必选参数，能够表示日期的可变类型数据、数值表达式或字符串表达式等。如果 date 包含 Null，则返回 Null
Firstdayofweek	必选参数，指定一星期第 1 天的常数，没有指定时以 vbSunday 为默认值

firstdayofweek 参数设置值如表所示。

常数	值	描述
vbUseSystem	0	使用 NLS API 设置
vbSunday	1	星期日（默认值）
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六

Weekday 函数的返回值如表所示。

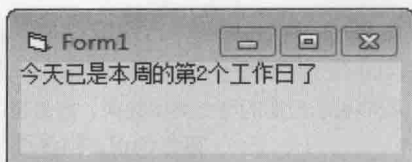
常数	值	描述
vbSunday	1	星期日
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六



例如, 返回当前天数是本周的第几个工作日。

```
01 Private Sub Form_Activate()  
02   Print "今天已是本周的第" & Weekday(Date) & "个工作日了" ' 返回系统工作日  
03 End Sub
```

效果如图所示。



## 6. Year() 函数

用于返回日期中的年份。

语法:

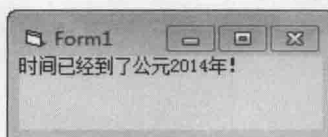
```
Year(date)
```

date 参数可以是任何能够表示日期的可变类型数据、数值表达式、字符串表达式或它们的组合。如果 date 包含 Null, 则返回 Null; 如果不为空, 则返回当前指定的年份。

例如, 在窗体上显示系统日期是哪一年。

```
01 Private Sub Form_Activate()  
02   Print "时间已经到了公元" & Year(Date) & "年!" ' 返回当前年份  
03 End Sub
```

结果如图所示。



## 5.1.5 随机函数

在编程处理数据的时候, 经常会用到随机函数, 包括 Randomize 函数和 Rnd 函数。

### 1. Randomize 函数

Randomize 函数是初始化随机数生成器。

语法:

```
Randomize[number]
```

number: 可选参数, 是 Variant 类型的值或任何有效的数值表达式。



**提示**

若想得到重复的随机序列，要在使用具有数值参数的 Randomize 之前直接调用具有负参数值的 Rnd，使用具有同样数值的 Randomize 是不会得到重复的随机数序列的。

## 2. Rnd 函数

Rnd 函数用于返回一个 Single 类型的随机数值。

语法：

Rnd[(number)]

如果 number 的值是 Rnd 生成，则 number 的值：

小于 0：每次都使用 number 作为随机数种子得到的相同结果。

大于 0：序列中的下一个随机数。

等于 0：最近生成的数。

默认：序列中的下一个随机数。



**注意**

Rnd 函数返回小于 1 但大于或等于 0 的值。number 的值决定了 Rnd 生成随机数的方式。

对最初给定的种子都会生成相同的数列，因为每一次调用 Rnd 函数都用数列中的前一个数作为下一个数的种子。在调用 Rnd 之前，先使用无参数的 Randomize 语句初始化随机数生成器，该生成器具有根据系统计时器得到的种子。

为了生成某个范围内的随机整数，可使用以下公式。

$\text{Int}((\text{upperbound} - \text{lowerbound} + 1) * \text{Rnd} + \text{lowerbound})$

这里，upperbound 是随机数范围的上限，而 lowerbound 则是随机数范围的下限。



**提示**

若想得到重复的随机数序列，在使用具有数值参数的 Randomize 之前直接调用具有负参数值的 Rnd。使用具有同样 number 值的 Randomize 是不会得到重复的随机数序列的。

## 5.1.6 判断函数

在进行数据验证的过程中，使用判断函数会为编程工作带来极大的方便，判断函数包括 IsNull 函数、IsNumeric 函数和 IsArray 函数。

### 1. IsNull 函数

IsNull 函数用于返回一个 Boolean 类型值，指出表达式是否不包含任何有效数据 ( Null )。

语法：

IsNull(expression)

expression：必要参数，是一个 Variant 类型的值，其中包含数值表达式或字符串表达式。

2. IsNumeric 函数

Isnumeric 函数用于返回一个 Boolean 类型的值，指出表达式的运算结果是否为数值。

语法：

IsNumeric(expression)

expression：必要参数，是一个 Variant 类型的值，包含数值表达式或字符串表达式。

3. IsArray 函数

IsArray 函数用于返回一个 Boolean 类型的值，指出变量是否为一个数组。

语法：

IsArray(varname)

Varname：必要参数，是一个指定变量的标识符。

5.1.7 格式化函数

Visual Basic 中的格式化函数也称 Format 函数。

用于返回 Variant(String) 类型值，其中含有一个表达式，它是根据格式表达式中的指令来格式化数据的。

语法：

Format(expression[,format[,firstdayofweek[,firstweekofyear]]])

Format 函数的语法具有下面几个部分。

部分	说明
expression	必要参数。任何有效的表达式
format	可选参数。有效的命名表达式或用户自定义格式表达式
firstdayofweek	可选参数。常数，表示一星期的第一天
firstweekofyear	可选参数。常数，表示一年的第一周

1. 设置值

firstdayofweek 参数有下面的设置。

常数	值	描述
vbUseSystem	0	使用 NLS API 设置
VbSunday	1	星期日 (缺省)
vbMonday	2	星期一
vbTuesday	3	星期二
vbWednesday	4	星期三
vbThursday	5	星期四
vbFriday	6	星期五
vbSaturday	7	星期六
vbUseSystem	0	使用 NLS API 设置
vbFirstJan1	1	从包含一月一日的那一周开始 (缺省)
vbFirstFourDays	2	从本年第一周开始, 而此周至少有四天在本年中
VbFirstFullWeek	3	从本年第一周开始, 而此周完全在本年中

## 2. 说明

firstweekofyear 参数有下面的设置。

格式化	设置
数字	使用预先定义的命名数值格式或创建用户自定义数值格式
日期和时间	使用预先定义的命名日期 / 时间格式或创建用户自定义日期 / 时间格式
日期和时间序数	使用日期和时间格式或数值格式
字符串	创建自定义的字符串格式

如果在格式化数字时没有指定 format, Format 会提供与 Str 函数类似的功能, 尽管它是国际化的。然而, 以 Format 作用在正数上不会保留正负号空间, 而以 Str 的话则会。

如果要格式化一个没有本地化的数值字符串, 应该使用一个用户自定义的数值格式, 以保证需要的外观。

注意如果 Calendar 属性设置是 Gregorian, 并且 format 指定了日期格式, 那么, 提供的 expression 必须是 Gregorian。如果 Visual BasicCalendar 属性设置是 Hijri, 则提供的 expression 必须是 Hijri。

如果日历是 Gregorian, 则 format 表达式的意义没有改变。如果日历是 Hijri, 则所有的日期格式符

号 (例如, dddd,mmmm,yyyy) 有相同的意义, 这些意义只应用于 Hijri 日历。格式符号保持英文, 用于文本显示的符号 (例如, AM 和 PM) 显示与该符号有关的字符串 (英文或阿拉伯数字)。当日历是 Hijri 时, 一些符号的范围会改变。

### 5.1.8 Shell 函数

除了以上介绍的几个函数外, 本小节介绍 Shell 函数。Shell 函数的作用是在 Visual Basic 应用程序的执行过程中调用执行其他的应用程序, 凡是能在 DOS 下或 Windows 下运行的可执行程序, 均可通过 Shell() 函数在 Visual Basic 应用程序中调用。

Shell() 函数的语法如下。

```
Shell( 命令字符串 [, 窗口类型])
```

命令字符串是要执行的应用程序名, 包括路径, 它必须是可执行文件 (扩展名为 .com、.exe 或 .bat)。

窗口类型是指执行应用程序的窗口大小, 可选择 0 ~ 4 或 6 的整型数值。取 1, 表示正常窗口状态。默认值为 2, 表示窗口被最小化成任务栏上的一个图标。

## 5.2 提升法宝——过程



本节视频教学录像: 32 分钟

过程就是一个具有相对独立的代码块。在一个大型系统中, 会有很多独立的代码块, 每个代码块将完成一定的功能, 最后把各个代码块组合起来就实现了大型系统的全部功能。

### 5.2.1 事件过程

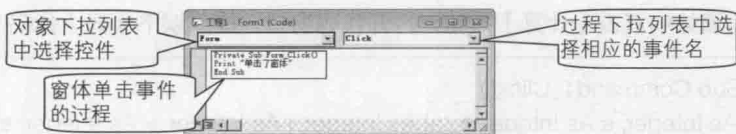
当用户对某个窗体或控件等对象发出了一个动作后, 将会产生一个事件。Visual Basic 会自动调用与该事件有关的过程, 该过程就是一个事件过程。

事件过程的语法是:

```
Private Sub <对象名>_<事件名>(<形式参数表>))
<代码段>
End Sub
```

不同的对象所对应的事件过程的命名方式有所不同, 窗体和控件都有各自的事件过程命名方式。

例如每个窗体都有自己的名称, 但是不论窗体的名称是什么, 在窗体的事件过程中统一用 Form 表示。在编写控件的事件过程时, 在代码编辑窗口左侧的对象下拉列表中选择控件, 在代码编辑窗口右侧的过程下拉列表中选择相应的事件名。



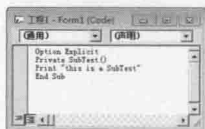
## 5.2.2 Sub 过程（子过程）

Sub 过程是指根据特定的实际需要而编写的，用来实现某种功能的代码块。例如，一本《Visual Basic 6.0 从入门到精通》的书的出版是一个最终的结果，在这期间，出版社会找优秀的作者去写这本书，然后再找排版人员对这本书进行排版，最后由编辑对书进行审校。一本书的出版，对出版社来说并不需要完全由自己亲自去写书、排版和审校，对于写书和排版来说，只需要调用相应的人员去做即可。

子过程的语法是：

```
[Private|Public|Static] Sub <过程名>(<形式参数表>)  
<代码段>  
End Sub
```

我们创建一个子过程的时候，不需要在某个特定窗体或者模块内编写，直接在代码窗口中输入子过程的代码即可。下图建立了一个新的子过程：SubTest。




创建一个通用过程的目的就是为了调用它。由于子过程是一个独立的过程，不能通过语句直接返回值，所以要用以下两种形式调用一个子过程。

- (1) 使用 Call 语句：Call <过程名>([<实参表>])。
- (2) 直接使用过程名：<过程名>([<实参表>])。

“实参表”是可选项，“实参表”中的参数必须与子过程的形式参数表中的参数一一对应，并且相应位置上的参数类型也要相同。实参表中的参数可以为常量、变量或者表达式，而子过程的形式参数表对过程传递参数的描述则包括形参变量的名称和类型。

**【范例 5-1】定义一个 Sub 子过程，主要用于进行阶乘的运算。输入一个数值，通过调用该 Sub 子过程，计算出该数值的阶乘值。**

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中。选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 窗体中添加 1 个命令按钮控件和 1 个文本框控件，并将命令按钮控件的 Caption 属性值设置为“计算”，将文本框控件的 Text 值设置为空。





(3) 在 Form1 窗体中双击【计算】按钮，打开代码窗口，输入以下代码（代码 5-1-1.txt）。

```
01 Private Sub Command1_Click()
02     Dim a As Integer, s As Integer, total As Integer, j As Integer, x As Integer, str As String
03     x = Val(Text1.Text) '将获取的文本内容转换为数值型
04     s = 0 's 用于得到计算的最终结果
05     str = "" '初始化字符串
06     For j = 1 To x '开始计数循环
07         Call fact(j, total) '调用阶乘计算的子过程，并向子过程传递两个参数
08         s = s + total '记录第 1 次运算的结果
09         str = str & j & "!" + " " '输入阶乘数
10     Next j '符合条件，进行下一次运算
11     str = str + "=" '得到最终计算结果
12     Print str; s; '输出最终计算结果
13 End Sub
```

(4) 在代码窗口中编写 fact 子过程的代码（代码 5-1-2.txt）。


```
01 Private Sub fact(n As Integer, t As Integer) '定义形式参数表
02     Dim i, s As Integer
03     s = 1 '初始化变量，用于计算阶乘的初始值
04     For i = 1 To n
05         s = s * i '累积
06     Next i
07     t = s '得到结果
08 End Sub
```

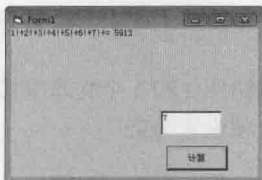


**提示**

子过程是一个独立的过程，可以写在其他过程以外的任何地方，但不可以在已有过程的内部编写。为了读取程序方便，通常在只有一个子过程的情况下，可以将其放在代码段的最后面或最前面。

## 【运行结果】

保存程序，单击【启动】按钮 ，运行程序。在文本框中输入数值“7”，单击【计算】按钮，结果如图所示。



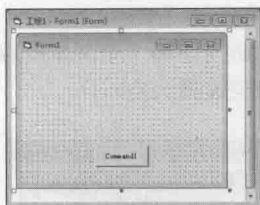
## 【拓展训练 5-1】

在 Visual Basic 中不仅可以调用有返回值的函数，同时还可以调用无返回值的函数。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单

击【打开】按钮。

(2) 在 Form1 窗体中添加 1 个命令按钮控件。



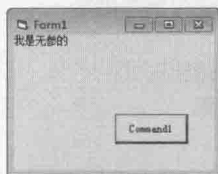
(3) 在 Form1 窗体中双击【Command1】按钮，进入代码窗口，输入以下代码。

```
01 Private Sub Command1_Click()  
02   Call XS '调用子函数  
03 End Sub
```

(4) 在代码窗口中编写 XS 子过程的代码。

```
01 Private Sub XS()  
02   Print "我是无参的" '输出字符串  
03 End Sub
```

(5) 保存程序，按【F5】快捷键运行程序，单击【Commad1】按钮查看程序效果。



### 5.2.3 Function 过程（函数过程）

函数过程也是用来完成具有特定功能的代码段，其语法是：

```
[Private|Public|Static] Function <函数过程名>(<形式参数表>)[As<返回值类型>]  
  <代码段>  
  <函数过程名>=<返回值>  
End Function
```




**提示**

Function 过程和 Sub 子过程有不同的地方，函数过程可以向调用过程明确地返回一个值。所以调用函数过程比调用子过程多了一种形式，共有以下 3 种形式。

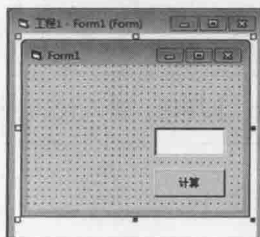
- (1) 使用 Call 语句：Call <函数名>([<实参表>])。
- (2) 直接使用过程名：<函数名>([<实参表>])。
- (3) 在表达式中调用：<返回值变量>=<函数名>([<实参表>])。

**【范例 5-2】** 定义一个 Function 过程，输入一个数值，通过调用 Function 函数，

## 计算出该数值的阶乘值。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在 Form1 窗体中添加 1 个命令按钮控件和 1 个文本框控件, 并将命令按钮控件的 Caption 属性值设置为“计算”, 将文本框控件的 Text 值设置为空。




(3) 在 Form1 窗体中双击【计算】按钮, 打开代码窗口, 输入以下代码 (代码 5-2-1.txt)。

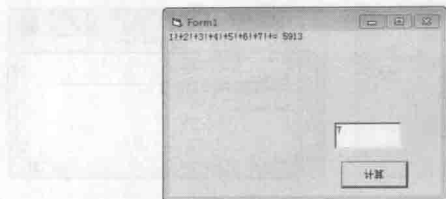
```
01 Private Sub Command1_Click()
02     Dim a As Integer, s As Integer, total As Integer, j As Integer, x As Integer, str As String
03     x = Val(Text1.Text)      ' 将获取的文本内容转换为数值型
04     s = 0                    ' s 用于得到计算的最终结果
05     For j = 1 To x           ' 利用计数循环开始计数
06         s = s + fact(j)      ' 调用过程函数
07         str = str & j & "!" + "    ' 格式化字符输出形式
08     Next j
09     str = str & "="          ' 再次格式化字符输出形式
10     Print str; s;           ' 输出最终计算结果
11 End Sub
```

(4) 在代码窗口中编写 fact 子过程的代码 (代码 5-2-2.txt)。

```
01 Private Function fact(n As Integer)                ' 定义形式参数表
02     Dim i, s As Integer
03     s = 1                                           ' 用于得到结果
04     For i = 1 To n
05         s = s * i                                   ' 累积
06     Next i                                         ' 满足条件进行下次循环
07     fact = s                                       ' 函数 fact 的返回值
08 End Function
```


## 【运行结果】

保存程序, 单击【启动】按钮 , 运行程序。在文本框中输入数值“7”, 单击【计算】按钮, 结果如图所示。



## 【拓展训练 5-2】

本实例主要讲解如何使用 Visual Basic 中提供的可视化图形窗口来创建一个过程。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 双击 Form1 窗体，进入代码窗口。

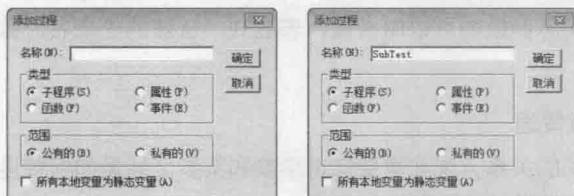


如果当前打开的窗口不是代码窗口，则无法进行下面的操作。

**技巧**

(3) 选择【工具】>【添加过程】菜单命令，打开【添加过程】对话框。

(4) 在【名称】文本框中输入过程名，例如输入“SubTest”。

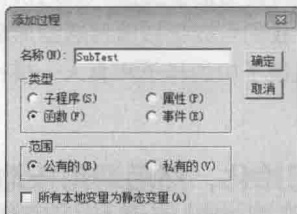


(5) 在【类型】选项组中选中【函数】单选按钮。



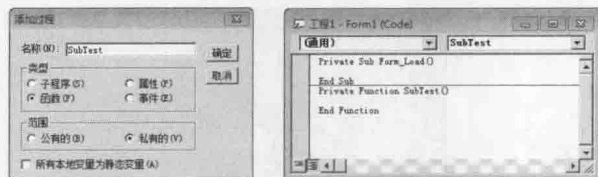
在【添加过程】对话框中的【类型】选项组中，可以选中【子程序】单选按钮来创建一个子过程 (Sub)，或者选中【函数】单选按钮来创建一个函数过程 (Function)。

**注意**



(6) 在【范围】选项组中，可以选中【公有的】单选按钮来创建一个公有过程，或者选中【私有的】单选按钮来创建一个私有过程。在这里选中【私有的】单选按钮，然后单击【确定】按钮。

(7) 这样就完成了一个函数过程的创建，但该函数过程并不能实现任何功能，而需要向函数过程中添加实现程序功能的代码。



## 5.2.4 参数的传递

我们使用过程的目的就是将一些数据传递给过程，过程经过计算后将结果传递出来。在调用的过程中，必须考虑调用过程和被调用过程之间的数据调用方式。在 Visual Basic 中，使用参数来实现主调过程和被调过程间的数据传递。

### 1. 形参与实参

参数在 Visual Basic 中根据功能被分为形式参数和实际参数两种。在编写过程的时候，我们并不知道所要计算数据的真实值，因此使用参数代表这些值，当真正的数值作为参数传递进入过程后，再将参数值替换为实际值。在过程中代表真正需要计算的值的参数就是形式参数，之所以称其为形式参数，是因为它仅仅是形式上代表真正需要计算的值。

(1) 形式参数。形式参数（简称形参）是接收数据的变量。形式参数表中的各个变量之间用逗号分隔。形式参数表中的变量类型可以是合法的简单变量，也可以是数组，但不能是定长字符串。

(2) 实际参数。实际参数（简称实参）是指在调用子过程或函数过程时，传递给子过程或函数过程的常量、变量或表达式。实际参数表可以由常量、表达式、合法的变量名、数组等组成，实际参数表中的各参数用逗号分隔。


### 2. 按地址传递和按值传递

知道了形参和实参之间的关系，现在需要知道形参和实参之间是如何传递数据的。实参与形参之间的参数传递方式有两种，即按地址传递和按值传递。

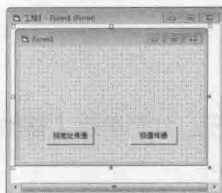
(1) 按地址传递（By Reference）。所谓按地址传递参数，就是将实参的地址传递给相应的形参，形参与实参使用相同的内存地址单元，这样通过调用被调程序就可以改变实参的值。在进行地址传递时，实参必须是变量，常量或表达式无法进行地址传递。按地址传递是 Visual Basic 中系统默认的参数传递方式。

(2) 按值传递（By Value）。所谓按值传递，就是当实参是常量时，直接将常量值传递给形参变量；当实参是变量时，仅仅将实参变量的值传递给形参变量。然后执行被调过程，在被调过程中，即使对形参发生改变也不会影响实参值的变化。我们可以把按值传递给形参的变量看做实参的一个“副本”或者“镜像”，就像镜子中的我们一样，虽然镜子中的样子和真人一模一样，但是镜子中的影像消失了并不会导致真人消失。

**【范例 5-3】对整型变量进行初始化，然后调用按地址传递和按值传递的函数，再将变量传递后的结果输出，比较按地址传递和按值传递的区别。**

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 窗体中添加两个命令按钮控件，将 Command1 命令按钮的 Caption 属性设置为“按地址传递”，将 Command2 命令按钮的 Caption 属性设置为“按值传递”。



(3) 在 Form1 窗体中双击【按地址传递】按钮，打开代码窗口，输入以下代码（代码 5-3-1.txt）。

```
01 Private Sub Command1_Click()  
02   Dim a As Integer, b As Integer  
03   a = 20 '为变量 a 赋初值  
04   b = 30 '为变量 b 赋初值  
05   Print "按地址传递（初始）: a="; a, "b="; b '输出变量 a 和 b 的初始值  
06   Call dizhisub(a, b) '调用名称为 dizhisub 的子过程  
07   Print "调用子过程结束以后: a="; a, "b="; b '输出调用函数后的结果  
08   Print "" '输出空行  
09 End Sub
```

(4) 在 Form1 窗体中双击【按值传递】按钮，打开代码窗口，输入以下代码（代码 5-3-2.txt）。

```
01 Private Sub Command2_Click()  
02   Dim a As Integer, b As Integer  
03   a = 20 '为变量 a 赋初值  
04   b = 30 '为变量 b 赋初值  
05   Print "按值传递（初始）: a="; a, "b="; b '输出变量 a 和 b 的初始值  
06   Call zhisub(a, b) '调用名称为 zhisub 的子过程  
07   Print "调用子过程结束以后: a="; a, "b="; b '输出调用函数后的结果  
08   Print "" '输出空行  
09 End Sub
```

(5) 在代码窗口中编写 dizhisub 的子过程的代码。

```
01 Private Sub dizhisub(m As Integer, n As Integer)  
02   Print "调用按地址传递的子程序（初始）: m="; m, "n="; n '输出形参 m 和形参 n 的初始值  
03   m = 50 '为形参 m 重新赋值  
04   n = 100 '为形参 n 重新赋值  
05   Print "调用按地址传递的子程序（子程序赋值以后）: m="; m, "n="; n '输出形参的新值  
06 End Sub
```

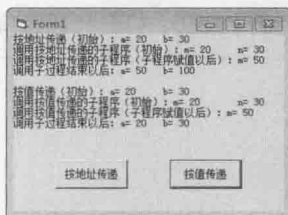
(6) 在代码窗口中编写 zhisub 的子过程的代码。

```
01 Private Sub zhisub(ByVal m, ByVal n)  
02   Print "调用按值传递的子程序（初始）: m="; m, "n="; n '输出形参 m 和形参 n 的初始值  
03   m = 50 '为形参 m 重新赋值  
04   n = 100 '为形参 n 重新赋值  
05   Print "调用按值传递的子程序（子程序赋值以后）: m="; m, "n="; n '输出形参的新值  
06 End Sub
```



## 【运行结果】

保存程序,按【F5】快捷键,运行程序。分别单击【按地址传递】按钮和【按值传递】按钮,可以看到按地址传递和按值传递后数值的变化。



### 5.2.5 过程的嵌套与递归

在 Visual Basic 中,在一个过程中调用另外一个或其他若干个过程的方法称为过程的嵌套调用,在过程中直接或间接地调用过程自身的方法则称为过程的递归调用。

#### 1. 过程的嵌套调用

在 Visual Basic 中,过程的定义都是相互平行独立的,一个过程内不能包含另一个过程,在程序代码中每一个过程之间都用一条分割线隔开。

过程与过程之间  
是用线分开的



虽然不能嵌套定义过程,但在 Visual Basic 中提供有嵌套调用过程的方法,也就是主程序可以调用子过程,在子过程中还可以调用另外的子过程,这种程序结构称为过程的嵌套。

#### 2. 过程的递归调用

递归调用的方法有两种:直接递归和间接递归。

(1) 直接递归。直接递归的实例代码是:


```
Private Sub Sub1() ' 定义过程 Sub1
...
Call Sub1 ' 调用自身
```

```
...
End Sub
```

(2) 间接递归。间接递归的实例代码是：

```
Private Sub SubA() '定义过程 SubA
...
Call SubB '调用 SubB
...
End Sub
Private Sub SubB() '定义过程 SubB
...
Call SubA '调用 SubA
...
End Sub
```

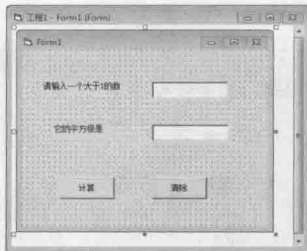
### 【范例 5-4】定义一个计算平方根的子过程，并在子过程中调用自己，以实现对其个数进行平方根的运算。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中添加两个命令按钮控件、两个标签控件和两个文本框控件，将两个文本框的 Text 属性值设置为空，将两个命令按钮控件和两个标签控件的 Caption 属性值按下表进行设置。

名称	Caption
Command1	计算
Command2	清除
Label1	请输入一个大于 1 的数
Label2	它的平方根是

(3) 将 Form1 窗体上的控件排列整齐，如图所示。



(4) 在 Form1 窗体中双击【计算】按钮，打开代码窗口，输入以下代码（代码 5-4-1.txt）。

```
01 Private Sub Command1_Click()
02 Dim x, x1, x2 As Single
```

```

03  x1 = 1 ' 为变量 x1 赋初值
04  x2 = Val(Text1.Text) ' 为变量 x2 赋初值, 并将其转换成数值型
05  x = x2 ' 将变量 x2 的值赋给变量 x
06  Call sqrt(x1, x2, x) ' 调用计算平方根的子过程
07  Text2.Text = x ' 将结果显示在 Text2 中
08  End Sub

```

(5) 在 Form1 窗体中双击【清除】按钮, 打开代码窗口, 输入以下代码。

```

01 Private Sub Command2_Click()
02  Text1.Text = "" ' 清空文本框 1 中的内容
03  Text2.Text = "" ' 清空文本框 2 中的内容
04 End Sub

```

(6) 在代码窗口中编写 sqrt 的子过程 (代码 5-4-2.txt)。

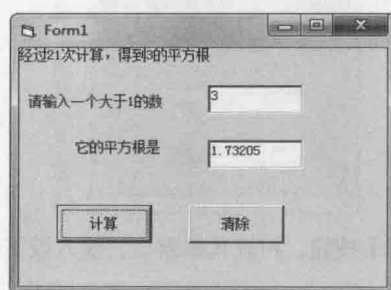
```

01 Private Sub sqrt(y1, y2, y)
02  Dim n As Integer ' 定义变量 n, 用于记录计算的次数
03  Dim y3 As Single ' 定义变量 y3, 用于接收计算后的值
04  Do While Abs(y1 - y2) > 0.000001 ' 判断是否大于 0.000001, 小于则退出循环体
05  y3 = (y1 + y2) / 2 ' 将传递后的 y1 与 y2 相加并对 2 求模
06  If (y3 * y3 - y) <= 0 Then ' 判断平方根值是否小于等于零
07  y1 = y3 ' 小于等于 0 则将平方根的值赋予 y1
08  Else
09  y2 = y3 ' 大于则将平方根的值赋予 y2
10  End If
11  n = n + 1 ' 计算的次数加 1
12  Loop
13  y = y3 ' 得到结果, 赋值给形参
14  Print " 经过 " & n & " 次计算, 得到 " & Text1.Text & " 的平方根 " ' 输出最终结果
15 End Sub

```


## 【运行结果】

保存程序, 按【F5】快捷键, 运行程序。在【请输入一个大于 1 的数】文本框中输入数值“3”, 单击【计算】按钮, 结果如图所示。

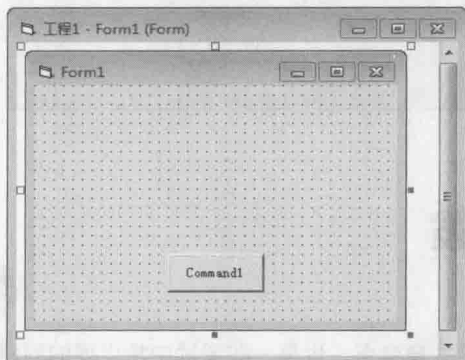


### 【拓展训练 5-3】

阶乘运算的方法很多，在这里使用递归的方法进行一次阶乘的运算。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 窗体上添加一个命令按钮控件。



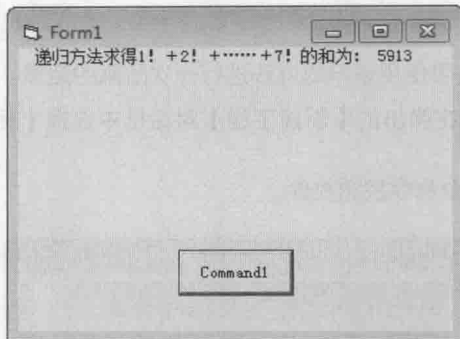
(3) 双击 Form1 窗体上名称为 Command1 的命令按钮控件，打开代码窗口，输入以下代码（拓展代码 5-3-1.txt）。

```
01 Private Sub Command1_Click()
02   Dim s As Long '定义长整型变量，用于接收 7 的阶乘值
03   Dim i As Integer '定义整型变量，用于计数循环
04   s = 0 '为变量 s 赋初值
05   For i = 1 To 7 Step 1 '进行阶乘运算循环
06     s = s + Factorial(i) '调用 Factorial 函数过程
07   Next i
08   Print "用递归方法求得 1! + 2! + ... + 7! 的和为: "; s '输出最终的结果
09 End Sub
```

(4) 在代码窗口中编写 Factorial 函数过程（拓展代码 5-3-2.txt）。

```
01 Public Function Factorial(ByVal n As Integer) As Long
02   If n = 1 Then '如果接收到的数为 1
03     Factorial = 1 '返回 1 的阶乘值
04   Else '否则递归调用函数过程
05     Factorial = Factorial(n - 1) * n '返回值不为 1 的其他阶乘值
06   End If
07 End Function
```

(5) 保存程序，按【F5】快捷键运行程序，单击【Command1】命令按钮，结果如图所示。



## 5.3 高手点拨



本节视频教学录像: 3 分钟

本章中有两个问题, 不但是本章难点, 也是一些考试中常出现的问题, 需要我们格外注意。

### 1. 确定自定义的过程是子过程还是函数过程

一般来说, 函数过程名有值, 子过程名无值。有一个返回值, 则使用函数过程; 有多个返回值或无返回值, 一般使用子过程。

### 2. 实参与形参结合时对应问题

在使用实参与形成那对照时, 要注意个数、类型、位置、次序的一一对应。

形参是按值传递, 对应实参可以是表达式、常量、数组元素。

形参是按址传递, 对应实参只能是简单变量。

数组、记录类型、对象只能是按地址传递。

## 5.4 实战练习

### 一、思考题

1. 简述调用 Sub 子过程与 Function 函数过程的区别是什么。
2. 参数传递有哪几种, 它们的区别是什么?

### 二、操作题

使用 Function 函数实现两数平均值的计算。

# 第 2 篇

## 核心技术

本篇介绍可视化编程，窗体和系统对象，标准模块和类模块，标准控件，ActiveX 控件、工具栏和状态栏，鼠标、键盘事件，菜单和对话框设计以及程序调试与错误处理等内容。通过本篇的学习，读者能轻松掌握 Visual Basic 编程的核心技术。

第 6 章 应用程序的精髓——可视化编程

第 7 章 应用程序的脸——窗体和系统对象

第 8 章 标准模块和类模块

第 9 章 VB 的简易之道——标准控件

第 10 章 扩展你的需求——ActiveX 控件、工具栏和状态栏

第 11 章 鼠标、键盘的另类编程应用——鼠标、键盘事件

第 12 章 程序与用户的交互——菜单和对话框设计

第 13 章 编程错误终结者——程序调试与错误处理



# 第6章



本章视频教学录像：30 分钟

## 应用程序的精髓——可视化编程

面向对象技术是计算机软件工程中的区别与面向过程技术一种程序设计方法。在软件工程中，面向对象是一种新式的计算机程序设计架构，其原则是计算机程序由对象组成，其技术有其自身的评价参数和特点，它的设计思想更符合人们对现实世界的认识方式。本章将介绍对象和类的基本概念和如何使用对象。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握对象和类的含义
- ☐ 熟悉简单的对象的建立与编辑
- ☐ 掌握对象属性及其设置的方法
- ☐ 掌握对象的方法及其调用过程
- ☐ 熟悉事件的调用过程

## 6.1 对象概念



本节视频教学录像：15 分钟

对象和类是面向对象中的基本组成部分，对象是现实世界中具体的事物，而类是将一类事物进行抽象而得到的。对象是代码和数据的组合，可以作为一个单位来处理。对象可以是应用程序的一部分，比如可以是控件或窗体。整个应用程序也是一个对象。

VB 中的每个对象都是用类定义的。用饼干模子和饼干之间的关系做比较，就会明白对象和它的类之间的关系。饼干模子是类。它确定了每块饼干的特征，比如大小和形状。用类创建对象，对象就是饼干。类是面向对象程序设计的核心技术，可以理解成一种定义了对象行为和外观的模板；把对象看作是类的原原本本的复制品。

### 6.1.1 对象和类

究竟什么是对象和类呢？首先我们通过一个例子来认识一下。

比如“树木”是一个抽象的名称，类似一个类的概念。而梧桐树、杨树、苹果树等就是“树木”类的具体对象。这些种类的树木因为外部差异所以称为不同的对象，但是内部机理具备树木的普遍特性，所以被称为一类被称为“树木”物体。

对象的概念是面向对象编程技术的核心。所以从面向对象的观点看，所有的面向对象的应用程序都是有对象组合而成的。而类（class）是同类对象的属性和行为特征的抽象描述。

在现实生活中，对象指的是具体的事物，如天上的卫星、地上的树木、海里的轮船等，每个客体都具有一些属性和行为，例如学生有学号、姓名、性别等属性，有上课、考试、做实验等行为。因此，每个个体都可以用属性和行为来描述。将对象的内部状态称为属性，将其行为称为方法和事件。对象之间的联系通过消息来传递，消息机制是对象间相互联系和相互作用的方式。

在 Visual Basic 6.0 中，系统为每一类对象都规定了若干属性。设计中可以改变具体对象的属性值。窗体的任何一个对象都有属性、事件和方法 3 个要素，它们各自从不同的角度表达了对象的构成，通过三者的有机结合，便构成 Visual Basic 应用程序的基本元素。也可以说，一个完整的 Visual Basic 应用程序就是若干个对象集合而成的，而每个对象又是通过属性、事件及方法构成的。



**提示**

类和对象是面向对象程序设计的语言基础。类是从相同类型的对象中抽象出来的一种数据类型，也可以说是所有具有相同数据结构、相同操作的对象的抽象。类的构成不仅包含描述对象属性的数据，还有对这些数据进行操作的事件代码，即对象的行为（或操作）。

### 6.1.2 VB 中对象的建立和编辑

在“工程设计”窗口，打开“代码”窗口有以下方法。

(1) 依次选择“视图”→“代码窗口”菜单选项，打开“代码窗口”。

(2) 选中某一对象，右击鼠标，打开快捷菜单，选择“代码窗口”菜单选项，打开“代码”窗口。

在“代码”窗口，首先通过“对象”组合框提供的参数选择对象，然后再通过“事件”组合框提供

的参数选择事件, 这时系统自动给出事件过程开头和结束语句。

例如:

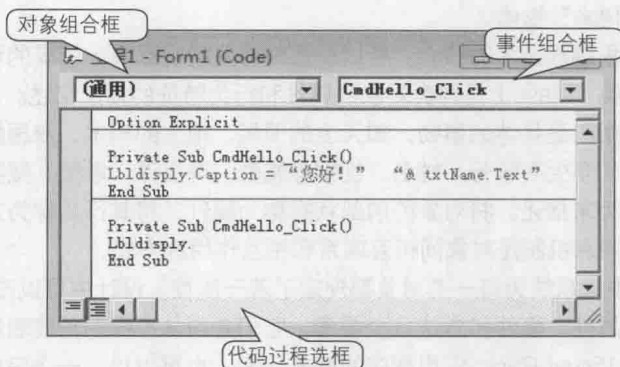
```
01 Private Sub Cmd1_Click ( )
02 End Sub
```

接下来便可以在上面开头和结束语句之间输入过程代码。在 Visual Basic 系统中, 过程代码是针对具体对象事件编写的, 为了确切得知某个对象的“操作”, 必须在方法和属性名前加上对象名, 中间用小数点 (.) 分隔。

例如:

```
01 Private Sub CmdHello_Click()
02 LblDisplay.Caption = "您好! " & txtName.Text
03 End Sub
```

在进行过程代码的编写时, Visual Basic 系统为用户提供了十分方便的定义对象属性、方法的操作, 当键入对象名后, 再键入“.”, 系统会自动弹出和对对象相关的属性、方法列表框, 用户可从中选择使用的方法, 如下图所示。



## 6.2 对象的属性、方法和事件



本节视频教学录像: 13 分钟

在面向对象的程序设计中, 对象具有属性、方法和事件。属性用来描述对象的特征, 方法告诉我们对对象应做的事情, 而事件是对象所产生的事情, 事件发生时可以编写代码进行处理。在 Visual Basic 6.0 中, 每一个窗体和控件都具有自己的属性、方法和事件的对象。可以把属性看作一个对象的性质, 把方法看作对象的动作, 把事件看作对象的响应。在程序设计中, 基本的设计机制是, 改变对象的属性、使用对象的方法、为对象事件编写事件过程。程序设计时要做的工作就是决定应更改哪些属性、调用哪些方法、对哪些事件作出响应, 从而得到希望的行为。

### 6.2.1 对象的属性及设置

在 Visual Basic 系统中, 各种对象拥有几十个属性。对象的属性可以在设计对象时通过属性窗口设


置，也可以在程序运行时通过事件代码来进行设置。

### 1. 利用“属性”窗口设置对象属性

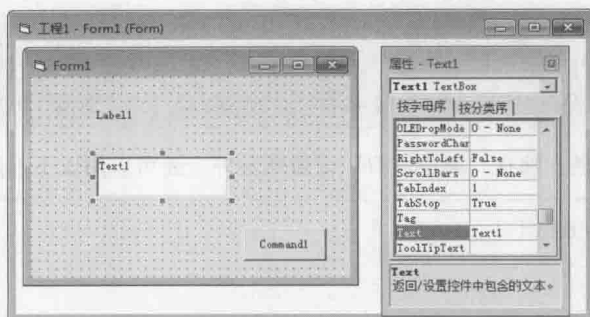
在“工程设计”窗口，有三种打开“属性”窗口的方法。

(1) 选择窗口上的“视图”→“属性窗口”菜单选项，打开“属性”窗口。

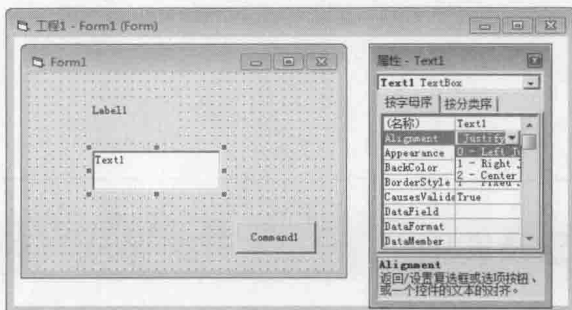
(2) 选中设置属性的“对象”，右击鼠标，打开快捷菜单，选择“属性窗口”菜单选项，打开“属性”窗口。

(3) 选中设置属性的“对象”，单击工具栏中的按钮，打开“属性”窗口。

在属性窗口，可以直接为对象设置属性，如图所示。



也可以通过组合框提供的参数选择对象属性，如图所示。



还可以通过对话框为对象设置属性，如图所示。



## 2. 利用属性设置为对象设置属性

属性设置语句格式 1:

[< 父类名 >].< 对象名 >. 属性名 =< 属性值 >

属性设置语句格式 2:

With < 对象名 >

< 属性值表 >

End with



**提示**

在 Visual Basic 中, 某些对象的若干属性不能通过程序语句设置, 只能在“属性”窗口设置; 还有些属性是只读属性, 只能继承, 不能改动; 另外有些对象(尤其后面讲到的 ActiveX 控件), 个别的属性窗口没有出现, 必须在程序语句设置或在专门的“属性”窗口设置。

如下是命令按钮的属性, 除在“属性窗口”设置属性外, 还可以用以下程序语句设置:

```
01 Private Sub Form_Load ( )
02 With Cmd1
03 caption = " 关闭 "
04 Left = 4680
05 Top = 5280
06 Height = 615
07 Width = 1335
08 End With
09 End Sub
```

## 6.2.2 对象的方法及调用

对象的方法, 是指控制对象动作行为的方式。方法不同于事件, 方法是对象本身内含有的函数和过程, 是一个简单、用户不需要知道细节、用户无法改变的一个动作。而事件是由用户定义的动作, 这个动作用户需要了解细节而且是可以改变的, 每一类对象都可能有一些自身特定的方法。在 Visual Basic 6.0 中, 方法的调用形式是:

对象名 . 对象名

例如, Adodc1.Refresh, 刷新的方法 Refresh 是数据连接对象 Adodc 的一个方法; Form1.Hide 也表示 Hide 是窗体对象 Form 的一个方法。

对象的方法是对象内部已经确定的。

### 6.2.3 对象的事件及事件过程

事件，指的是发生在对象上的动作。事件的发生并非随意的，某些事件仅发生在某个时间或某些对象上。在 Visual Basic 系统中，一个对象可以识别和响应一个或者多个事件，这些事件的代码是通过“事件过程”定义的。

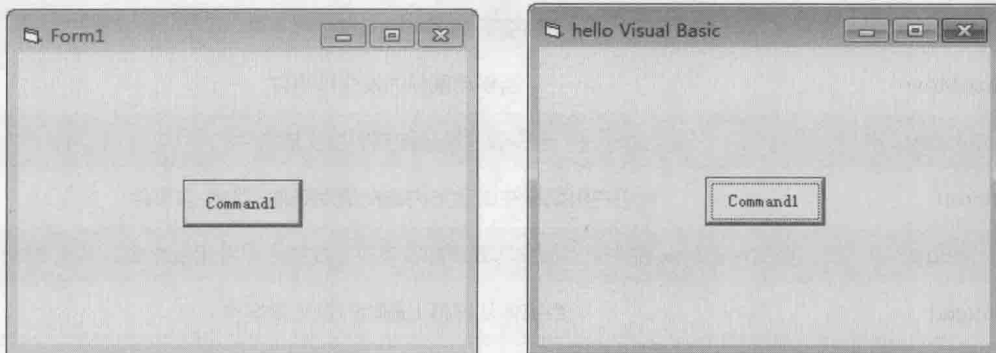
在 Visual Basic 6.0 中，事件的定义方式为：

```
Private Sub 对象名 - 事件名称
```

```
(事件内容)
```

```
End Sub
```

例如，当用户单击 Command1 按钮时，显示文字信息“hello Visual Basic”，如下图所示。



在设计界面上添加一个 command Button，双击 Command1，进入代码编辑区，输入以下代码。

```
01 Private Sub Command1_Click()
```

```
02 Form1.Caption = " hello Visual Basic " '设置对象的标题为 " hello Visual Basic "
```

```
03 End Sub
```

## 6.3 高手点拨



本节视频教学录像：2 分钟

1. “对象名称”指的是对象（名称）属性定义的标识符，这一属性必须在“属性”窗口进行定义。

2. “事件名称”是由 Visual Basic 系统定义好的某一对象能够识别和响应的事件。

3. “程序代码”是 Visual Basic 提供的操作语句及特定的方法。

在 Visual Basic 系统中，对象可以响应的事件有很多，在多数情况下，事件是通过用户的操作行为引发的（如单击鼠标、移动鼠标、按键等）。当事件发生时，将执行包含在事件过程中的全部代码。

事件有的适用于专门控件，有的适用于多种控件，下表列出 Visual Basic 系统中的核心事件。



事 件	触发事件的操作
Click	当鼠标单击某个对象时发生该事件
DblClick	当在窗体上用鼠标拖动一个控件然后放开时发生该事件
GotFocus	当对象获得焦点时，发生该事件
KeyPress	在键盘上按下并松开键盘上一个键时发生该事件
KeyUp	当一个对象具有焦点并释放一个键时发生该事件
Load	当窗体被装载时发生该事件
LostFocus	当对象失去焦点时发生该事件
MouseDown	当按下鼠标按钮时发生该事件
MouseMove	当移动鼠标时发生该事件
MouseUP	当移释放鼠标按钮时发生该事件
Scroll	当用户用鼠标在滚动条内拖动滚动条框时发生该事件
Selchange	RichTextBox 控件中当前文本的选择发生改变或插入点发生变化时，发生该事件
Unload	当窗体从屏幕上删除时发生该事件

## 6.4 实战练习

### 一、思考题

试总结对象的属性、方法、事件的关系。

### 二、操作题

设计一个窗体，当单击“显示”按钮时，在文本框内显示“Hello Visual Basic”；当单击“清除”按钮时，清除文本框内的内容；当单击“退出”按钮时，结束程序运行。

# 第

# 7

# 章



本章视频教学录像：1 小时 5 分钟

## 应用程序的脸——窗体和系统对象

窗体是应用程序直接与用户交互的主要界面，是应用程序最重要的一部分，所以窗体的设计显得尤为重要。具有相同功能的两个应用程序，良好的窗体设计会带给用户带来更好的使用体验。

本章将一步步带领读者熟悉窗体的相关概念及属性，掌握窗体设计的基本方法，以及学会设计多窗体应用程序。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉窗体的基本概念
- ☐ 熟悉窗体的属性、事件和方法
- ☐ 掌握窗体的启动和结束
- ☐ 了解多窗体设计
- ☐ 了解多窗体特性

## 7.1 窗体简介



本节视频教学录像: 5 分钟

图形界面的 Windows 操作系统诞生以来, 软件的编写又多了一项重要部分——用户界面的设计。设计良好的软件界面能够让用户感觉舒适并易于使用。在 Visual Basic 的软件设计中, 好的窗体设计是成功的一半。

### 7.1.1 窗体的基本概念

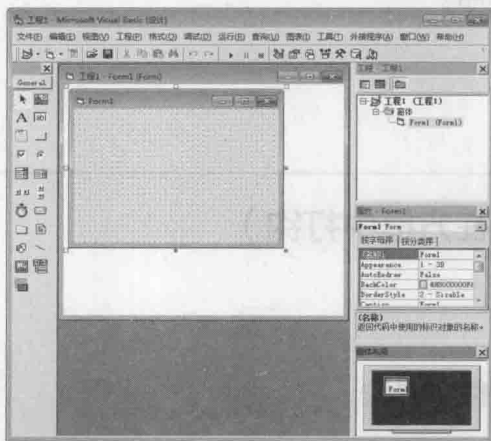
我们日常使用的 Windows 操作系统又称为视窗操作系统, 它是由一个个窗体所组成的。我们对计算机的所有操作几乎都是通过各种窗体来完成的, 所以窗体既是我们完成各种计算机操作的载体, 又是我们与计算机进行沟通的主要方式。一个优秀软件的界面设计是非常重要的。在 Visual Basic 中要设计一个窗体, 首先要了解的概念是它的属性、事件和方法。



**提示**

Windows 在英文中的原意是“窗户、视窗”。在 Windows 系统诞生之前, 计算机上看到的只是枯燥的字幕数字 (DOS), 微软公司开发的 Windows 操作系统, 打开了用户与计算机之间交流的“视窗”, 使我们能够以更加直观和方便的方式使用计算机。

打开 Visual Basic 并新建一个工程后, 在打开的界面上就会加载一个默认窗体, 我们把各种需要的控件放在这个窗体上并编写每个控件的代码, 即可完成应用程序设计。



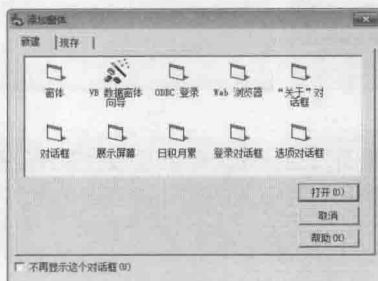
### 7.1.2 在工程中添加窗体的方法

大部分的软件都拥有多个窗口, 因此仅有默认的窗体是不够的。如果需要创建更多的窗体, 可以使用以下两种方法。

- (1) 选择【工程】>【添加窗体】菜单命令。
- (2) 在工程资源管理器窗口中右击, 在弹出的快捷菜单中选择【添加】>【添加窗体】菜单命令。



不管使用哪种方式，都会打开【添加窗体】对话框，从中选择需要的窗体类型，然后单击【打开】按钮，即可将该窗体添加到工程中。



提示

根据窗体的功能，可以将窗体分为单文档窗体（SDI）和多文档窗体（MDI）。每个单文档窗体均是一个独立的窗口，而每个多文档窗体中可以包含多个窗口。

## 7.2 控制窗体表情——窗体的属性、方法和事件



本节视频教学录像：18 分钟

创建出窗体后，接下来需要根据软件设计的需要对这个窗体的各种属性进行设置。可以直接在程序源代码中编写和修改窗体的属性，但是更常用和方便的方法是通过属性窗口设置。

### 7.2.1 窗体的属性

下面介绍一些最常用的窗口属性的意义和设置方法。

#### 1. 标题属性（Caption）

标题属性就是用来设置窗体标题所显示的文字的。标题属性既可以通过代码来设置，也可以直接在属性窗口中设置。

在代码中设置标题属性的语法是：

```
窗体名.Caption = " String "
```

其中 String 就是希望标题所显示的文字。

在属性窗口中的设置更为简单：直接在属性窗口中的 Caption 属性所对应的输入框中输入字符即可。

例如有一个窗体 Form1，要设置窗体标题为“Hello!”，则可在属性窗口中设置 Caption 值为“Hello!”。

设置完成，按【F5】快捷键运行程序，实际效果如下图所示。



**提示**

使用这两种方法都可以设置窗口的标题，实际运用中应根据不同情况使用不同的方法。对于在运行中不需要改变标题的窗口，可以采用第2种方法；而对于需要在运行中更改标题的窗口，则应该使用第1种方法。

## 2. 外观属性 (Appearance)

外观属性中只有两个选项，分别是 Flat 和 3D，Flat 的意思是“平的”，而 3D 则是常见的三维立体效果。

对比可见，3D 属性的立体感更强一些。外观属性的设置和标题属性的设置类似，直接在属性窗口中选择相应的属性即可。



## 3. 边框样式属性 (BorderStyle)

边框样式属性决定了窗体的边框外观和功能。可以通过属性窗口来对其边框样式进行设置。



可以看到，边框样式属性拥有 6 个选项，含义如表所示。

属性名称	对应值	含义
vbBSNone	0	没有边框或者与边框相关的元素
vbFixedSingle	1	固定边框。可以包含控制菜单框、标题栏、最大（最小）化按钮，只能通过【最大化】和【最小化】按钮改变窗体大小
vbSizable	2	该值为系统默认值，在运行时可以通过鼠标调节窗体边框大小
vbFixedDialog	3	固定对话框，可包含控制菜单框、标题栏，但不包含最大（最小）化按钮，不能改变窗体大小
vbFixedToolWindows	4	固定窗口，不能改变窗口大小，显示【关闭】按钮，并用缩小字体显示标题栏
vbSizableToolWindows	5	可变尺寸窗口，显示【关闭】按钮，并用缩小字体显示标题栏

#### 4. ControlBox 属性

ControlBox 属性用于设置窗体是否显示控制菜单和【最大化】、【最小化】以及【关闭】等按钮。当 ControlBox 属性为 True 时，将显示控制菜单和【最大化】、【最小化】以及【关闭】等按钮；为 False 时不显示。



**提示**

在 Windows 操作系统中，每个应用程序都有一个控制菜单，提供还原、移动、调整大小、最大化、最小化和关闭窗口等功能。

右击标题栏的任意位置，或单击标题栏左侧的图标，都可以弹出控制菜单。同时按住【Alt+ 空格】键也可以弹出控制菜单。同一程序的控制菜单具有相同的菜单命令。



对 ControlBox 的属性可以通过属性窗口来设置。

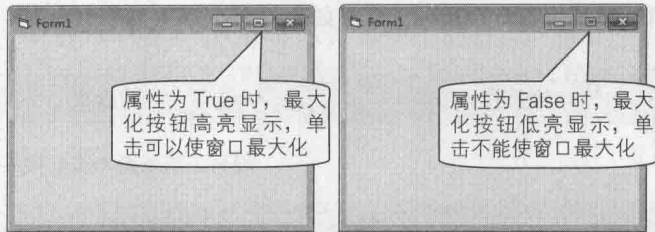
#### 5. 最大化按钮属性 (MaxButton)

最大化按钮属性决定最大化按钮是否起作用。

(1) 当最大化按钮属性为 True 时，最大化按钮高亮度显示，单击可以使窗口最大化。

(2) 当最大化按钮属性为 False 时，最大化按钮低亮度显示，单击不能使窗口最大化。



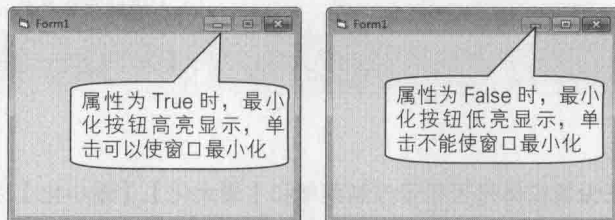


对 MaxButton 的属性可以通过属性窗口来设置。

#### 6. 最小化按钮属性 (MinButton)

该属性决定最小化按钮是否起作用。

- (1) 当最小化按钮属性为 True 时，最小化按钮高亮度显示，单击可以使窗口最小化。
- (2) 当最小化按钮属性为 False 时，最小化按钮低亮度显示，单击不能使窗口最小化。



可以通过属性窗口来设置最小化按钮属性。

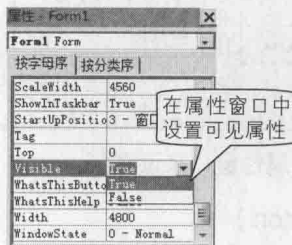


如果想显示最小化按钮，必须将 BorderStyle 属性值设置为 1 或者 2。这样程序运行时，窗体最小化后，窗体最小化显示为图标。

#### 提示

#### 7. 可见属性 (Visible)


该属性决定窗体或者窗体上的控件在运行时是否可见。当可见属性为 True 时，程序运行后将显示该窗体或控件；当可见属性为 False 时，程序运行后将不会显示该窗体或者控件。可以通过属性窗口来设置可见属性。



#### 8. 图标属性 (Icon)

通过设置该属性可以改变窗体最小化时所显示的图标，该图标也将显示在窗体标题栏左侧。可以通过属性窗口来设置图标属性。



如果你没有设置图标也没有关系，Visual Basic 6.0 将为其设置一个默认图标：。但如果所有的应用程序都采用这种默认的图标，那么在较多的程序中找到需要的窗口就很不方便，所以应该适当地为程序指定意义鲜明的图标。



#### 提示

对于窗体的属性，可以实际动手设置一下，看看选择不同的属性值后有什么不一样的地方，通过实际操作可以对这些属性值有更切身的感受。

## 7.2.2 窗体的方法

窗体的方法是指对窗体的各种操作，如“显示窗体”方法（Show）、“隐藏窗体”方法（Hide）、“移动窗体”方法（Move）、“显示弹出式菜单”方法（PopupMenu）和“打印窗体”方法（PrintForm）等。现将常用的方法介绍如下。

### 1. “显示窗体”方法（Show）

“显示窗体”方法用来显示窗体，其语法格式如下。

```
[Object].Show [Model],[OwnerForm]
```

其中，Object 用于指定窗体对象，Model 用于指定窗体是模式窗体还是无模式窗体。Model 值为 0 时，窗体为无模式窗体；值为 1 时，窗体为模式窗体。

需要注意的是，模式窗体显示后一直占据焦点，其他窗体暂时不可用，所以使用模式窗体来实现多窗体有严格操作步骤的程序，如软件安装程序。

OwnerForm 用于指定窗体对象，指定的窗体可以看作是 Object 窗体的父窗体。



#### 提示

Object 窗体始终位于指定窗体前面，且关闭或最小化 OwnerForm 窗体时，Object 窗体也随之关闭或最小化。

### 2. “隐藏窗体”方法（Hide）

大部分的软件一般含有多个窗体，分别用于实现不同的功能。在很多时候并不需要显示所有的窗口，使用隐藏窗体功能可以为工作区域节省很多屏幕空间。

“隐藏窗体”方法用于隐藏窗体但不关闭窗口，其语法格式如下。

```
[Object].Hide 'Object 用于指定窗体对象
```

### 7.2.3 窗体的事件

事件驱动的编程机制是 Visual Basic 的一大特点。窗体中提供有很多的事件来表达各种触发条件, 如表所示。

事件名称	触发条件	事件处理过程	备注
Click	鼠标单击窗体的空白区或无效控件时被触发	Private Sub Form_Click()	若用户已经定义了 Click 事件, 则 DblClick 事件永远不会被触发
DblClick	鼠标双击窗体的空白区或无效控件时被触发	Private Sub Form_DblClick()	
Initialize	创建窗体实例时被触发	Private Sub Form_Initialize()	初始化窗体所用数据
Load	当窗体被程序装载时被触发	Private Sub Form_Load()	窗体的启动代码
Unload	窗体从屏幕中删除	Private Sub Form_Unload (Cancel As Integer)	Cancel 值为 0 时, 窗体将被删除; 通常将 Cancel 的值设置为非零值, 以防止窗体被删除
Activate	在窗体成为活动窗口时被触发	Private Sub Form_Activate()	这个事件只在所在应用程序中焦点移动时发生
Deactivate	窗体不再成为活动窗口时被触发	Private Sub Form_Deactivate()	这个事件只在所在应用程序中焦点移动时发生
Paint	窗体移动或放大后, 窗体部分或全部暴露时被触发	Private Sub Form_Paint()	该方法确保图形方法输出在必要时重绘
Resize	窗体第 1 次显示或者窗体状态改变时被触发	Private Sub Form_Resize()	
KeyDown	窗体具有焦点且有一个键被按下时被触发	Private Sub Form_KeyDown (KeyCode As Integer, Shift As Integer)	KeyCode 用于返回键代码; Shift 用来响应【Shift】、【Ctrl】和【Alt】键
KeyUp	窗体具有焦点且有一个键被释放时被触发	Private Sub Form_KeyUp (KeyCode As Integer, Shift As Integer)	
KeyPress	按下或者松开一个 ANSI 键时被触发	Private Sub Form_KeyPress (KeyAscii As Integer)	KeyAscii 用于返回标准数字 ANSI 键代码

续表

事件名称	触发条件	事件处理过程	备注
MouseDown	在窗体上按下鼠标时被触发	Private Sub Form_MouseDown (Button As Integer, Shift As Integer, X As Single, Y As Single)	Button 返回值, 以标识鼠标的哪个键被按下; X、Y 用于返回鼠标指针当前位置
MouseMove	在窗体上移动鼠标时被触发	Private Sub Form_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)	
MouseUp	在窗体上释放鼠标时被触发	Private Sub Form_MouseUp (Button As Integer, Shift As Integer, X As Single, Y As Single)	

例如：有一个窗体 Form1，要在加载窗体时设置窗体标题为“此窗体已加载”，则可添加如下代码。

```
01 Private Sub Form_Load() ' 窗体加载事件
02     Form1.Caption = " 此窗体已加载 " 设置 Form1 的 Caption 值
03 End Sub
```

程序运行效果如图所示。



## 7.3 窗体的生命周期



本节视频教学录像：7 分钟

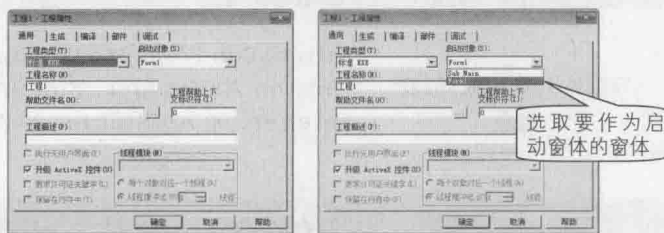
窗体的启动、运行和关闭，是窗体由“生”到“亡”的一个生命周期。下面就来了解一下窗体的“一生”过程。

### 7.3.1 选择启动窗体

当我们建立一个应用程序的时候，应用程序中的第 1 个窗体默认被指定为启动窗体。应用程序开始运行时，这个窗体就会显示出来。可以通过改变启动窗体来使其他窗体在启动时显示出来，操作步骤如下。

(1) 选择【工程】>【属性】菜单命令，弹出【工程属性】对话框。

(2) 在【通用】选项卡中的【启动对象】下拉列表中，选取要作为启动窗体的窗体。



(3) 单击【确定】按钮，即可完成更改启动窗体的操作。



**注意**

选择的启动对象一定要是已经存在的，如果选择了不存在的启动对象，在编译运行时会出现错误，并弹出【工程属性】对话框，要求重新选择启动对象。

## 7.3.2 快速显示窗体

如果启动时有一个较长的执行过程，例如要从数据库中装入大量的数据，或者要装入一些大型位图，这时如果能够提供一个快速显示窗体，将会消除用户等待时的焦急感并了解一些软件使用信息。



**技巧**

快速显示是一种窗体，它通常显示的是应用程序名、版权信息或者一个简单的位图之类的内容。启动 Word 2007 时所显示的屏幕就是一个快速显示。



要显示快速显示窗体，需要用 Sub Main 过程作为启动对象，并用 Show 方法显示该窗体。

```
01 Private Sub Main()
02     frmSplash.Show      ' 显示快速显示
03     ...                 ' 在此处添加启动过程
04     frmMain.Show        ' 显示主窗体并卸载快速显示
05     Unload frmSplash
06 End Sub
```

启动过程完成，可以装入第 1 个窗体并使快速显示卸载。

## 7.3.3 结束窗体

在很多人的印象里，当一个应用程序的所有窗体都关闭的时候，就意味着这个程序停止运行了。然而事实并非如此，很多程序虽然没有任何窗体显示了，但是在后台仍然默默地运行着。出现这种情



况的原因是对任何已卸载窗体的属性或控件的任何访问，都将导致隐含地、不予显示地加载那个窗体。

为了保证在关闭所有的窗体后程序确实已经停止运行，最好的办法是确保所有的窗体都被卸载。如果应用程序只有一个窗体，则在主窗体上可以用一个名为 cmdQuit 的命令按钮退出程序，Click 事件过程代码如下。

```
01 Private Sub cmdQuit_Click ()
02     Unload Me      ' 卸载窗体
03 End Sub
```

如果有 1 个以上的窗体，可以使用 Forms 集合和 Unload 语句，通过把代码放入主窗体的 Unload 事件过程卸载这些窗体。

通过使用 Forms 集合可以确保找到并关闭所有窗体。下列代码就是使用窗体集合来卸载所有窗体的。

```
01 Private Sub Form_Unload (Cancel As Integer)
02     Dim i as integer
03     For i = Forms.Count - 1 to 0 Step - 1      ' 在窗体集合中遍历并卸载每个窗体
04         Unload Forms(i)
05     Next
06 End Sub
```

有的时候我们需要不顾现存窗体或对象的状态而强制结束应用程序，此时可以使用 Visual Basic 中的 End 语句。

End 语句用于使应用程序立即结束。在 End 语句之后的代码不会执行，任何窗体的 QueryUnload、Unload 或 Terminate 等事件过程也不会被执行，对象的各个引用将被释放，也不会再有事件发生。

## 7.4 多窗体设计



本节视频教学录像：5 分钟

多文档界面（Multiple Document Interface，MDI）和单文档界面（Single Document Interface，SDI）是 Windows 应用程序最为常见的两种结构。MDI 是在一个应用程序中能够同时处理两个或者两个以上窗体的界面形式，利用鼠标单击某个文档，该文档则成为活动文档，也称多文档程序。



技 巧


多文档界面和单文档界面的不同之处在于：SDI 中的各个窗体是相互独立的，例如 IE 浏览器；而 MDI 中的窗体则有主从关系，例如 Excel 软件。

### 7.4.1 创建多窗体应用程序

创建 MDI 应用程序，首先要在工程中创建一个 MDI 窗体，然后在这个 MDI 窗体中创建多个子窗体。



具体的操作步骤如下。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 选择【工程】下拉菜单中的【添加 MDI 窗体】选项, 弹出【添加 MDI 窗体】对话框, 从中选择新建或者现存的 MDI 窗体, 在这里选择新建 MDI 窗体, 然后单击【打开】按钮。

(3) 弹出一个颜色为深褐色的名为 MDIForm1 的新窗体。



(4) 选择 Form1 窗体, 将属性窗口中的 MDIChild 属性设置为【True】, 即将 Form1 窗体设置为 MDIForm1 窗体的子窗体。

(5) 运行程序, Form1 窗体已经在 MDIForm1 窗体中了。



## 7.4.2 多窗体特性

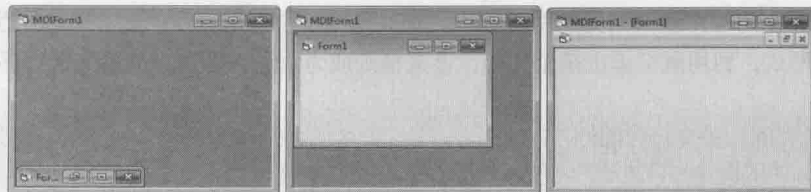
多窗体工程中的子窗体具有以下一些特性。

(1) 所有子窗体均只在主窗体的内部显示, 用户在主窗体中可以任意地移动和改变子窗体的大小, 但是不能将子窗体移动到主窗体之外。

(2) 子窗体最小化后, 其图标将出现在主窗体的最下方, 而不是出现在用户桌面上。

(3) 子窗体未最大化时, 各个子窗体具有自己的标题。

(4) 最大化后, 子窗体的标题将和主窗体的标题合并, 并在主标题栏中显示出来。




## 7.5 登录窗体设计实例



本节视频教学录像: 15 分钟

本节通过一个实例来加深对窗体程序设计的理解。

**【范例 7-1】** 用户登录界面设计，实例运行后将显示类似软件登录界面的效果，输入正确的用户名和密码后，即可显示主窗体；如果用户名和密码不正确，将显示警告窗口，要求重新输入。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性改为“用户登录”。



(3) 在窗体中添加两个 Command Button 控件、两个 Label 控件和两个 TextBox 控件。属性设置如表所示。

名称	Text/Caption 属性	用途
Cmd1	确定	进入主界面
Cmd2	取消	退出程序
Txt1	空字符串	用于输入用户名
Txt2	空字符串	用于输入用户名密码
Label1	用户名：	显示文本“用户名：”
Label2	密 码：	显示文本“密 码：”



(4) 选择【工程】菜单下的【添加窗体】菜单项，在弹出的【添加窗体】对话框中选择【窗体】为工程添加一个窗体 Form2。

(5) 更改 Form2 的 Caption 属性为“系统主界面”，ControlBox 属性值为“False”。在 Form2 中添加文本框控件，设置该控件的【名称】属性为“Txt1”，Text 属性值为“欢迎你进入系统”；添加命令按钮控件，设置该控件的【名称】属性为“Cmd3”，Caption 属性值为“退出”。将 Form2 中的控件拖动到合适位置并调整大小。



(6) 双击 Form1 窗体空白处, 在弹出的代码窗口中输入以下代码 (代码 7-1-1.txt)。

```
01 Private Sub Form_Load() ' 窗体 1 加载事件代码
02     Form1.Show
03     Form2.Hide
04 End Sub
05 Private Sub Txt1_KeyDown(KeyCode As Integer, Shift As Integer)
    ' 用户名文本框 KeyDown 事件代码
06     If Txt1.Text = "user" And KeyCode = vbKeyReturn Then
        ' 若用户名文本框输入正确且 Enter 键被按下
07         Txt2.SetFocus ' 密码文本框激活
08     ElseIf Txt1.Text <> "user" And KeyCode = vbKeyReturn Then ' 若用户名文本框输入不正确
09         MsgBox "请输入正确的用户名!", vbOKOnly + vbInformation, "注意" ' 显示警告窗口
10     End If
11 End Sub
12 Private Sub Txt2_KeyDown(KeyCode As Integer, Shift As Integer)
    ' 密码文本框 KeyDown 事件代码
13     If Txt2.Text = "123456" And KeyCode = vbKeyReturn Then
        ' 若密码文本框输入正确且 Enter 键被按下
14         Form1.Hide ' 窗体 1 隐藏
15         Form2.Show
16     ElseIf Txt2.Text <> "123456" And KeyCode = vbKeyReturn Then
        ' 若密码文本框输入不正确
17         MsgBox "请输入正确的密码!", vbOKOnly + vbInformation, "注意" ' 显示警告窗口
18     End If
19 End Sub
```

(7) 双击 Form1 窗体中的【确定】按钮控件, 在弹出的代码窗口中输入以下代码 (代码 7-1-2.txt)。

```
01 Private Sub Cmd1_Click() ' 确定按钮鼠标单击事件代码
02     If Txt1.Text = "user" Then ' 若用户名文本框输入正确
03         Txt2.SetFocus ' 密码文本框激活
04     ElseIf Txt1.Text <> "user" Then ' 若用户名文本框输入不正确
05         MsgBox "请输入正确的用户名!", vbOKOnly + vbInformation, "注意" ' 显示警告窗口
06     End If
07     If Txt2.Text = "123456" Then ' 若密码文本框输入正确
08         Form1.Hide ' 窗体 1 隐藏
09         Form2.Show ' 窗体 2 显示
10     ElseIf Txt2.Text <> "123456" Then ' 若密码文本框输入不正确
11         MsgBox "请输入正确的密码!", vbOKOnly + vbInformation, "注意" ' 显示警告窗口
12     End If
13 End Sub
```

(8) 双击 Form1 中的【取消】按钮控件, 在弹出的代码窗口中输入以下代码。

```
01 Private Sub Cmd2_Click() '取消按钮鼠标单击事件代码
02 End '退出程序
03 End Sub
```

(9) 双击 Form2 中的【退出】按钮控件，在弹出的代码窗口中输入以下代码。

```
01 Private Sub Cmd3_Click() '退出按钮鼠标单击事件代码
02 End '退出程序
03 End Sub
```


## 【代码详解】

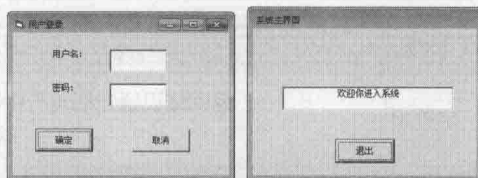
在第(6)步代码中，Form1.Show 用于显示 Form1，Form2.Hide 用于隐藏 Form2。

在第(7)步代码中用 If 判断函数来判断输入的用户名及密码是否正确，如果正确则显示 Form2，如果不正确则弹出警告窗口提示“请输入正确的用户名！”及“请输入正确的密码！”。

第(8)及第(9)步则是为 Form1 中的【取消】按钮控件及 Form2 中的【退出】按钮控件添加结束程序语句。

## 【运行结果】

单击工具栏中的【启动】按钮, 即可看到程序运行结果。



## 【范例分析】

通过本例，我们学习了在工程中添加窗体、在窗体中添加不同控件、改变窗体外观、控制窗体的显示与隐藏等方法。

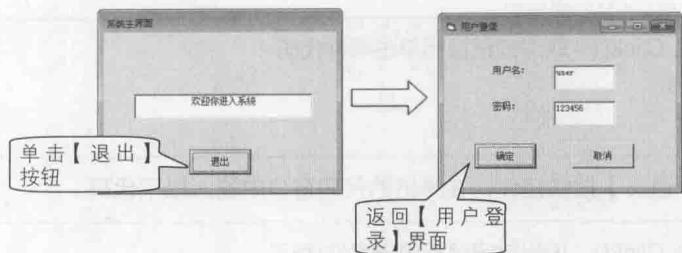
## 【拓展训练 7-1】用户登录界面设计 2。

在【范例 7-1】中，在关闭系统主界面后直接退出程序，如果想单击【退出】按钮后，再返回到用户登录界面中，应该怎么办呢？

只需修改 Form2 中的【退出】按钮为如下代码就可以实现。

```
01 Private Sub Cmd3_Click() '退出按钮鼠标单击事件代码
02 Form1.Show '窗体 1 显示
03 Unload Me '卸载窗体
04 End Sub
```

这样，单击【退出】按钮后，Form1 窗体将显示，而 Form2 窗体将隐藏。



## 7.6 系统对象



本节视频教学录像：12 分钟

系统对象指的 Visual Basic 提供的内置对象，主要包括 App 对象、Clipboard 对象、Screen 对象、Printer 对象等，能够在 Visual Basic 6.0 中被随时调用，完成一定的功能。本小节将对几种常用系统对象进行详细介绍。

### 7.6.1 应用程序 APP 对象

App 对象是通过关键字 App 访问的全局对象。其提供了十多个属性，包括应用程序的标题、版本信息、可执行文件和帮助文件的路径及名称以及是否运行前一个应用程序的示例。App 对象的主要属性如表所示。

属性	类型	说明
CompanyName	字符串	返回或设置包括运行中的应用程序的公司或创建名称，运行时只读
EXEName	字符串	返回当前正运行的可执行文件的根名（不带扩展名）。如果是在开发环境下运行，则返回工程名
Helpfile	字符串	确定 Microsoft Windows Help 文件的路径和文件名，应用程序使用这个文件显示 Help 或联机文档
Major, MinorRevision	数值	返回或设置应用程序的主要版本号、小版本号、订版本号，运行时只读
Previnstance	逻辑	返回一个值，该值指示是否已经有前一个应用程序实例在运行
TaskVisible	逻辑	返回或设置一个值，用来确定应用程序是否出现在窗口任务列表中

其中，App 对象的两种重要属性 App.Path 和 App.EXEName 应用较多。App.Path 在 Visual Basic 编程状态下返回 .MAK 文件所在目录；在以 .EXE 文件运行时，则返回 .EXE 文件所在目录。App.EXEName 返回程序名。

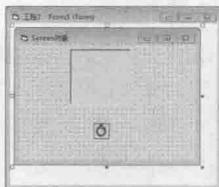
App 对象的典型应用在实现相对路径上。在 Visual Basic 中，程序总是按编程时固定好的路径读写文件，但文件路径改变，程序就找不到路径了，因此不能正常写文件。

## 7.6.2 屏幕 Screen 对象

Screen 对象指的是整个 Windows 桌面。Screen 对象可以访问当前窗体、控件以及其他与屏幕相关的信息。如屏幕鼠标指针的修改。下表为 Screen 对象的常见属性。

属性	说明
ActiveControl	返回拥有焦点的控件
ActiveForm	返回活动窗口的窗体
FontCount	返回或设置当前显示设备或者活动打印机可用的字体数
Fonts	返回当前设置显示器或者活动打印机可用的字体名
Height、Width	返回屏幕的高度和宽度；在设计时无效，在运行时为只读
MouseIcon	返回或设置自定义的鼠标图标
MousePointer	返回或设置鼠标指针的类型

例如实现在系统字体中搜索 Arial 字体，找到后给出提示。设计结果如下。



在窗体上添加图片框 Picture1 和时钟 Timer1 控件，将其 Interval 属性设置为 1000，双击图片框 Picture1 编写代码如下。

```

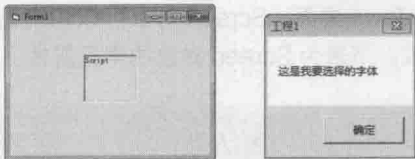
01 Private Sub Timer1_Timer()           ' 定时器的 Timer 事件
02 Static screenfontcount As Integer
03 If screenfontcount < Screen.FontCount Then
04 Screen.ActiveControl , Cls           ' 调用方法清除屏幕
05 Screen.ActiveControl.Print Screen.Fonts(screenfontcount) ' 调用方法显示字体
06 If Trim(Screen.Fonts(screenfontcount)) = "Arial" Then
07 Timer1.Enabled = False               ' 停止搜索
08 MsgBox " 这是我要选择的字体 "       ' 给出提示
09 Exit Sub                             ' 退出程序
10 End If
11 screenfontcount = screenfontcount + 1 ' 循环变量加 1
12 End If
13 End Sub

```

Timer 控件在后续章节会详细讲解。设置其 Interval 属性为 1000，即设置其每过 1000 毫秒执行一



次其中的代码。运行上述示例。其结果如下图所示。其中，Picture1 控件中的字体名称一直会变化，直到找到 Arial 字体为止，找到后，显示如下的提示信息。



### 7.6.3 剪贴片 Clipboard 对象

Clipboard 对象用于操作剪贴板加上的文本和图形，它使用户能够复制、剪切和粘贴应用程序中的文本和图形。该对象的常用方法如表所示。

方法	说明
Clear	用于清除系统剪贴板的内容
SetData 图形数据 [, 格式] GetData (格式)	按指定的图形格式将数据放到 Clipboard 对象中粘贴 Clipboard 对象中指定的数据
SetText 文字数据 GetText	将选中的文字保存到 Clipboard 对象中粘贴 Clipboard 对象中的文本字符串
GetFormat	返回一个整数，指出 Clipboard 对象中的项目是否匹配期望的格式。常用的格式有：vbText 文字 vbCFbitm.bmp 格式 vbCFMetafile.wmg 格式

例如，利用 Clipboard 对象将图片框 Picture1 中的图片粘贴到文本框 Picture2 中，将 Clipboard 对象中的文本粘贴到文本框 Text1 中。设计如图所示的窗体。

在 Picture2 控件的单击实践中输入代码如下。

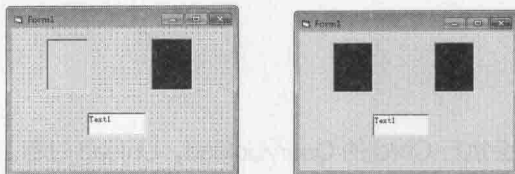
```
01 Private Sub Picture2-click ( )
02 Clipboard.Clear '清除 Clipboard 对象'
03 Clipboard.SetData Picture1.picture '用指定图像格式将图片放到 Clipboard 对象上'
04 Picture2=loadpicture ( " " ) '返回图片的位置'
05 Picture2=Clipboard.GetData() '从 Clipboard 对象返回一个图形'
06 End sub
```

在 Text1 控件的单击事件中输入如下代码。

```
01 Private Sub Text1-Click ( )
02 Text1=Clipboard.GetText() '返回 Clipboard 对象中的文本字符串'
03 End Sub
```

运行该示例，当用户单击 picture2 控件时，其将 Picture1 控件中的对象复制到 Picture2 控件中。

示例运行结果如下图所示。



## 7.6.4 调试 Debug 对象

Debug 是一个隐藏的对象，用于进行和调试相关的工作。在调试程序的时候使用 Debug 会很方便。比如 Debug.Print a 会在 VB 界面下方的“立即”窗口中显示出 a 的值，这样你就不必修改程序的界面来显示 a 或者中断程序来查看 a。再比如当你在某处需要  $a > 5$  的时候，你可以加一句 Debug.Assert( $a > 5$ )，当  $a \leq 5$  的时候程序执行到此处就会暂停，然后你就可以查看 a 为何没有满足  $a > 5$  的条件，从而找到代码错误。

Debug 的一个重要优点是，当你把程序编译成成品 EXE 之后，这些 Debug 语句都被剔除，也就是说你在程序中加入 Debug 语句并不会影响最终成品的效率和界面。Debug 只是为调试人员存在的。

## 7.7 高手点拨



本节视频教学录像：3 分钟

### 1. Form\_Activate

当一个对象成为活动窗口时发生，使用此过程请注意一下，如果在一个多窗体的程序中，例如，Form1 中从慢点 Command1 按钮 Form2.Show 的话，并且 Form2 的 Borderstyle 属性为 0, 3, 4, 5 中之一的值的话，千万不要再在这个过程里面加入 MsgBox 函数，要不然就会反复地弹出对话框。如果再在 form1 中的这个过程中加入 MsgBox 的话那就更热闹了，因为在 MsgBox 点击确定后会执行 Form\_Activate 过程。

### 2. Form\_QueryUnload(Cancel As Integer, UnloadMode As Integer)

发生于 Form\_Unload 之前，多一个可以判别结束方式的变量。

UnloadMode 值的含义如下表所示。

常数	值	描述
vbFormControlMenu	0	用户从窗体上的“控件”菜单中选择“关闭”指令
vbFormCode	1	Unload 语句被代码调用
vbAppWindows	2	当前 Microsoft Windows 操作环境会话结束
vbAppTaskManager	3	Microsoft Windows 任务管理器正在关闭应用程序
vbFormMDIForm	4	MDI 子窗体正在关闭，因为 MDI 窗体正在关闭
vbFormOwner	5	因为窗体的所有者正在关闭，所以窗体也在关闭

如果用于 MDI 窗口中并且直接关闭 parent 窗口时, 过程顺序如下。

- (1) parent.QueryUnload
- (2) child.QueryUnload
- (3) child.Unload
- (4) parent.Unload

只要子窗体被 Load, 都会执行 Child 的 QueryUnload, Unload 过程。

### 3. 另外一种窗体打开方式

一般的打开方式就是 Form1.Show 或者是 Form1.Visible = True。但是如果在打开某个窗体前使用窗体的属性时, 就会产生 Load 事件, 如果在 Load 里面在自己 Show 一下就显示出来了。不同打开方式的结果列表如下。

```
Private Sub Command1_Click()
```

```
Form2.Show ' 依次执行 Form_Initialize, Form_Load, Form_Activate
```

```
End Sub
```

```
Private Sub Command2_Click()
```

```
Form2.Visible = True ' 依次执行 Form_Initialize, Form_Load, Form_Activate
```

```
End Sub
```

```
Private Sub Command3_Click()
```

```
Form2.Visible = False ' 依次执行 Form_Initialize, Form_Load, 不显示窗口
```

```
End Sub
```

```
Private Sub Command4_Click()
```

```
Form2.Left = 0 ' 依次执行 Form_Initialize, Form_Load, 不显示窗口
```

```
End Sub
```

第一次打开窗口后, 如果 Unload 时不写代码 Set Form1 = Nothing, 那么在此打开时会不执行 Form\_Initialize 过程。

如果窗体已经显示, 那么 Form2.Visible = True 是不能激活 Form\_Activate 过程的, 而 Form2.Show 却可以。

还有如果是 MDI 文档的子窗体, 在用 Show 或者 Visible 方法时都是不会执行 Form\_Activate 过程的。

## 7.8 实战练习

### 一、思考题

1. 列举窗体的 3 个主要属性及其作用。
2. 单窗体应用程序和多窗体应用程序有什么不同?

### 二、操作题

在 Visual Basic 6.0 中设计一个单窗体应用程序, 完成以下功能。

- (1) 窗体名称为 “Form1”, 标题为 “显示与删除”, 最小化按钮可用, 最大化按钮不可用。
- (2) 在窗体中添加一个名称为 Cmd1, 标题为 “显示” 的命令按钮。
- (3) 如果单击命令按钮, 则执行语句 Form1.Print “显示”; 如果单击窗体, 则执行语句 Form1.Cls。

# 第8章



本章视频教学录像：25 分钟

## 标准模块和类模块

Visual Basic 中的代码都存储在模块中。简单的应用程序可以只有一个窗体，所用的程序都驻留在窗体模块中，而当应用程序庞大复杂时，可创建一个独立的模块，用它实现代码公用。该独立模块即是标准模块。此外还可以建立包含共享代码与数据的类模块。本章将详细讲解怎样添加标准模块和类模块。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解标准模块的概念
- ☐ 掌握标准模块的添加方法
- ☐ 了解类模块的概念
- ☐ 掌握类模块的添加方法
- ☐ 了解标准模块和类模块的区别
- ☐ 熟悉标准模块和类模块的应用实例

## 8.1 标准模块



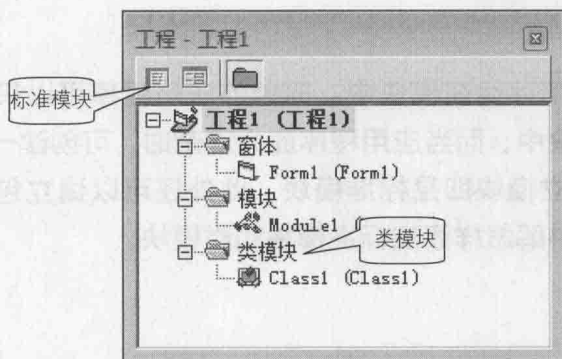
本节视频教学录像: 4 分钟

标准模块用于放置工程中公用的变量、常量、数据类型、函数过程和子过程等。下面介绍什么是标准模块及在工程中如何添加标准模块。

### 8.1.1 标准模块概述

标准模块 (文件扩展名为 .BAS) 是应用程序内其他模块访问的过程和声明的容器。它们可以包含变量、常数、类型、外部过程和全局过程的全局 (在整个应用程序范围内有效) 声明或模块级声明。标准模块中的代码不仅可以应用于一个工程, 还可以应用到其他工程中。

在工程资源管理器中, 一般情况下会存在标准模块 (Module)、类模块 (Class) 等资源, 如图所示。

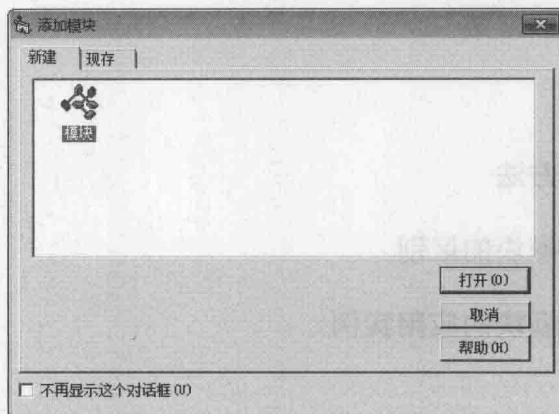


### 8.1.2 添加标准模块

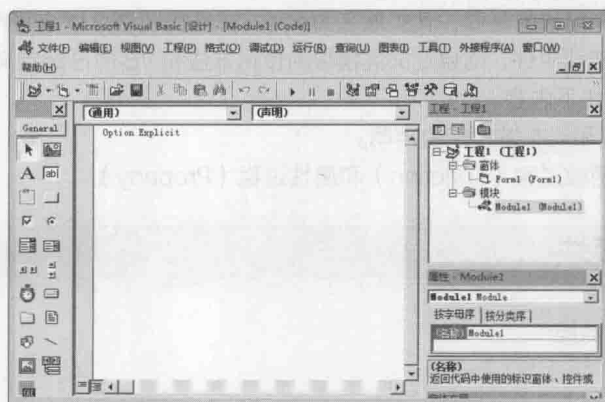
本节介绍添加标准模块的操作。

#### 1. 添加新标准模块

选择“工程”/“添加模块”命令, 弹出“添加模块”对话框。



选择“新建”选项卡，选择“模块”图标，单击“打开”按钮，即可添加一个标准模块到工程中，如图所示。

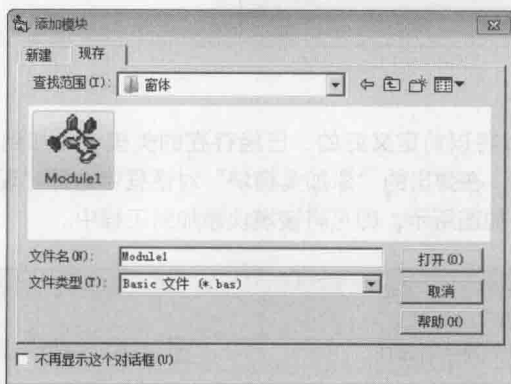


在工程资源管理器中单击鼠标右键，在弹出的快捷菜单中选择“添加/“模块”命令，同样可以弹出如图所示的对话框。

**提示**

## 2. 添加现存标准模块

对于一些比较通用的模块，如数据链接等，只需对其中的少部分内容进行修改，就可以将其应用到其他的程序中，这样减少了程序代码的编写量，加快了程序的开发速度。具体方法为：选择“工程”/“添加模块”命令，在弹出的“添加模块”对话框中选择“现存”选项卡，选中需要添加的模块，单击“打开”按钮，如图所示，即可将现存的标准模块添加到工程中。



## 8.2 类模块



本节视频教学录像：3分钟

类模块和标准模块在功能上比较类似。下面介绍什么是类模块以及在工程中如何添加类模块。



## 8.2.1 类模块概述

类模块是 VB 面向对象编程的基础,其扩展名为 .cls。在类模块中可以编写代码创建新对象,该对象可以包括自己的属性、方法和事件,而自定义类模块的使用方法和 VB 中已经定义好的类是完全相同的。

在类模块中一般包括如下内容。

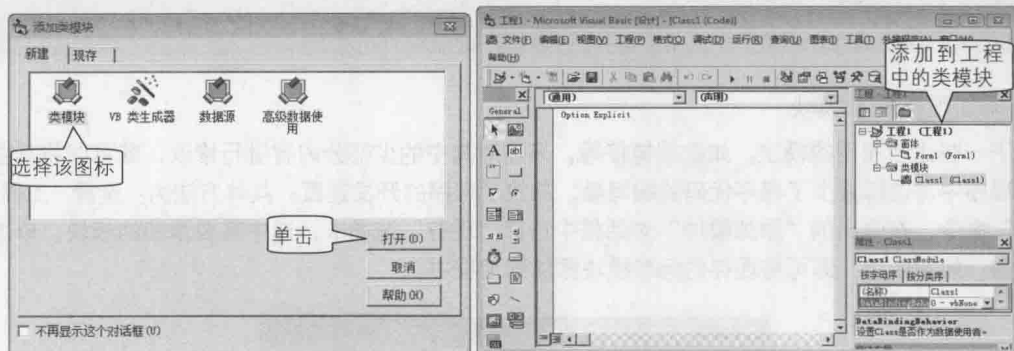
- (1) 常数、类型、变量和动态链接库的声明。
- (2) 子过程 (Sub)、函数过程 (Function) 和属性过程 (Property)。

## 8.2.2 添加类模块

本节介绍添加类模块的操作。

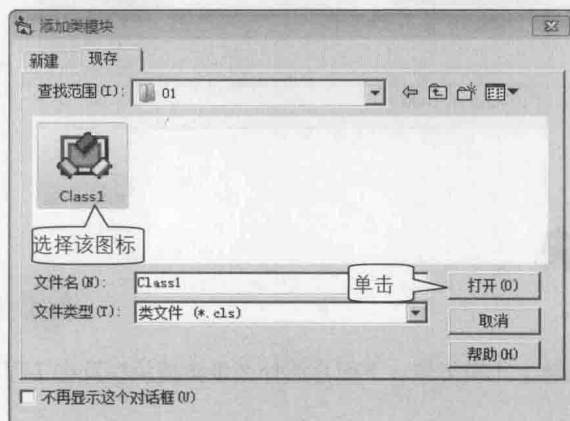
### 1. 添加新类模块

添加类模块和添加标准模块的方法类似,选择“工程”/“添加类模块”命令,弹出“添加类模块”对话框。选择“新建”选项卡,选择“类模块”图标,单击“打开”按钮即可将新的类模块添加到工程中。



### 2. 添加现存的类模块

和标准模块相同,也可以将以前定义好的、已经存在的类模块添加到工程中。具体方法为:选择“工程”/“添加类模块”命令,在弹出的“添加类模块”对话框中选择“现存”选项卡,选择要添加的类模块,单击“打开”按钮,如图所示,即可将该模块添加到工程中。



## 8.3 标准模块和类模块的区别



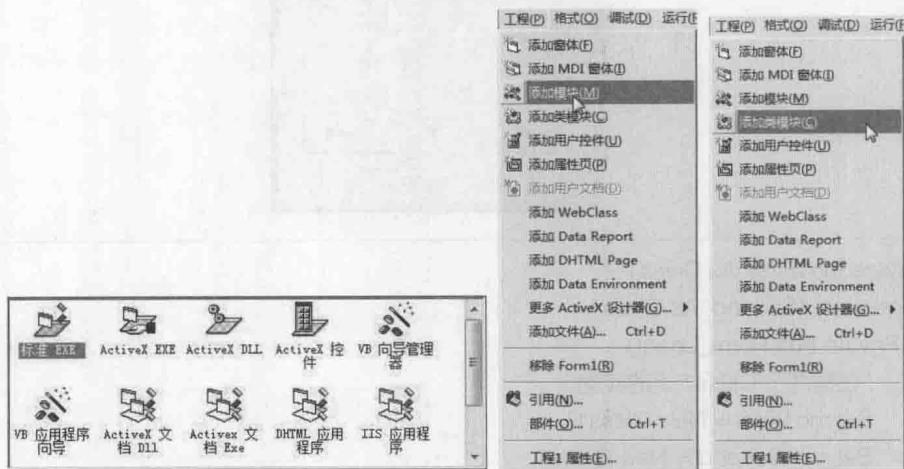
本节视频教学录像: 14 分钟

类模块和标准模块的不同点在于存储数据方法的不同。标准模块的数据只有一个备份。这意味着标准模块中一个公共变量的值改变以后, 在后面的程序中再读取该变量时, 它将得到同一个值。

而类模块的数据, 是相对于类实例 (也就是, 由类创建的每一对象) 而独立存在的。同样地, 标准模块中的数据在程序作用域内存在, 也就是说, 它存在于程序的存活期中; 而类实例中的数据只存在于对象的存活期, 它随对象的创建而创建, 随对象的撤消而消失。

最后, 当变量在标准模块中声明为 Public 时, 则它在工程中任何地方都是可见的; 而类模块中的 Public 变量, 只有当对象变量含有对某一类实例的引用时才能访问。

上面的比较, 同样适用于标准模块和类模块中的公共过程。用下面的例子可以说明。打开一个新 Standard Exe 工程, 并在“工程”菜单中个添加一个标准模块和一个类模块, 然后运行以下的代码。



把下面的代码放在 Class1 中。

```
01 '下面是 Class1 对象的一个属性。
02 Public Comment As String
03 '下面是 Class1 对象的一个方法。
04 Public Sub ShowComment()
05     MsgBox Comment, vbstrVisibleEverywhere
06 End Sub
```

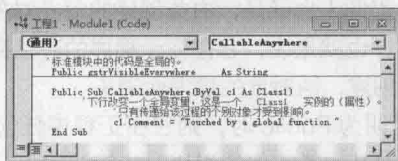


把下面的代码放在 Module1 中。

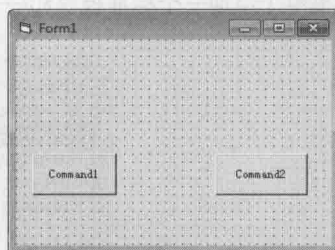
```

01 Public gstrVisibleEverywhere As String
02 Public Sub CallableAnywhere(ByVal c1 As Class1)
03     c1.Comment = "Touched by a global function."
04 End Sub

```



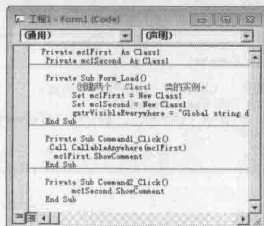
把两个命令按钮放在 Form1 上，并在 Form1 中添加以下的代码。



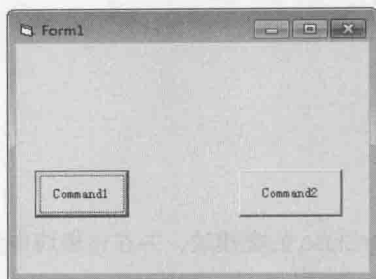
```

01 Private mc1First As Class1
02 Private mc1Second As Class1
03 Private Sub Form_Load()
04     ' 创建两个 Class1 类的实例。
05     Set mc1First = New Class1
06     Set mc1Second = New Class1
07     gstrVisibleEverywhere = "Global string data"
08 End Sub
09 Private Sub Command1_Click()
10     Call CallableAnywhere(mc1First)
11     mc1First.ShowComment
12 End Sub
13 Private Sub Command2_Click()
14     mc1Second.ShowComment
15 End Sub

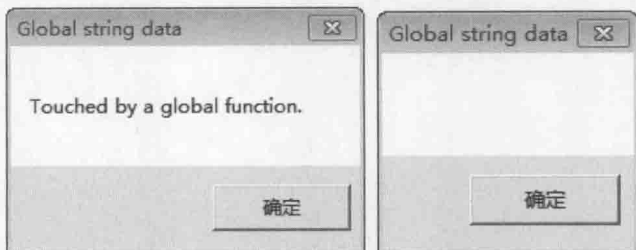
```



按【F5】键，运行该工程。当 Form1 加载时，它创建两个 Class1 类实例，每个实例有自己的数据。同时，Form1 设置了下面全局变量 gstrVisibleEverywhere 的值。



按下 Command1，调用全局过程并传递引用给第一个 Class1 对象。全局过程设置 Comment 属性，然后 Command1 调用 ShowComment 方法显示该对象的数据。



如上左图所示第一个 Class1 对象的信息框，结果信息框演示了全局过程 CallableAnywhere 如何设置对象的 Comment 属性，而且全局字符串在 Class1 内部是可见的。

按下 Command2，调用第二个 Class1 类实例的 ShowComment 方法。右图所示为第二个 Class1 对象的信息框，两个对象都访问了全局字符串变量；然而，第二个对象的 Comment 属性是空的，因为对全局过程 CallableAnywhere 的调用只改变第一个对象的 Comment 属性。

另外，要避免类的代码依赖于全局变量，也就是标准模块中的公共变量。一个类的许多实例可以同时存在，所有这些对象在程序中共享全局数据。在类模块代码中使用全局变量也违背了面向对象封装的编程原则，因为由这样的类所创建的对象并没有包含它们的所有数据。

## 8.4 高手点拨



本节视频教学录像：4 分钟

在编写程序时，很可能会遇到一些使用相同变量和例程的窗体和事件过程。在缺省状态下，变量对于事件过程来说是局部的，就是说仅能在创建这些变量的事件过程中读取或者修改变量。与之相似，事件过程对于创建它们的窗体来说也是局部。为了在工程中的所有窗体和事件中共享变量和过程，需要在该工程的一个或多个标准模块中对它们进行声明和定义。标准模块或代码模块是具有文件扩展名 .bas，并包含能够在程序任何地方使用的变量和过程的特殊文件。正如窗体一样，标准模块被单独列在 Project(工程)窗口内，并可通过使用 File(文件)菜单中的 Save Module1 As 菜单项存盘。但是，与窗体不同，标准模块不包含对象或属性设置而只包含可在代码窗口中显示和编辑的代码。

在 VB 中类模块是面向对象编程的基础。可以在类模块中编写代码建立新对象。这些新对象可以包

含自定义的属性和方法。实际上,窗体正是这样一种类模块,在其上可安放控件,可显示窗体窗口用类模块创建对象,这些对象可被应用程序内的过程调用。标准模块只包含代码,而类模块包含代码又包含数据,可视为没有物理表示的控件。

## 8.5 实战练习

尝试在工程中添加一个名为 MyClass 的类模块,并在该模块中创建一个名为 Myval 的属性,属性值为字符串型。

步骤如下:

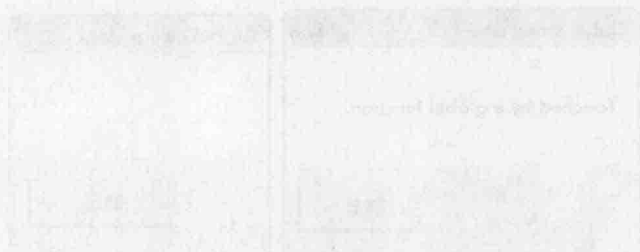


图 8-5-1 所示为在工程中添加名为 MyClass 的类模块,并在该模块中创建一个名为 Myval 的属性,属性值为字符串型。

## 8.4 高手点拨

本章主要介绍类模块的用法。

本章主要介绍类模块的用法。类模块是 Visual Basic 中一种特殊的模块,它用于定义对象的结构和行为。类模块可以包含属性、方法和事件。本章将详细介绍如何创建和使用类模块。

# 第9章



本章视频教学录像：2 小时 14 分钟

## VB 的简易之道——标准控件

控件的应用在 Visual Basic 程序设计中有着举足轻重的地位。通过对各种控件的属性、方法及事件的处理，可以实现各种需要的功能。本章是 Visual Basic 图形化编程的入门篇章，主要介绍了 Visual Basic 6.0 的 13 种通用标准控件。对于每一种控件，都分别从其属性、事件和方法等各方面进行了介绍，并通过多个具体实例使用户掌握这些控件在具体程序中的使用。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握标签、文本框、命令按钮控件
- ☐ 了解单选按钮、复选框、框架、列表框控件
- ☐ 了解组合框、图像框控件
- ☐ 熟悉滚动条控件
- ☐ 熟悉定时器控件
- ☐ 熟悉文件系统控件
- ☐ 熟悉控件数组

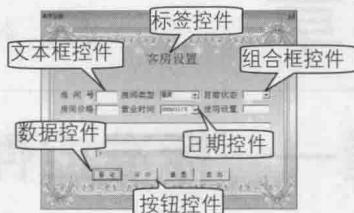


## 9.1 控件概述



本节视频教学录像: 5 分钟

控件是指图形用户界面屏幕上的一种对象, 用户可操作该对象来执行某一行为, 它是 Visual Basic 6.0 中最重要的组成部分。如果把一个程序系统比作一个高楼大厦的话, 那么控件就相当于这个高楼大厦的一小块砖、一层层楼板。例如在下图所示的一个窗体中, 就用到了标签控件、按钮控件、文本框控件、组合框控件、日期控件以及数据控件等。



**控件属性:** 是指控件的名称、字体、大小等自身的性质。就如同砖有砖的颜色, 楼板有楼板的大小一样, 通过对属性的设置可以实现一些需要的功能。

**控件方法:** 是指该控件所具有规定好的, 用于实现某些特殊功能的过程。例如窗体的 Show 方法就是用来显示窗体的, Hide 方法就是用来隐藏窗体的。

**控件事件:** 控件事件是对系统消息的响应, 例如鼠标的 Click (单击) 事件, 窗体的 Load (加载) 事件等。

## 9.2 标签控件



本节视频教学录像: 13 分钟

标签控件 (Label) 主要用于显示字符和各种文字信息。后面将要介绍的文本框控件 (TextBox) 也是用于显示字符和各种文字信息的控件。标签控件和文本框控件的区别在于标签控件只是用于向用户显示文字信息, 用户不能更改所显示的文字; 而文本框控件除了能够向用户显示文字信息外, 用户还可以修改文本框中的内容。

新建一个标签控件后, 系统会为这个标签控件设置一个形如 Labelx (其中 x 为 1、2、3……) 的默认名称, 但是在实际编程时, 可以将其名字改为更具有实际意义的, 如 LabelColor、LabelFont 之类的名称。



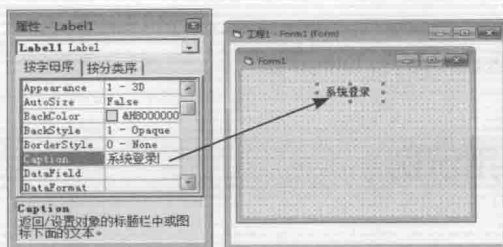
**提示**

这里采用的命名方式是每个单词的第 1 个字母大写, 其中第 1 个单词表示该控件的类别, 第 1 个单词就是 Label; 第 2 个单词表示该控件所实现的功能, 如果标签表示的是颜色信息, 使用 Color 来表示。如果单词非常长, 也可以使用它的缩写, 以免过长的单词引发代码输入错误。好处就在于代码的可读性强。

### 9.2.1 标签控件的主要属性

#### 1. 标题属性 (Caption)

标题属性是最常用的属性之一，此属性用来设置标签控件所显示的文本信息。用户可以在标签控件属性窗口中 Caption 属性对应的输入框中进行设置，例如设置标签的 Caption 属性为“系统登录”。



## 2. 边框属性 (BorderStyle)

通过边框属性可以改变标签控件的边框类型。可以在标签控件属性窗口中 BorderStyle 属性对应的下拉列表中进行设置。



对于初学者来说不需要记住每一个控件的属性，当不知道某个属性的意义时，尝试改变它的值，看看发生了什么。

变化一下就会明白了，或者查阅 MSDN，上面有很详细的说明。另外，一般来说，绝大多数控件的属性都可以在属性列表中设置，这时建议大家通过使用属性列表来设置，尽量不要通过源代码设置。因为在属性列表中很多属性的合理值都被列出以供选择，因此能够避免因为手动输入代码而出现输入错误。

## 9.2.2 标签控件 (Label) 的主要事件

Visual Basic 是具有条件触发机制的编程工具，标签控件理所当然拥有它自己的事件。标签控件的事件不是很多，主要有以下两种。

### 1. 鼠标单击事件 (Click Event)

鼠标按下再弹起的一次过程就是一次单击。在控件上单击鼠标后将运行的程序，就是鼠标单击事件所触发的程序。

语法格式如下。

```
Private Sub Label_Click([index As Integer])
```

### 2. 鼠标双击事件 (DbClick Event)


快速按下鼠标再弹起两次的过程就是一次双击。在控件上双击鼠标后将运行的程序,就是鼠标双击事件所触发的程序。

语法格式如下。

```
Private Sub Label_DblClick ([index As Integer])
```

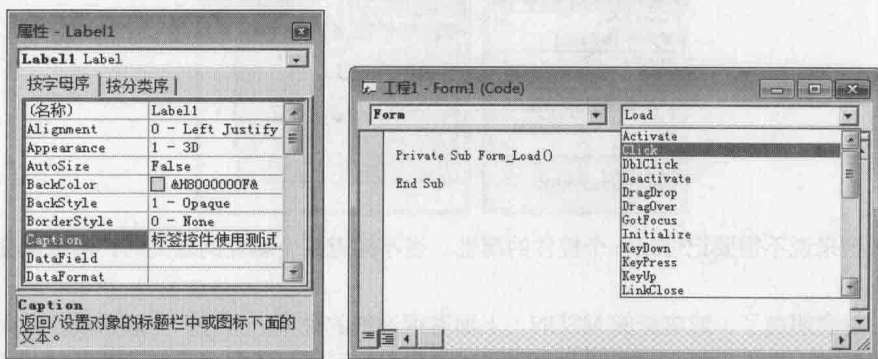
### 9.2.3 标签控件应用示例

**【范例 9-1】** 标签控件的变化是由标签的属性决定的,通过对标签控件的字体、字号、颜色及边框等属性的控制,来改变标签控件的外观。

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标,然后单击【打开】按钮。

(2) 在窗体中添加 Label 控件,并将 Label1 控件的 Caption 属性设置为“标签控件使用测试”。

(3) 双击窗体,进入代码窗口,从中选择【Form】窗体的【Click】事件。



(4) 在【Click】事件中输入以下代码(代码 9-1.txt)。

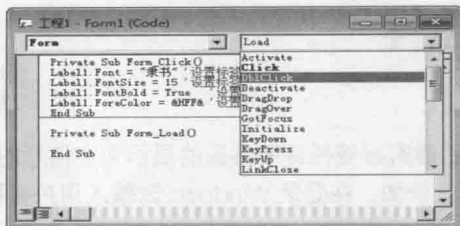
```
01 Private Sub Form_Click()
02     Label1.Font = "隶书"      ' 设置标签中文字的字体为隶书
03     Label1.FontSize = 15      ' 设置标签中文字的字体大小为 15 号字
04     Label1.FontBold = True    ' 设置标签中文字的字体为粗体
05     Label1.ForeColor = &HFF& ' 设置标签中文字的字体为红色
06 End Sub
```



**注意**

该事件下的代码主要是对标签控件的字体进行设置。

(5) 在代码窗口中选择【Form】窗体的【DblClick】事件。



(6) 在【DbClick】事件中输入以下代码。


```
01 Private Sub Form_DbClick()  
02   Label1.BorderStyle = 1 '为 Label1 添加边框  
03 End Sub
```



**提示**

该事件下的代码主要是对标签控件再添加一个边框。

## 【运行结果】

保存程序，单击【启动】按钮 ，运行程序。分别用鼠标单击（下图左）窗体和双击（下图右）窗体，查看标签变化。



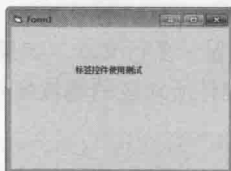
## 【拓展训练 9-1】

单击鼠标后，更改标签字体为华文行楷，并设置字体对齐方式为居中。

看到这个题目后，经分析可知，主要涉及 Label 的 Font 属性和 Alignment 属性，因此对【范例 9-1】中步骤(4)的代码进行相关的修改即可。

```
01 Private Sub Form_Click()  
02   Label1.Font = " 华文行楷" '设置标签中文字的字体为华文行楷  
03   Label1.Alignment = 2 '设置标签中文字的对齐方式为居中形式  
04 End Sub
```

运行后，单击窗体就会看到所需的结果。



## 9.3 文本框控件



本节视频教学录像：13 分钟

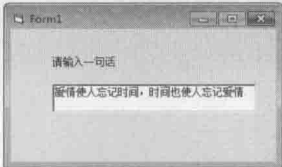
文本框控件（TextBox）除了具有标签控件所具备的显示文字信息的功能外，还可以在文本框中输入文本，让用户和软件进行交互。例如，在登录 Windows 时输入用户名和密码的地方就是文本框。

### 9.3.1 文本框的主要属性

本小节介绍几个最常用的文本框控件属性。

#### 1. 文本属性（Text）

文本框控件的文本属性和标签控件的标题属性类似，用于控制文本框中所显示的文字。用户可以利用文本框控件对应属性窗口中的 Text 属性输入框进行设置。

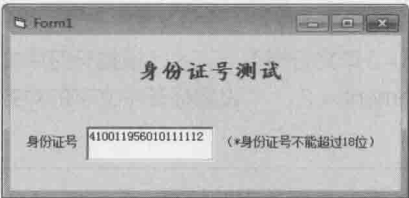


提示

如果没有设置文本框的多行属性，所输入的文本内容则不能换行，若要强制换行，可以按【Ctrl+Enter】组合键换行。

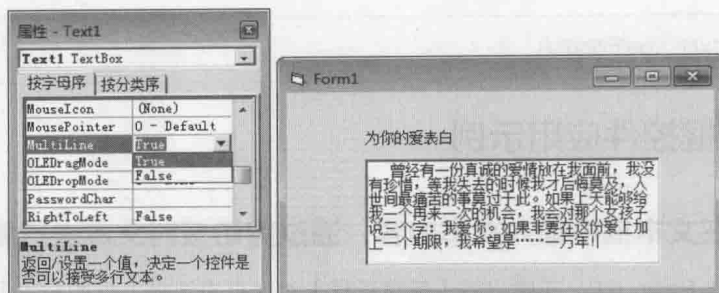
#### 2. 最大长度属性（MaxLength）

顾名思义，最大长度属性是指文本框所能输入的最大字符数。它的默认值为 0，代表不限制文本框所能输入的最大长度。但是实际上还是有最大限度的，如果不限最大长度，文本框能够输入最多 64KB 字符。可以在文本框控件所对应的属性窗口中的 MaxLength 属性输入框中设置最大长度属性。



#### 3. 多行属性（MultiLine）

多行属性就是控制该文本框是否可以显示多行文本。如果该值为 True，则允许多行显示；如果为 False，则只能单行显示。可以在文本框控件所对应的属性窗口中的【MultiLine】下拉列表框中设置多行属性。



#### 4. 密码属性 ( PasswordChar )

密码属性用来设置文本框中输入的值如何显示，多用于对密码格式的设置。可以设置任意字符作为掩码，这样无论输入什么字符，在文本框中始终显示的是我们设置的字符。例如想让密码以符号“\*”显示，只需要将 PasswordChar 属性值设置为“\*”即可。



### 9.3.2 文本框控件常用的事件

文本框控件的常用事件有 Change、GotFocus、LostFocus 等。

#### 1. Change 事件

当用户向文本框中输入内容，或者将程序中文本框控件中的“Text”属性设置为新值时，将触发 Change 事件。语法如下。

```
Private Sub Text1_Change()
```

#### 2. GotFocus 事件

GotFocus 事件又叫作获得焦点事件。所谓获得焦点是指控件处于活动状态。即用【Alt+Tab】键在各个程序中切换，处于活动中的程序就会获得焦点。语法格式如下。

```
Private Sub Text1_GotFocus()
```

#### 3. LostFocus 事件


LostFocus 事件又名失去焦点事件。所谓失去焦点是指处于不活动状态。即用【Alt+Tab】键在各个程序中切换，不处于活动的程序则会失去焦点 (LostFocus)。语法格式如下。



```
Private Sub Text1_LostFocus()
```

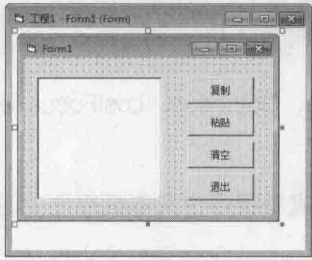
9.3.3 文本框控件应用示例

【范例 9-2】在文本框中输入文本内容，通过剪切板对文本进行复制和粘贴。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在窗体中添加一个 TextBox 控件，并将 TextBox1 控件的 Text 属性值清空，将多行属性（Multiline）设置为“True”。
- (3) 在窗体中添加 4 个命令按钮（CommandButton）控件，控件的属性设置如表所示。

名称	Caption	控件作用
CmdCopy	复制	将选中的文本内容复制到剪贴板
CmdPaste	粘贴	将剪贴板中的内容粘贴到文本框中
CmdEmpty	清空	将文本框中的内容清空
CmdExit	退出	退出程序

- (4) 将窗体中的控件摆放到合适的位置，效果如图所示。



- (5) 双击【复制】按钮，在代码窗口中输入以下代码。

```
01 Private Sub CmdCopy_Click()  
02     Clipboard.SetText Text1.Text      ' 复制选中的内容  
03 End Sub
```

- (6) 双击【粘贴】按钮，在代码窗口中输入以下代码。

```
01 Private Sub CmdPaste_Click()  
02     Text1.Text = Clipboard.GetText      ' 将选中的内容粘贴到文本框中  
03 End Sub
```


- (7) 双击【清空】按钮，在代码窗口中输入以下代码。

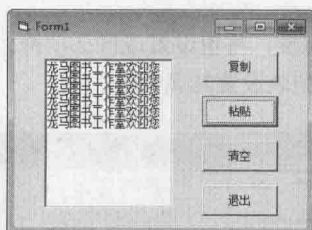
```
01 Private Sub CmdEmpty_Click()
02     Text1.Text = "" '清空文本框中的内容
03 End Sub
```

(8) 双击【退出】按钮，在代码窗口中输入以下代码。

```
01 Private Sub CmdExit_Click()
02     End '退出程序
03 End Sub
```


## 【运行结果】

保存程序，单击【启动】按钮 , 运行程序。在文本框中输入一段文字，选中文本后可以复制，将光标定位于文本框后可以粘贴，同时，还可以进行将文本框内容清空、退出程序等操作。




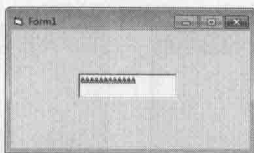
## 【拓展训练 9-2】

在文本框中输入一段字符，以密码形式显示，即以“\*”号来代替所输入的内容。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在窗体中添加一个 TextBox 控件，将 TextBox1 控件的 Text 属性值清空，并将 TextBox1 控件的 PasswordChar 属性值设置为“&”。

(3) 保存程序，单击【启动】按钮 , 运行程序。在文本框中输入内容，可以看到输入的文本内容是以密码形式显示的。



## 9.4 命令按钮控件



本节视频教学录像：12 分钟

命令按钮在 Visual Basic 中也是一种常用的控件。在前面的学习中，我们对这个控件已经有了一个初步的认识，它的主要作用就是当我们单击某一个命令按钮时，将会执行相关的命令。

### 9.4.1 命令按钮控件的主要属性

#### 1. 取消属性（Cancel）和默认属性（Default）

当用户按下键盘上的【Enter】键和【Esc】键时，若想触发窗体上相应的按钮单击事件，就需要用到按钮控件的 Cancel 属性和 Default 属性。当按钮控件的 Default 属性被设置为 True，运行程序时，按下【Enter】键就相当于用鼠标单击该按钮；当按钮控件的 Cancel 属性被设置为 True，运行程序时，按下【Esc】键就相当于用鼠标单击该按钮。

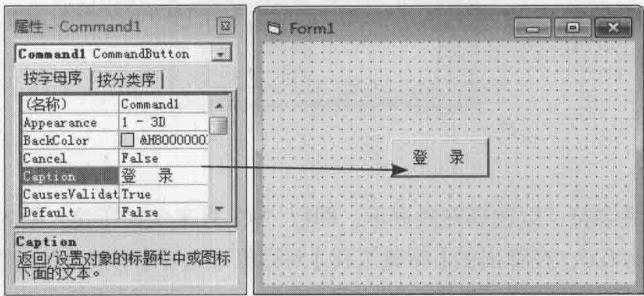


注意

在一个窗体中，最多只能设置一个默认按钮，原因依然非常容易理解，如果有两个或两个以上的按钮是默认按钮，当用户没有选择任何按钮直接按回车键的时候，究竟应该打开哪一个按钮？用户可以通过属性窗口中的【Default】下拉列表框设置其属性值。

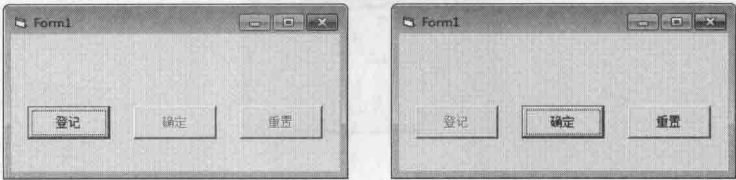
#### 2. 标题属性（Caption）

标题属性是用来显示控件标题的属性，将值设置成什么字符，控件的标题就显示什么字符。用户可以通过属性窗口中的【Caption】下拉列表设置其属性值。



#### 3. 可用属性（Enabled）

按钮的 Enabled 属性用来设置控件是否可用。在程序设计过程中，当条件不符合要求时，往往会将该控件设置为失效状态（即不可用）。例如，当向数据库中写入数据时，在初始状态下，窗体中的任何控件都不允许使用，当单击【登记】按钮后，窗体中的其他控件允许操作，同时【登记】按钮失效。




### 9.4.2 命令按钮控件的事件

命令按钮之所以叫作命令按钮，是因为它的主要功能就是按下该按钮的时候执行一些命令，所以命令按钮中最常用的事件就是鼠标单击（Click）事件。

### 9.4.3 命令按钮控件应用示例

当单击按钮A后按钮B可用，同时按钮A不可用；当单击按钮B后按钮A可用，同时按钮B不可用。

**【范例 9-3】**在窗体上添加两个命令按钮，当单击第 1 个按钮时，第 2 个按钮可用，第 1 个按钮变为不可用；当单击第 2 个按钮时，第 1 个按钮可用，第 2 个按钮变为不可用。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中添加两个命令按钮控件，将这些控件的属性按照下表所示进行设置。

名称	Caption	控件功能
Cmd01	&A 按钮 1	设置按钮 2 可用，并为按钮 1 设置快捷键
Cmd02	&B 按钮 2	设置按钮 1 可用，并为按钮 2 设置快捷键


(3) 双击【A 按钮 1】，进入代码窗口，输入以下代码。

```
01 Private Sub Cmd01_Click()
02   Cmd02.Enabled = True   ' 设置按钮 2 可用
03   Cmd01.Enabled = False  ' 设置按钮 1 不可用
04 End Sub
```

(4) 双击【B 按钮 2】，进入代码窗口，输入以下代码。

```
01 Private Sub Cmd02_Click()
02   Cmd02.Enabled = False  ' 设置按钮 2 不可用
03   Cmd01.Enabled = True   ' 设置按钮 1 可用
04 End Sub
```


#### 【运行结果】

保存程序，单击【启动】按钮 ，运行程序，然后单击两个按钮查看效果。



#### 【拓展训练 9-3】

在窗体上添加一个命令按钮，通过对命令按钮的属性用代码进行设置，使其在单击时外观能够发生变化。

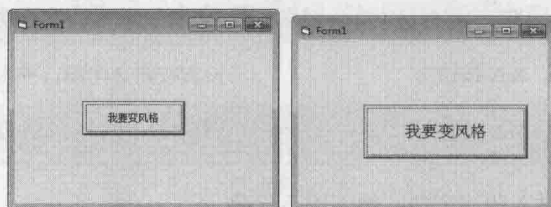
(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标  , 然后单击【打开】按钮。

(2) 在窗体中添加一个命令按钮控件, 将其 Caption 属性值设置为“我要变风格”。

(3) 双击【我要变风格】按钮, 在代码窗口中输入以下代码。

```
01 Private Sub Command1_Click()  
02   Command1.Width = 3000      ' 调整按钮控件宽度  
03   Command1.Height = 1000     ' 调整按钮控件高度  
04   Command1.FontSize = 15     ' 调整按钮控件字体大小  
05 End Sub
```

(4) 运行程序查看效果, 单击按钮前效果如左下图所示, 单击按钮后效果如右下图所示。



## 9.5 单选按钮控件



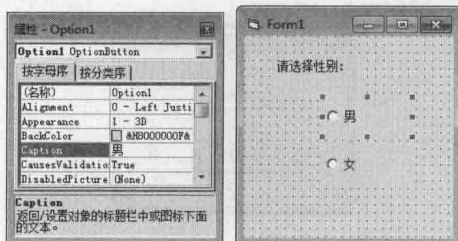
本节视频教学录像: 6 分钟

在给出一组选项时, 如果只允许用户进行单一选择, 就需要用到单选按钮控件。单选按钮控件是由一组互斥的单选按钮组成的, 每组中在某个时间内只能有一个单选按钮被选中, 被选中的单选按钮内会显示一个黑点, 而其他的单选按钮则没有黑点。

### 9.5.1 单选按钮的主要属性

#### 1. 标题属性 (Caption)

Caption 属性用于设置单选按钮的显示标题, 以此来说明该控件所表示的功能。

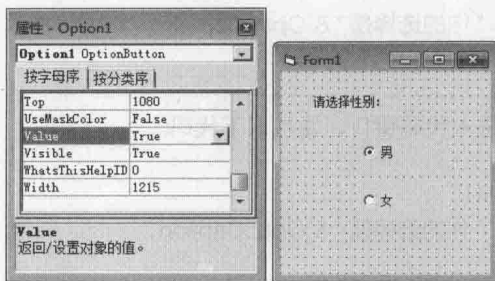


#### 2. 值属性 (Value)

Value 属性用于设置该单选按钮是否被选中。属性值为 True 时, 表示已经选中; 属性值为 False 时,



表示没有选中。




## 9.5.2 单选按钮的常用事件

单选按钮的常用事件为鼠标单击事件 (Click)。用户在单选按钮上单击鼠标时激活该事件。

## 9.5.3 单选按钮控件应用示例

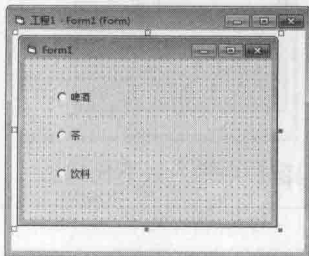
利用单选按钮，根据用户的选择，显示出用户的需求。

**【范例 9-4】**在窗体中添加 3 个单选按钮，当选择其中的 1 个单选按钮时，在窗体上显示出所选择的单选按钮内容。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中添加 3 个单选按钮和 1 个 Label 控件，控件的属性按下表所示修改，并将各控件按照下图所示排列。

名称	Caption	控件作用
Opt1	啤酒	用户选择选项 1
Opt2	茶	用户选择选项 2
Opt3	饮料	用户选择选项 3
LblDegree	“ ”	用于显示用户选择的信息



(3) 双击单选按钮【啤酒】，进入代码窗口，输入以下代码。



```

01 Private Sub Opt1_Click()
02     LblDegree.Caption = " 你的选择是 " & Opt1.Caption           ' 显示选择为 " 啤酒 "
03 End Sub

```

(4) 双击单选按钮【茶】，进入代码窗口，输入以下代码。

```

01 Private Sub Opt2_Click()
02     LblDegree.Caption = " 你的选择是 " & Opt2.Caption           ' 显示选择为 " 茶 "
03 End Sub

```


(5) 双击单选按钮【饮料】，进入代码窗口，输入以下代码。

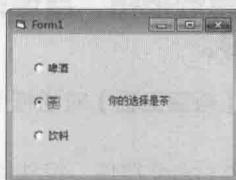
```

01 Private Sub Opt3_Click()
02     LblDegree.Caption = " 你的选择是 " & Opt3.Caption           ' 显示选择为 " 饮料 "
03 End Sub

```


## 【运行结果】

保存程序，单击【启动】按钮，运行程序验证结果。

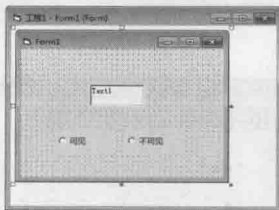


## 【拓展训练 9-4】

使用单选按钮可以很方便地控制某一控件的诸多属性，下面就来演练一下，如何用单选按钮控制文本框是否可见。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标，然后单击【打开】按钮。

(2) 在窗体中添加两个单选按钮控件和一个文本框控件，并修改单选按钮 Option1 的 Caption 属性为“可见”，修改单选按钮 Option2 的 Caption 属性为“不可见”，并将各控件按照下图所示排列。



(3) 双击【可见】单选按钮，在代码窗口中输入以下代码。

```

01 Private Sub Option1_Click()
02     Text1.Visible = True           ' 设置文本框控件可见

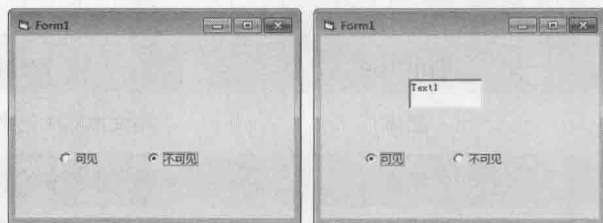
```

03 End Sub

(4) 双击【不可见】单选按钮，在代码窗口中输入以下代码。

```
01 Private Sub Option2_Click()  
02 Text1.Visible = False ' 设置文本框控件不可见  
03 End Sub
```

(5) 运行程序查看效果，单击【不可见】按钮效果如左下图所示，单击【可见】按钮效果如右下图所示。



## 9.6 复选框控件

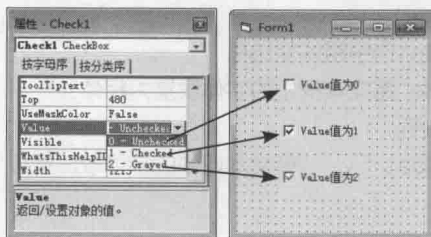


本节视频教学录像：10 分钟

复选框控件与单选按钮控件一样，具有“选中”和“未选中”两种状态，在程序设计中常常用来设置某一属性的多个选择项。当复选框被选中时，控件左边的方框中就会出现一个“√”号。

### 9.6.1 复选框的主要属性

Value 属性用来表示用户的选择情况，属性值为 0 表示没有选中该复选框，值为 1 表示选中该复选框，值为 2 表示该复选框不可选。可以通过复选框所对应的属性窗口中的【Value】下拉列表设置。




### 9.6.2 复选框的常用事件

复选框的常用事件和单选按钮一样为鼠标单击事件（Click 事件）。用户在复选按钮上单击鼠标时激活该事件。

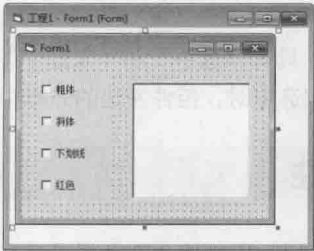
9.6.3 复选框控件应用示例

**【范例 9-5】**在文本框中输入一段文字，通过对复选框的选择来改变所输入文字的字体、颜色等。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在窗体中添加 4 个复选框控件和 1 个文本框控件，将 Text1 的名称属性设置为“Txt Show”，并将 Text 的属性值清空。4 个复选框的 Caption 属性设置如表所示。

名称	Caption	控件作用
ChkBold	粗体	将文本框中的字体设置为粗体
ChkItalic	斜体	将文本框中的字体设置为斜体
ChkUnderLine	下划线	为文本框中的文字设置下划线
ChkRed	红色	将文本框中的文字颜色设置为红色

- (3) 将各个控件按照下图所示排列。



- (4) 双击复选按钮【粗体】，进入代码窗口，输入以下代码（代码 9-5-1.txt）。

```
01 Private Sub ChkBold_Click()  
02   If ChkBold.Value = 1 Then ' 选中粗体复选框  
03     TxtShow.FontBold = True' 将文本框中的字体设置为粗体  
04   Else   ' 未选中粗体复选框  
05     TxtShow.FontBold = False   ' 将文本框中的字体取消粗体  
06   End If   ' 结束循环  
07 End Sub
```

- (5) 双击复选按钮【斜体】，进入代码窗口，输入以下代码（代码 9-5-2.txt）。

```
01 Private Sub ChkItalic_Click()  
02   If ChkItalic.Value = 1 Then ' 选中斜体复选框  
03     TxtShow.FontItalic = True' 将文本框中的字体设置为斜体  
04   Else   ' 未选中斜体复选框
```

```
05 TxtShow.FontItalic = False      '字体恢复为正体
06 End If '结束循环
07 End Sub
```


(6) 双击复选按钮【下划线】，进入代码窗口，输入以下代码（代码 9-5-3.txt）。

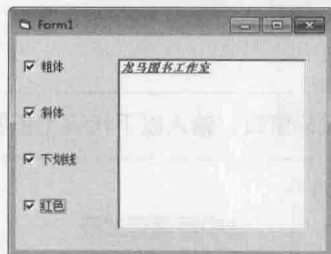
```
01 Private Sub ChkUnderLine_Click()
02 If ChkUnderLine.Value = 1 Then '选中下划线复选框
03 TxtShow.FontUnderline = True '为文本框中的字体设置下划线
04 Else '未选中下划线复选框
05 TxtShow.FontUnderline = False '将文本框中的字体取消下划线
06 End If '结束循环
07 End Sub
```

(7) 双击复选按钮【红色】，进入代码窗口，输入以下代码（代码 9-5-4.txt）。

```
01 Private Sub ChkRed_Click()
02 If ChkRed.Value = 1 Then '选中红色复选框
03 TxtShow.ForeColor = &HFF& '将文本框中的字体颜色设置为红色
04 Else '未选中红色复选框
05 TxtShow.ForeColor = &H0& '将文本框中的字体颜色设置为黑色
06 End If '结束循环
07 End Sub
```


## 【运行结果】

保存程序，单击【启动】按钮 ，在文本框中输入文字，并单击复选框按钮。



## 【拓展训练 9-5】

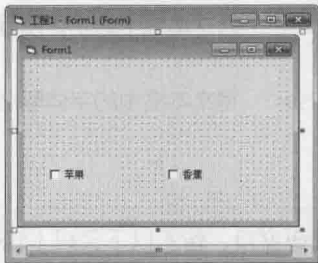
复选框最大的优点，就是可以贪婪地选择自己喜欢的东西，下面就来演示一下对喜爱的水果如何进行多次选择。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中添加两个复选框控件和一个标签控件，将 Label1 的名称属性设置为“LbShow”，并将 Caption 的属性值清空。两个复选框的 Caption 属性设置如表所示。

名称	Caption	控件作用
Chkapple	苹果	标签中将显示喜欢的水果有苹果
Chkbanana	香蕉	标签中将显示喜欢的水果为有香蕉

(3) 将各个控件按照下图所示排列。



(4) 双击复选按钮【苹果】，进入代码窗口，输入以下代码（拓展代码 9-5-1.txt）。

```
01 Private Sub Chkapple_Click()  
02     If Chkapple.Value = 1 Then          ' 选中苹果复选框  
03         If Chkbanana.Value = 1 Then      ' 同时选中香蕉复选框  
04             LbShow.Caption = "最喜欢的水果不仅有 " & Chkapple.Caption & " 还有 " & Chkbanana.Caption  
            ' 标签中显示最喜欢的水果为苹果和香蕉  
05         Else  
06             LbShow.Caption = "最喜欢的水果为 " & Chkapple.Caption ' 标签中显示最喜欢的水果为苹果  
07         End If ' 结束内判断  
08     Else ' 苹果复选框没有选中，香蕉复选框选中  
09         LbShow.Caption = "最喜欢的水果为 " & Chkbanana.Caption ' 标签中显示最喜欢的水果为香蕉  
10     End If ' 结束外判断  
11 End Sub
```

(5) 双击复选按钮【香蕉】，进入代码窗口，输入以下代码（拓展代码 9-5-2.txt）。

```
01 Private Sub Chkbanana_Click()  
02     If Chkbanana.Value = 1 Then          ' 选中香蕉复选框  
03         If Chkapple.Value = 1 Then      ' 同时选中苹果复选框  
04             LbShow.Caption = "最喜欢的水果不仅有 " & Chkapple.Caption & " 还有 " & Chkbanana.Caption  
            ' 标签中显示最喜欢的水果为苹果和香蕉  
05         Else  
06             LbShow.Caption = "最喜欢的水果为 " & Chkbanana.Caption ' 标签中显示最喜欢的水果为香蕉  
07         End If ' 结束内判断  
08     Else ' 香蕉复选框没有选中，苹果复选框选中  
09         LbShow.Caption = "最喜欢的水果为 " & Chkapple.Caption ' 标签中显示最喜欢的水果为苹果  
10     End If ' 结束外判断  
11 End Sub
```

(6) 运行程序查看效果。

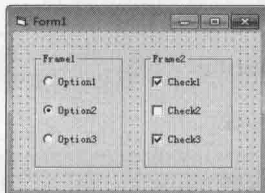


## 9.7 框架控件



本节视频教学录像：4 分钟

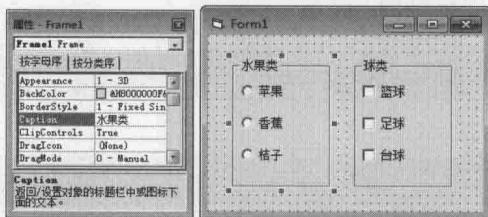
可以将框架控件 (Frame) 看作是一个容器控件，用于为控件进行可标识的分组。例如，可以对单选按钮、复选框等控件进行分组，以使软件界面的逻辑更加清晰，用户使用起来更加方便。



### 9.7.1 框架的主要属性


标题属性 (Caption)。

Caption 属性用于设置框架控件的显示标题，以此对框架内的控件组进行说明。



### 9.7.2 框架控件应用示例

**【范例 9-6】** 本例所实现的功能是：根据所选择的分类进行选择，然后单击【确定】按钮，将所选择的内容显示在标签控件中。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中创建两个框架 (Frame)，并将这些控件的属性按照下表所示设置。



名称	Caption	控件作用
Frame1	饮料	用于分组显示饮料选项
Frame2	运动	用于分组显示运动选项

(3) 在第 1 个框架 (Frame1) 中创建 3 个单选按钮, 并将这些单选按钮的属性按照下表所示设置。

名称	Caption	控件作用
OptDrink1	可乐	用户饮料选项 1
OptDrink2	啤酒	用户饮料选项 2
OptDrink3	绿茶	用户饮料选项 3

(4) 在第 2 个框架 (Frame2) 中创建 3 个复选框, 并将这些复选框的属性按照下表所示设置。

名称	Caption	控件作用
ChkSport1	足球	用户运动选项 1
ChkSport2	篮球	用户运动选项 2
ChkSport3	跑步	用户运动选项 3

(5) 在窗体中创建 4 个标签控件 (Label), 并将这些标签的属性按照下表所示设置。

名称	标题属性 (Caption)	控件作用
LblNam1	您最喜欢的饮料是 :	提示显示饮料信息
LblNam2	你最喜欢的运动是 :	提示显示运动信息
Lbl1	“ ”	显示饮料信息
Lbl2	“ ”	显示运动信息

(6) 在窗体中添加两个控制按钮, 把 Command 1 的名称属性设置为 “Cmd1”, Caption 属性设置为 “确定”; 把 Command2 的名称属性设置为 “Cmd2”, Caption 属性设置为 “取消”。将所有控件按照下图所示排列。




(7) 双击【确定】按钮，进入代码窗口，输入以下代码（代码 9-6-1.txt）。

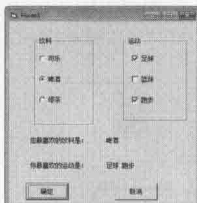
```
01 Private Sub Cmd1_Click()
02   Lbl1.Caption = ""      ' 将 Lbl1 的 Caption 属性设置为空
03   Lbl2.Caption = ""      ' 将 Lbl2 的 Caption 属性设置为空
04   ' 以上实现将上一次显示的信息消除
05   If OptDrink1.Value = True Then    ' 选中 OptDrink1 单选按钮
06     Lbl1.Caption = OptDrink1.Caption    ' 在 Lbl1 中显示 OptDrink1 的信息
07   End If    结束循环
08   If OptDrink2.Value = True Then    ' 选中 OptDrink2 单选按钮
09     Lbl1.Caption = OptDrink2.Caption    ' 在 Lbl1 中显示 OptDrink2 的信息
10   End If    ' 结束循环
11   If OptDrink3.Value = True Then    ' 选中 OptDrink3 单选按钮
12     Lbl1.Caption = OptDrink3.Caption    ' 在 Lbl1 中显示 OptDrink3 的信息
13   End If    ' 结束循环
14   If ChkSport1.Value = 1 Then        ' 选中 ChkSport1 复选框
15     Lbl2.Caption = ChkSport1.Caption    ' 在 Lbl2 中显示 ChkSport1 的信息
16   End If    ' 结束循环
17   If ChkSport2.Value = 1 Then        ' 选中 ChkSport2 复选框
18     Lbl2.Caption = Lbl2.Caption & " " & ChkSport2.Caption
19     ' 在 Lbl2 中显示 ChkSport2 的信息
20   End If    ' 结束循环
21   If ChkSport3.Value = 1 Then        ' 选中 ChkSport3 复选框
22     Lbl2.Caption = Lbl2.Caption & " " & ChkSport3.Caption
23     ' 在 Lbl2 中显示 ChkSport3 的信息
24   End If    ' 结束循环
25 End Sub
```

(8) 双击【取消】按钮，进入代码窗口，输入以下代码。

```
01 Private Sub Cmd2_Click()
02   End    ' 退出程序
03 End Sub
```

## 【运行结果】

保存程序，单击【启动】按钮, 选择相关选项，然后单击【确定】按钮。



## 9.8 列表框控件



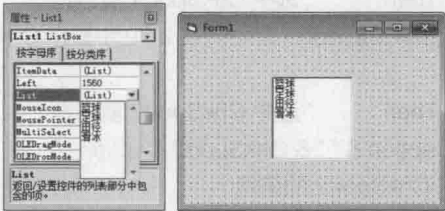
本节视频教学录像：24 分钟

列表框控件用于显示项目列表，供用户进行多个项目的选择。当项目总数超过可显示的项目数后，就会自动在列表框控件上添加滚动条。

### 9.8.1 列表框的主要属性

#### 1. 列表属性 (List)

该属性用于设置列表中的内容。列表框中所有的选项都可以通过 List (下标值) 的形式指定。列表框中的第 1 项用 List(0) 表示，列表框中的第 2 项用 List(1) 表示，依次类推。需要注意的是，列表框的下标值和数组下标一样都是从 0 开始。用户可以在属性窗口中的 List 属性输入框中直接输入内容来添加列表项。



在向 List 属性值中输入数据时，按【Enter】键表示结束数据输入，按【Ctrl + Enter】组合键表示换行继续输入数据。

提示

#### 2. 项目数目属性 (ListCount)

该属性返回列表框表项数量值。语法是：

X = 列表框名称.ListCount (X 为数值变量)

例如：

```
01 Private Sub Form_Load()  
02 Label1.Caption = "列表有 " & List1.ListCount & " 个项目 " ' 显示列表项目总数  
03 End Sub
```



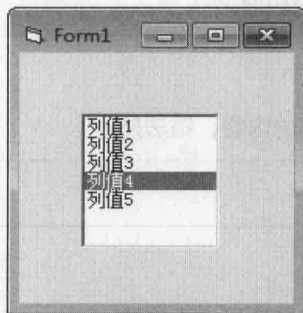
### 3. 索引属性 (ListIndex)

该属性返回控件中当前选择项目的索引号。第 1 个选项的索引号是 0，第 2 个选项的索引号是 1，依次类推。当列表框没有选择项目时，ListIndex 值为 -1。可以根据返回的索引号进行相应的操作，语法是：

X% = 列表框名称.ListIndex

例如：

```
01 Private Sub Form_Load()  
02 List1.ListIndex = 3      ' 列表项目中的第 4 项被选中  
03 End Sub
```



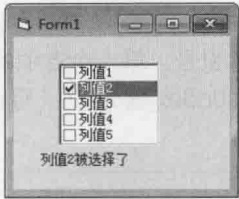
## 9.8.2 列表框的主要事件

列表框控件的主要事件为 Click (单击) 事件与 ItemCheck (项目检查) 事件。Click 事件用于当鼠标选中列表框控件中的列表项时所触发的事件；ItemCheck 事件主要用于当 Style 属性设置为 1，并且列表控件中一个项目的复选框被选中或撤选时，所发生的事件。

例如：在 List 控件下的 ItemCheck 事件下输入以下代码。

```
01 Private Sub List1_ItemCheck(Item As Integer)  
02 Dim i As Integer ' 定义整型变量  
03 For i = 0 To List1.ListCount - 1 ' 用计数循环判断所选项目列表状态  
04 List1.Selected(i) = False ' 所有项目列表前的复选框中不显示“√”号  
05 List1.Selected(List1.ListIndex) = True ' 所选项目列表前的复选框中显示“√”号  
06 Next ' 条件满足，进行下一次循环  
07 Label1.Caption = "" ' 将标签的标题属性清空  
08 Label1.Caption = Label1.Caption & List1.List(List1.ListIndex) & " 被选择了" ' 显示选择的项目  
09 End Sub
```

运行程序，查看效果。



### 9.8.3 列表框控件的方法

1. 增加项目（AddItem）

使用该方法可以为列表框增加项目，语法是：

列表框名称.AddItem 欲增项目 [ , 索引值 ]

2. 清除所有（Clear）

使用该方法可以清除列表框中所有的内容，语法是：

列表框名称.Clear

3. 删除选项（RemoveItem）


使用该方法可以删除列表框中指定的项目，语法是：

列表框名称.RemoveItem 索引值

### 9.8.4 列表框控件应用示例

【范例 9-7】本例实现对列表框中所选择的列表项，传递至另一个窗体中显示。

1. 界面设计

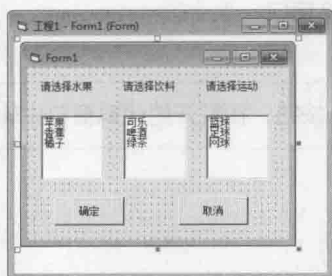
- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在窗体中创建 3 个列表框（ListBox），并将这些列表框的属性按照下表所示设置。


控件名称	名称	MultiSelect	控件作用
List1	苹果、香蕉、橘子	0 - None	用于列表显示水果选项
List2	可乐、啤酒、绿茶	0 - None	用于列表显示饮料选项
List3	篮球、足球、网球	2 - Extended	用于列表显示运动选项

- (3) 在 Form1 中创建 3 个标签控件，并将这些标签的属性按照下表所示设置。

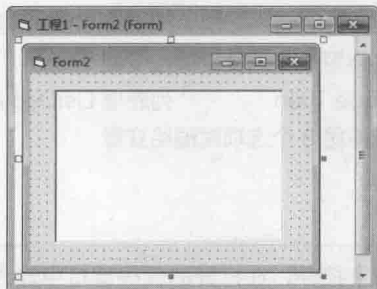
名称	Caption	控件作用
Lbl1	请选择水果	提示用户选择水果选项
Lbl2	请选择饮料	提示用户选择饮料选项
Lbl3	请选择运动	提示用户选择运动选项

(4) 在 Form1 中创建两个控制按钮，设置 Command1 的名称属性为“Cmd1”，Caption 属性为“确定”；设置 Command2 的名称属性为“Cmd2”，Caption 属性为“取消”。窗体各控件的排列如图所示。



(5) 选择【工程】>【添加窗体】菜单命令，在弹出的【添加窗体】对话框中选择【新建】选项卡下的【窗体】图标，单击【打开】按钮，为工程添加一个窗体 Form2。

(6) 在 Form2 中添加文本框控件 Text1，将该控件的 Text 属性值设置为空，并将 MultiLine 属性为“True”。将 Form2 中的控件拖动到合适的位置，完成后的 Form2 效果如图所示。



## 2. 编写代码

(1) 在 Form1 中没有控件的任意空白处右击，在弹出的快捷菜单中选择【查看代码】选项，在弹出的代码窗口中输入以下代码。

```

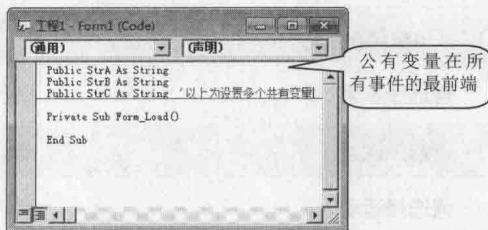
01 Public StrA As String      ' 设置公有变量，用于记录列表框 1 中的列表项
02 Public StrB As String      ' 设置公有变量，用于记录列表框 2 中的列表项
03 Public StrC As String      ' 设置公有变量，用于记录列表框 3 中的列表项
    
```



**提示**

公有变量应放在所有事件的最前端。





(2) 双击 Form1 窗体上的【确定】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub Cmd1_Click()
02     Form2.Show '显示 Form2 窗口
03 End Sub
```

(3) 双击 Form1 窗体上的【取消】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub Cmd2_Click()
02 End '程序结束
03 End Sub
```

(4) 双击 Form1 窗体上的【List1】控件，在打开的代码窗口中输入以下代码（代码 9-7-1.txt）。

```
01 Private Sub List1_Click()
02     If List1.Selected(0) = True Then '列表框 ListDeg 第 1 个选项被选中
03         StrA$ = List1.List(0) '将第 1 个选项赋值给变量
04     ElseIf List1.Selected(1) = True Then '列表框 ListDeg 第 2 个选项被选中
05         StrA$ = List1.List(1) '将第 2 个选项赋值给变量
06     ElseIf List1.Selected(2) = True Then '列表框 ListDeg 第 3 个选项被选中
07         StrA$ = List1.List(2) '将第 3 个选项赋值给变量
08 End If '结束条件语句
09 End Sub
```

(5) 双击 Form1 窗体上的【List2】控件，在打开的代码窗口中输入以下代码（代码 9-7-2.txt）。

```
01 Private Sub List2_Click()
02     If List2.Selected(0) = True Then '列表框 List2 第 1 个选项被选中
03         StrB$ = List2.List(0) '将第 1 个选项赋值给变量
04 End If '结束条件语句
05     If List2.Selected(1) = True Then '列表框 List2 第 2 个选项被选中
06         StrB$ = List2.List(1) '将第 2 个选项赋值给变量
07 End If '结束条件语句
08     If List2.Selected(2) = True Then '列表框 List2 第 3 个选项被选中
09         StrB$ = List2.List(2) '将第 3 个选项赋值给变量
10 End If '结束条件语句
11 End Sub
```

(6) 双击 Form1 窗体上的【List3】控件，在打开的代码窗口中输入以下代码（代码 9-7-3.txt）。

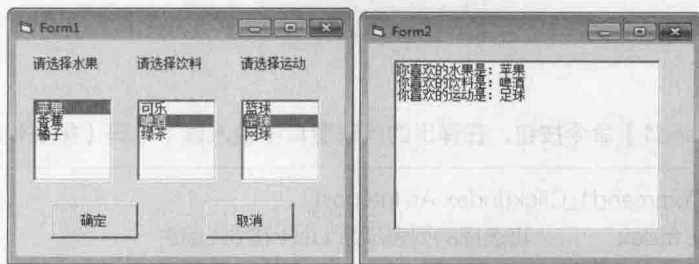
```
01 Private Sub List3_Click()  
02 If List3.Selected(0) = True Then '列表框 List3 第 1 个选项被选中  
03 StrC$ = List3.List(0) '将第 1 个选项赋值给变量  
04 ElseIf List3.Selected(1) = True Then '列表框 List3 第 2 个选项被选中  
05 StrC$ = List3.List(1) '将第 2 个选项赋值给变量  
06 ElseIf List3.Selected(2) = True Then '列表框 List3 第 3 个选项被选中  
07 StrC$ = List3.List(2) '将第 3 个选项赋值给变量  
08 End If  
09 End Sub
```

(7) 在 Form2 窗口的空白处双击，在打开的代码窗口中输入以下代码。

```
01 Private Sub Form_Load()  
02 Text1.Text = "你喜欢的水果是：" & Form1.StrA & vbCrLf _ '显示列表框 1 中的信息并换行操作  
03 & "你喜欢的饮料是：" & Form1.StrB & vbCrLf _ '显示列表框 2 中的信息  
& "你喜欢的运动是：" & Form1.StrC & vbCrLf '显示列表框 3 中的信息  
05 End Sub
```

## 【运行结果】

选择【工程】>【工程属性】菜单命令，在【工程属性】对话框中设置 Form1 为启动窗体。保存程序，并按快捷键【F5】运行程序，在列表框中分别选择一项，如左下图所示。单击【确定】按钮，选择的各项将显示在弹出的窗口中，效果如右下图所示。




**提示**

如果在【工程属性】对话框中设置 Form2 为启动窗体，按快捷键【F5】运行程序后将会直接启动 Form2 窗体。

## 【拓展训练 9-6】

使用列表框将一个列表框中的内容移动到另一个列表框中。

## 1. 界面设计

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标  , 然后单击【打开】按钮。

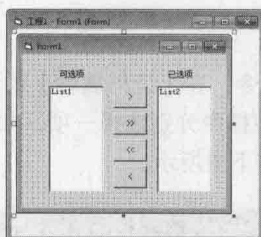
(2) 在窗体中添加两个标签 (Label) 控件, 将“Caption”属性分别设置为“可选项”和“已选项”。

(3) 在窗体中添加两个列表框 (ListBox) 控件, 如图所示调整控件的大小及位置。

(4) 在窗体中添加 1 个命令按钮 (CommandButton) 控件, 将其“Caption”属性设置为“>” (大于号)。

(5) 右击步骤(4)中所创建的 Command1 控件, 选择【复制】选项, 在窗体中连续粘贴 3 次, 完成创建包含 4 个命令按钮控件的控件组。各控件的属性设置如表所示。再如图所示调整各控件的大小及位置。

控件名称	Index	Caption	控件功能
Command1	0	> (大于号)	向右移动 1 项
Command1	1	>> (两个大于号)	向右移动所有项
Command1	2	<< (两个小于号)	向左移动 1 项
Command1	3	< (小于号)	向左移动所有项



## 2. 编写代码

(1) 双击【Command1】命令按钮, 在弹出的代码窗口中输入以下代码 (拓展代码 9-6-1.txt)。

```

01 Private Sub Command1_Click(Index As Integer)
02     Select Case Index      ' 将选择的列表项从 List1 移到 List2
03     Case 0
04         If List1.ListCount = 0 Then      ' 判断当前列表项个数是否为零
05             Exit Sub      ' 列表中无项目退出程序
06         End If
07         If List1.ListIndex = -1 Then      ' 判断列表项有无选中
08             List1.SetFocus      ' 设置列表项焦点
09             List1.Selected(0) = True      ' 列表项第 1 项被选中
10         End If
11         DoEvents      ' 防止列表项数目过多, 移动时死机
12         List2.AddItem List1.Text      ' 将选择的第 1 个列表框中的项目添加至第 2 个列表框中

```

```

13 List1.RemoveItem List1.ListIndex '添加后清除第1个列表框中所选择的项目
14 Case 1 '将 List1 的所有列表项移到 List2 中
15 If List1.ListCount = 0 Then '判断当前列表项个数是否为零
16 Exit Sub '列表中无项目退出程序
17 End If
18 If List1.ListIndex = -1 Then '判断列表项有无选中
19 List1.SetFocus '设置列表项焦点
20 List1.Selected(0) = True '列表项第1项被选中
21 End If
22 DoEvents '防止列表项数目过多, 移动时死机
23 For i = (List1.ListCount - 1) To 0 Step -1 '开始从后向前移动项目
24 List2.AddItem List1.List(i) '将 List1 的所有列表项添加到 List2 中
25 DoEvents '防止列表项数目过多, 添加时死机
26 Next i
27 List1.Clear '删除 List1 中的所有列表项
28 Case 2 '将选择的列表项从 List2 移到 List1
29 If List2.ListCount = 0 Then '判断当前列表项个数是否为零
30 Exit Sub '如果 List2 中没有列表项则退出
31 End If
32 If List2.ListIndex = -1 Then '判断列表项有无选中
33 List2.SetFocus '设置列表项焦点
34 List2.Selected(0) = True '列表项第1项选中
35 End If
36 List1.AddItem List2.Text '将选择的第2个列表框中项目添加至第1个列表框中
37 List2.RemoveItem List2.ListIndex '添加后清除第2个列表框中所选择的项目
38 Case 3 '将 List2 的所有列表项移到 List1 中
39 If List2.ListCount = 0 Then '判断当前列表项个数是否为零
40 Exit Sub '如果 List2 中没有列表项则退出
41 End If
42 If List2.ListIndex = -1 Then '判断列表项有无选中
43 List2.SetFocus '设置列表项焦点
44 List2.Selected(0) = True '列表项第1项被选中
45 End If
46 For i = (List2.ListCount - 1) To 0 Step -1 '开始从后向前移动项目
47 List1.AddItem List2.List(i) '将 List2 的所有列表项添加到 List1 中
48 DoEvents '防止列表项数目过多, 添加时死机
49 Next i
50 List2.Clear '删除 List2 中的所有列表项
51 End Select
52 End Sub

```

(2) 在 Form1 窗体的空白处双击鼠标, 进入代码窗口, 然后输入以下代码。

```

01 Private Sub Form_Load()
02 List1.AddItem "姓名", 0 '对列表框1追加项目, 追加的项目位于第1行

```

```

03 List1.AddItem "性别", 1 '对列表框 1 追加项目, 追加的项目位于第 2 行
04 List1.AddItem "年龄", 2 '对列表框 1 追加项目, 追加的项目位于第 3 行
05 List1.AddItem "籍贯", 3 '对列表框 1 追加项目, 追加的项目位于第 4 行
06 List1.AddItem "文化程度", 4 '对列表框 1 追加项目, 追加的项目位于第 5 行
07 End Sub

```



提示

步骤(2)中的代码主要用于在启动窗体时, 向 List1 列表框控件中加载数据。

### 3. 运行程序

(1) 保存程序并按快捷键【F5】运行程序。在【可选项】列表中选择一项, 单击【>】按钮, 即可将所选项移动到【已选项】列表框中。

(2) 单击【>>】按钮, 即可将所有的【可选项】移动到【已选项】列表框中。



## 9.9 组合框控件



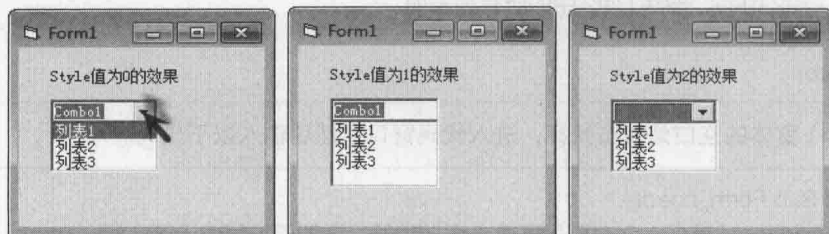
本节视频教学录像: 6 分钟

组合框控件 (ComboBox) 是一种更为灵活的控件, 既可以像列表框一样, 让用户选择所需项目, 又可以如文本框一样输入文本。

### 9.9.1 组合框控件的主要属性

#### 1. 风格属性 (Style)

该属性用于设置列表框的外观, 可以通过属性窗口中的 Style 选项进行设置, 共有 3 个值备选: 0、1 和 2。它们分别对应 3 种下拉框, 即下拉式组合框、简单组合框和下拉式列表框。



## 2. 文本属性 (Text)

该属性返回用户选择的文本或直接在编辑区域输入的文本，用户可以通过属性窗口中的 Text 属性输入框进行设置。

### 9.9.2 组合框的事件和方法


不同组合框的类型所响应的事件是不同的，例如 Change 事件 (指示改变控件的文本框部分的正文) 仅在 Style 属性设置为 0 或 1，且正文被改变或者通过代码改变了 Text 属性的设置时才会发生。

组合框的方法和列表框大同小异，AddItem、Clear 和 RemoveItem 等方法也适用于组合框。

### 9.9.3 组合框应用示例

**【范例 9-8】** 本例主要通过通过组合框的选择来改变文本框中文本的字体、字型 and 大小等。

#### 1. 界面设计

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在窗体中创建 3 个组合框 (ComboBox)，并将这些组合框的属性按照下表所示设置。

名称	List 属性	Text 属性	控件功能
CboFont	宋体	“”	用于显示字体选项
	隶书		
	楷体_GB2312		
	黑体		
CboFontStyle	常规	“”	用于显示字形选项
	粗体		
	斜体		
	粗斜体		
CboFontSize	8	“”	用于显示字号选项
	9		
	10		
	11		
	12		
	14		
	16		
	18		
	20		

(3) 在窗体中创建一个包含 4 个标签控件的 Label 控件组 (先创建一个 Label 控件，再采用复制粘

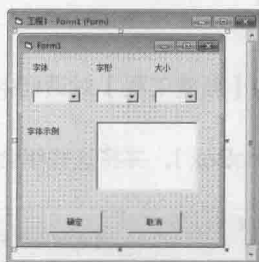


贴的方法创建控件组),并将这些标签的属性按照下表所示设置。

控件名称	Caption	作用
Lbl1(0)	字体	提示用户后面显示的组合框为“字体”组合框
Lbl1(1)	字形	提示用户后面显示的组合框为“字形”组合框
Lbl1(2)	大小	提示用户后面显示的组合框为“大小”组合框
Lbl1(3)	字体示例	提示用户后面显示的组合框为“字体示例”组合框

(4) 在窗体中创建一个文本框 (TextBox), 将文本框 Text1 的名称改为 “Txt1”, Text 属性设置为 “”, MultiLine 属性设置为 “True”。

(5) 在窗体中创建一个命令按钮控件组 (先创建一个 Command 控件, 然后复制粘贴此控件), 把控件组 Command1 的名称属性设置为 “Cmd 1”。把 Cmd1(0) 的 Caption 属性设置为 “确定”, Cmd1(1) 的 Caption 属性设置为 “取消”。将所有控件摆放整齐, 如图所示。



## 2. 编写代码

(1) 双击 Form1 窗体上名称为 CboFont 的组合框控件, 进入代码窗口, 输入以下代码。

```
01 Private Sub CboFont_Change() ' 字体组合框单击事件代码
02   Lbl1(3).FontName = CboFont.Text ' 按选中的选项设置标签中字体
03 End Sub
```

(2) 双击 Form1 窗体上名称为 CboFontStyle 的组合框控件, 进入代码窗口, 输入以下代码 (代码 9-8-1.txt)。

```
01 Private Sub CboFontStyle_Change() ' 字体大小组合框单击事件代码
02   If CboFontStyle.Text = "常规" Then ' 若选中的选项为常规执行下面代码
03     Lbl1(3).FontBold = False ' 取消粗体
04     Lbl1(3).FontItalic = False ' 取消斜体
05   End If ' 结束条件语句
06   If CboFontStyle.Text = "粗体" Then ' 若选中的选项为粗体执行下面代码
07     Lbl1(3).FontBold = True ' 设置标签中字体为粗体
08   End If ' 结束条件语句
09   If CboFontStyle.Text = "斜体" Then ' 若选中的选项为斜体执行下面代码
10     Lbl1(3).FontItalic = True ' 设置标签中字体为斜体
```

```

11 End If '结束条件语句
12 If CboFontStyle.Text = "粗斜体" Then '若选中的选项为粗斜体执行下面代码
13     Lbl1(3).FontBold = True '设置标签中字体为粗体
14     Lbl1(3).FontItalic = True '设置标签中字体为斜体
15 End If '结束条件语句
16 End Sub

```

(3) 双击 Form1 窗体上名称为 CboFontSize 的组合框控件，进入代码窗口，输入以下代码。

```

01 Private Sub CboFontSize_Change() '字体大小组合框单击事件代码
02     Lbl1(3).FontSize = CboFontSize.Text '按选中的选项设置标签中字体大小
03 End Sub

```

(4) 双击 Form1 窗体上名称为 Cmd1 的命令按钮控件，进入代码窗口，输入以下代码（代码 9-8-2.txt）。

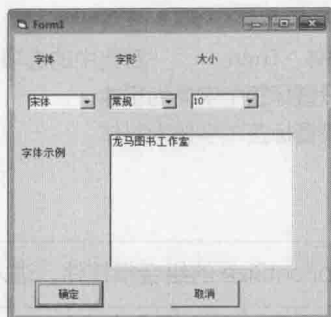
```

01 Private Sub Cmd1_Click(Index As Integer) '命令按钮单击事件代码
02     If Index = 0 Then '若选中确定按钮执行下面代码
03         Txt1.FontName = CboFont.Text '按选中的选项设置文本框中字体
04         Txt1.FontSize = CboFontSize.Text '按选中的选项设置文本框中字体大小
05         If CboFontStyle.Text = "常规" Then '若选中的选项为常规执行下面代码
06             Txt1.FontBold = False '取消文本框中的粗体
07             Txt1.FontItalic = False '取消文本框中的斜体
08         End If '结束条件语句
09         If CboFontStyle.Text = "粗体" Then '若选中的选项为粗体执行下面代码
10             Txt1.FontBold = True '设置文本框中字体为粗体
11         End If '结束条件语句
12         If CboFontStyle.Text = "斜体" Then '若选中的选项为斜体执行下面代码
13             Txt1.FontItalic = True '设置文本框中字体为斜体
14         End If '结束条件语句
15         If CboFontStyle.Text = "粗斜体" Then '若选中的选项为粗斜体执行下面代码
16             Txt1.FontBold = True '设置文本框中字体为粗体
17             Txt1.FontItalic = True '设置文本框中字体为斜体
18         End If '结束条件语句
19     End If
20     If Index = 1 Then '若选中取消按钮执行下面代码
21     End '结束程序
22 End If '结束条件语句
23 End Sub

```


## 【运行结果】

保存程序，并按快捷键【F5】运行程序。在文本框中输入文字，依次选择【字体】、【字形】和【大小】等下拉列表中的选项，然后单击【确定】按钮，即可更改文本框内文字的字体、字形和大小。

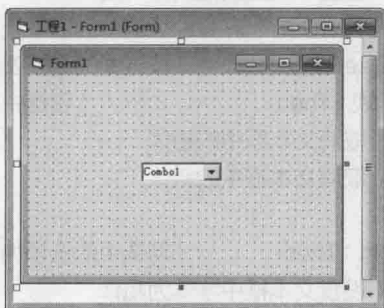


### 【拓展训练 9-7】

使用组合框控件，用户可以很方便地选择数据，从而可减少手动输入的麻烦。利用组合框控件还可以自定义日期，例如可以利用一个组合框控件显示出 1949 ~ 2049 年的所有年份。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 窗体中添加一个组合框控件。



(3) 在 Form1 窗体的任意空白处双击，打开代码窗口，输入以下代码（拓展代码 9-7.txt）。

```
01 Private Sub Form_Load()  
02 Dim i As Integer  
03 Combo1.Text = "" '清空 Combo1 控件中的内容  
04 For i = 1949 To 2049 '利用循环显示年份  
05 With Combo1  
06 .AddItem i '添加年份  
07 End With  
08 Next  
09 Combo1.ListIndex = 0 '使第 1 项处于选中状态  
10 End Sub
```



**提示**

在窗体的 Load 事件中主要使用了一个 With 语句，可以对 Combo1 对象执行一系列的语句，而不用重复指出该对象的名称。

(4) 保存程序，并按快捷键【F5】运行程序，结果如图所示。



## 9.10 图像框控件



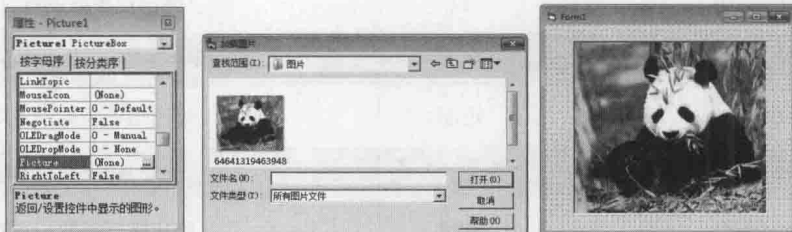
本节视频教学录像：6 分钟

图像框 (PictureBox) 控件既可以用来显示图形，也可以用来在控件上输出图形和使用 Print 方法输出文本。

### 9.10.1 图像框控件的主要属性

#### 1. 图片属性 (Picture)

该属性用来返回或设置控件中要显示的图片。单击图片框控件的【Picture】属性，会打开【加载图片】对话框，从中选择所要加载的图片，即可将图片加载到控件上。



#### 2. 自动调整属性 (AutoSize)

该属性用于设置图片是否能够自动调整大小以完整显示。用户可以通过属性窗口中的【AutoSize】下拉列表进行设置。

### 9.10.2 图像框控件的主要事件和方法

该控件可以被 Click (单击) 事件和 DblClick (双击) 事件所触发，还可以在图像框中使用 Cls (清屏) 和 Print 等方法。下面介绍图像框控件常用的 Cls 方法和 Print 方法。

#### 1. Cls (清屏) 方法

该方法的作用是把图像框中所有的内容清空，语法如下。

图像框名称 .Clear

2. Print 方法

该方法的作用是在图像框中输出文本，语法如下。


图像框名称 .print" 文本内容 "

9.10.3 图像框应用示例

自制一个画图板，通过输入所要绘制图形的各个参数，就可以绘制出一个满意的图形。

**【范例 9-9】**通过对直线、圆、矩形等参数的设置，使其在图像框中绘制出相应的图形。

1. 界面设计

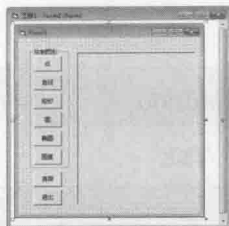
- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在窗体中创建 1 个框架控件（Frame），把 Frame1 的 Caption 属性设置为“绘制图形”。
- (3) 在 Frame1 中添加 1 个包含 6 个命令按钮的命令按钮组（先创建一个控件，再采用复制粘贴的方法创建控件组），并将这些命令按钮的属性按照下表所示设置。


控件名称	Caption	控件作用
CmdPicture(0)	点	弹出点参数设置对话框
CmdPicture(1)	直线	弹出直线参数设置对话框
CmdPicture(2)	矩形	弹出矩形参数设置对话框
CmdPicture(3)	圆	弹出圆参数设置对话框
CmdPicture(4)	椭圆	弹出椭圆参数设置对话框
CmdPicture(5)	圆弧	弹出圆弧参数设置对话框

- (4) 在 Form1 中添加 1 个包含两个命令按钮的命令按钮组，并将这些命令按钮的属性按照下表所示设置。

控件名称	Caption	控件作用
Cmd1(0)	清屏	将图像框中显示的图像清除
Cmd1(1)	退出	退出程序

- (5) 在 Form1 中添加 1 个图像框控件，把 Picture1 图像框的名称属性设置为“PicShow”。将各控件按照下图所示排列。



(6) 选择【工程】菜单下的【添加窗体】菜单项，在【添加窗体】对话框中选择【窗体】图标 ，在工程中再添加一个窗体 Form2。

(7) 在 Form2 中添加包含两个文本框的文本框控件组 Text1 和 Text2，再添加 5 个文本框，将这些控件的属性按照下表所示设置。

控件名称	Text	控件作用
TxtPoint1(0)	“ ”	输入点 1 的横坐标
TxtPoint1(1)	“ ”	输入点 1 的纵坐标
TxtPoint2(0)	“ ”	输入点 2 的横坐标
TxtPoint2(1)	“ ”	输入点 2 的纵坐标
TxtRad	“ ”	用于输入半径参数
TxtColor	“ ”	用于输入颜色参数
TxtScale	“ ”	用于输入长短轴比率参数
TxtAng1	“ ”	用于输入起始角参数
TxtAng2	“ ”	用于输入终止角参数

(8) 在 Form2 窗体中添加 7 个标签控件，属性设置如下表所示。

控件名称	Caption	控件作用
Label1	输入点 1 坐标：	提示用户输入点 1 的坐标
Label2	输入点 2 坐标：	提示用户输入点 2 的坐标
Label3	输入颜色：	提示用户输入颜色参数
Label4	输入半径：	提示用户输入半径参数
Label5	输入起始角：	提示用户输入起始角参数
Label6	输入终止角：	提示用户输入终止角参数
Label7	长短轴比率：	提示用户输入长短轴比率参数

(9) 在 Form2 中添加 1 个包含两个命令按钮的命令按钮组，属性设置如下表所示。将各控件按照下



图所示排列。

控件名称	Caption	控件作用
Cmd2(0)	确定	按照输入的参数进行绘图
Cmd2(1)	取消	取消输入的参数，返回主界面



2. 编写代码

(1) 在 Form1 窗体中没有控件的任意空白处右击，在弹出的快捷菜单中选择【查看代码】选项，在弹出的代码窗口中定义公有变量。

```
Public IntA As Integer ' 定义变量
```

(2) 在 Form1 窗体中双击名称为 Cmd1 的按钮控件，进入代码窗体，输入以下代码（代码 9-9-1.txt）。

```
01 Private Sub Cmd1_Click(Index As Integer) ' 命令按钮组 Cmd1 的鼠标单击事件代码
02 If Index = 0 Then ' 用户单击清屏按钮时
03 PicShow.Cls ' 将图片框中显示的图形清除
04 End If
05 If Index = 1 Then ' 用户单击退出按钮时
06 End ' 退出程序
07 End If
08 End Sub
```

(3) 在 Form1 窗体中双击名称为 CmdPicture 的按钮控件，进入代码窗体，输入以下代码（代码 9-9-2.txt）。

```
01 Private Sub CmdPicture_Click(Index As Integer) ' 命令按钮组 CmdPicture 的鼠标单击事件代码
02 If Index = 0 Then ' 用户单击绘制点按钮时
03 IntA = 0 ' 变量赋值
04 Form2.TxtPoint2(0).Visible = False ' 在 Form2 中显示相关的参数输入文本框
05 Form2.TxtPoint2(1).Visible = False
06 Form2.TxtRad.Visible = False
```

```

07 Form2.TxtAng1.Visible = False
08 Form2.TxtAng2.Visible = False
09 Form2.TxtScale.Visible = False
10 End If
11 If Index = 1 Then      ' 用户单击绘制直线按钮时
12     IntA = 1          ' 变量赋值
13     Form2.TxtPoint2(0).Visible = True ' 在 Form2 中显示相关的参数输入文本框
14     Form2.TxtPoint2(1).Visible = True
15     Form2.TxtRad.Visible = False
16     Form2.TxtAng1.Visible = False
17     Form2.TxtAng2.Visible = False
18     Form2.TxtScale.Visible = False
19 End If
20 If Index = 2 Then      ' 用户单击绘制矩形按钮时
21     IntA = 2          ' 变量赋值
22     Form2.TxtPoint2(0).Visible = True ' 在 Form2 中显示相关的参数输入文本框
23     Form2.TxtPoint2(1).Visible = True
24     Form2.TxtRad.Visible = False
25     Form2.TxtAng1.Visible = False
26     Form2.TxtAng2.Visible = False
27     Form2.TxtScale.Visible = False
28 End If
29 If Index = 3 Then      ' 用户单击绘制圆按钮时
30     IntA = 3          ' 变量赋值
31     Form2.TxtPoint2(0).Visible = False ' 在 Form2 中显示相关的参数输入文本框
32     Form2.TxtPoint2(1).Visible = False
33     Form2.TxtRad.Visible = True
34     Form2.TxtAng1.Visible = False
35     Form2.TxtAng2.Visible = False
36     Form2.TxtScale.Visible = False
37 End If
38 If Index = 4 Then      ' 用户单击绘制椭圆按钮时
39     IntA = 4          ' 变量赋值
40     Form2.TxtPoint2(0).Visible = False ' 在 Form2 中显示相关的参数输入文本框
41     Form2.TxtPoint2(1).Visible = False
42     Form2.TxtRad.Visible = True
43     Form2.TxtAng1.Visible = False
44     Form2.TxtAng2.Visible = False
45     Form2.TxtScale.Visible = True
46 End If
47 If Index = 5 Then      ' 用户单击绘制圆弧按钮时
48     IntA = 5          ' 变量赋值
49     Form2.TxtPoint2(0).Visible = False ' 在 Form2 中显示相关的参数输入文本框

```

```

50 Form2.TxtPoint2(1).Visible = False
51 Form2.TxtRad.Visible = True
52 Form2.TxtAng1.Visible = True
53 Form2.TxtAng2.Visible = True
54 Form2.TxtScale.Visible = False
55 End If
56 Form2.Show ' 显示 Form2 窗口
57 End Sub

```

(4) 在 Form2 中没有控件的任意空白处右击, 在弹出的快捷菜单中选择【查看代码】选项, 在弹出的代码窗口中定义公有变量 (代码 9-9-3.txt)。

```

Public SinA As Variant
Public SinB As Variant
Public SinC As Variant
Public SinD As Variant
Public SinE As Variant
Public SinF As Variant
Public SinG As Variant
Public SinH As Variant
Public SinI As Variant
Const PI = 3.1415926 ' 以上定义程序中需要的变量

```

(5) 在 Form2 窗体中双击名称为 Cmd2 的按钮控件, 进入代码窗体, 输入以下代码 (代码 9-9-4.txt)。

```

01 Private Sub Cmd2_Click(Index As Integer) ' 命令按钮组 Cmd1 的鼠标单击事件代码
02 If Index = 0 Then ' 用户单击确定按钮时
03 SinA = Val(TxtPoint1(0)): SinB = Val(TxtPoint1(1)) ' 将不同文本框输入的量赋值给变量
04 SinC = Val(TxtPoint2(0)): SinD = Val(TxtPoint2(1))
05 SinE = Val(TxtColor)
06 SinF = Val(TxtRad.Text)
07 SinG = Val(TxtAng1.Text)
08 SinH = Val(TxtAng2.Text)
09 SinI = Val(TxtScale.Text)
10 Form2.Hide ' 隐藏 Form2 窗口
11 Form1.Show ' 显示 Form1 窗口
12 If TxtColor.Text = "" Then ' 当颜色文本框中无输入时
13 Print "请输入颜色值" ' 显示警告信息
14 End If
15 If Form1.IntA = 0 Then ' 当用户选择绘制点命令按钮时
16 Form1.PicShow.PSet (SinA, SinB), SinE ' 绘制点
17 End If
18 End If

```

```

19 If Form1.IntA = 1 Then '当用户选择绘制直线命令按钮时
20     Form1.PicShow.Line (SinA, SinB)-(SinC, SinD), SinE '绘制直线
21 End If
22 If Form1.IntA = 2 Then '当用户选择绘制矩形命令按钮时
23     Form1.PicShow.Line (SinA, SinB)-(SinC, SinD), SinE, B '绘制矩形
24 End If
25 If Form1.IntA = 3 Then '当用户选择绘制圆命令按钮时
26     Form1.PicShow.Circle (SinA, SinB), SinF, SinE '绘制圆
27 End If
28 If Form1.IntA = 4 Then '当用户选择绘制椭圆命令按钮时
29     Form1.PicShow.Circle (SinA, SinB), SinF, SinE, , SinI '绘制椭圆
30 End If
31 If Form1.IntA = 5 Then '当用户选择绘制圆弧命令按钮时
32     If SinG >= 0 And SinG <= 2 * PI And SinH >= 0 And SinH <= 2 * PI Then
33         '当 SinG 和 SinH 变量值大于等于0 小于等于 2PI
34         Form1.PicShow.Circle (SinA, SinB), SinF, SinE, SinG, SinH '绘制圆弧
35     ElseIf SinG < 0 Then '当 SinG 小于 0
36         Print "请输入大于0 小于 2PI 的值" '显示警告信息
37     ElseIf SinG > 2 * PI Then '当 SinG 大于 2PI
38         Print "请输入大于0 小于 2PI 的值" '显示警告信息
39     ElseIf SinH < 0 Then '当 SinH 小于 0
40         Print "请输入大于0 小于 2PI 的值" '显示警告信息
41     ElseIf SinH > 2 * PI Then '当 SinH 大于 2PI
42         Print "请输入大于0 小于 2PI 的值" '显示警告信息
43     End If
44 End If
45 If Index = 1 Then '用户单击取消按钮时
46     Form2.Hide '隐藏 Form2 窗口
47 End If

```

## 【运行结果】

选择【工程】>【工程属性】菜单命令，在【工程属性】对话框中设置 Form1 为启动窗体。保存程序，并按快捷键【F5】运行程序。单击相应的命令按钮，输入相应的参数，单击【确定】按钮，图形即可显示在图像框控件中。






这是一个功能比较丰富的程序，也非常有趣，大家可以试着修改一下源代码，看看程序中各个部分的代码到底有什么作用。

**提示**

## 【拓展训练 9-8】

在图像框中加载现有的各种类型图片，以减少代码的编写难度和编写量。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 窗体上加载一个 PictureBox 控件。

(3) 选中图像框控件，在属性窗口中，单击 Picture 属性后面的扩展按钮 ，打开加载图片对话框，从中选择一幅合适的图片，然后单击【打开】按钮即可。



## 9.11 滚动条控件



本节视频教学录像：8 分钟

当一个屏幕显示不下过多的内容时，滚动条是一种很好的解决方式，在浏览网页和编辑文档等很多地方都要使用滚动条来查看更多的内容。在 Visual Basic 中，滚动条分为横向滚动条（HScrollBar）和纵向滚动条（VScrollBar）两种，它们除了在外观形状上有所不同之外，其操作形式是相同的，即它们具有相同的属性、事件和方法。

### 9.11.1 滚动条控件的主要属性

#### 1. 最大值属性（Max）与最小值属性（Min）

Max 和 Min 用于返回滚动条的最大值和最小值。用户可以通过属性窗口中的 Max 和 Mix 属性输入框进行设置。



Min 属性值必须总是等于或大于 0，同时，如果 Max 值设置的比 Min 值小的话，最大值将会自动设置为水平条的最左边的位置处，或被自动设置为垂直滚动条的最上位置处。

**提示**

#### 2. 数值属性（Value）

Value 属性用来返回或设置滚动滑块在当前滚动条中位置的值。用户可以通过属性窗口中的

【Value】文本框进行设置。

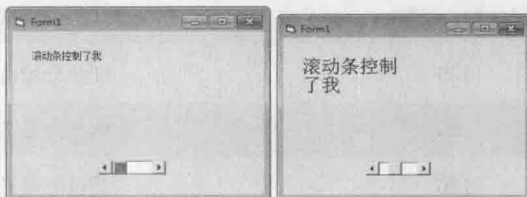
### 3. 小改变属性 (SmallChange)

该属性用于设置和返回用户单击滚动条左右边上的箭头时，滚动条 Value 值的改变量的大小，系统默认为 1。可以通过属性窗口中的 SmallChange 属性输入框进行设置。

例如在窗体中添加一个 Label 控件，设置 Caption 属性为“滚动条控制了我”，再添加一个 HScrollBar 控件。双击 HScrollBar 控件，输入以下代码。

```
01 Private Sub HScroll1_Change()  
02 Label1.FontSize = HScroll1.Value ' 拖动滚动条改变字体大小  
03 End Sub
```

效果如图所示。



### 4. 大改变属性 (LargeChange)

该属性用于设置和返回用户单击滚动条左右边上的箭头时，滚动条 Value 值的改变量的大小，系统默认为 1。用户可以通过属性窗口中的 LargeChange 属性输入框进行设置。

## 9.11.2 滚动条控件的主要事件

滚动条控件相关的事件主要是滚动 (Scroll) 与改变 (Change)。当在滚动条内拖动滚动框时会触发 Scroll 事件，滚动框的位置发生改变时会触发 Change 事件。

语法格式如下。

```
Private Sub DataGrid_Scroll([cancel As Integer])
```

或：

```
Private Sub object_Scroll( )
```

## 9.11.3 滚动条应用示例

单击滚动条的两端或直接拖动滚动条，可以在画板中调制 RGB 颜色中的任意一种。

**【范例 9-10】使用水平滚动条来控制标签控件的前景色，从而使标签控件的前景色能够成为 RGB 颜色中的任意一种。**

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打



开】按钮。

- (2) 在窗体中创建 1 个标签 (Label) 控件, 将名称属性设置为 “LblShow”, “Caption” 属性设置为空。  
(3) 在 Form1 中添加 1 个包含 3 个标签控件的标签控件组, 各标签的属性设置如下表所示。

控件名称	Caption 属性	控件作用
Lbl(0)	红	提示用户选择红色分量
Lbl(1)	绿	提示用户选择绿色分量
Lbl(2)	蓝	提示用户选择蓝色分量

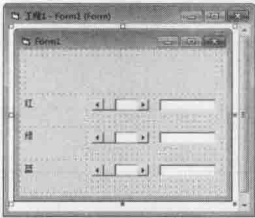
- (4) 在 Form1 中添加 1 个包含 3 个水平滚动条的控件组, 各水平滚动条的属性设置如下表所示。

控件名称	Value 属性	控件作用
ScrColor(0)	125	红色分量选择滚动条
ScrColor(1)	125	绿色分量选择滚动条
ScrColor(2)	125	蓝色分量选择滚动条

- (5) 在 Form1 中再添加 1 个包含 3 个文本的文本框控件组, 各标签的属性设置如下表所示。

控件名称	Caption	BorderStyle	控件作用
TxtCorlor(0)	“ ”	1	显示用户选择的红色分量值
TxtCorlor(1)	“ ”	1	显示用户选择的绿色分量值
TxtCorlor(2)	“ ”	1	显示用户选择的蓝色分量值

- (6) 将上面创建的这些控件按照下图所示排列好。



- (7) 在 Form1 窗体中没有控件的任意空白处右击, 在弹出的快捷菜单中选择【查看代码】选项, 在弹出的代码窗口中输入以下代码 (代码 9-10-1.txt)。

```
01 Private Sub Form_Load()  
02   ScrColor(0).Min = 0: ScrColor(1).Min = 0: ScrColor(2).Min = 0 ' 设置滚动条最小值  
03   ScrColor(0).Max = 255: ScrColor(1).Max = 255: ScrColor(2).Max = 255 ' 设置滚动条最大值
```

```

04 LblCorlor(0).Caption = CStr(ScrColor(0).Value)
    ' 将滚动条的 Value 值赋值给标签 LblCorlor(0) 的 Caption 属性
05 LblCorlor(1).Caption = CStr(ScrColor(1).Value)
    ' 将滚动条的 Value 值赋值给标签 LblCorlor(1) 的 Caption 属性
06 LblCorlor(2).Caption = CStr(ScrColor(2).Value)
    ' 将滚动条的 Value 值赋值给标签 LblCorlor(2) 的 Caption 属性
07 LblShow.BackColor = RGB(ScrColor(0).Value, ScrColor(1).Value, ScrColor(2).Value)
    ' 将标签控件的背景色设置为滚动条确定的颜色值
08 End Sub

```

(8) 在 Form1 窗体中双击名称为 ScrColor 的控件，在弹出的代码窗口中输入以下代码（代码 9-10-2.txt）。

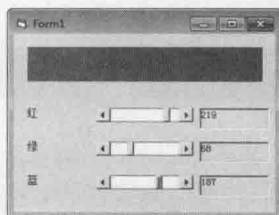
```

01 Private Sub ScrColor_Change(Index As Integer)
02     If Index = 0 Then          ' 当水平滚动条 ScrColor(0) 位置改变时
03         LblCorlor(0).Caption = CStr(ScrColor(0).Value)
            ' 将滚动条的 Value 值赋值给标签控件
04         LblShow.BackColor = RGB(ScrColor(0).Value, ScrColor(1).Value, ScrColor(2).Value)
            ' 将标签控件的背景色设置为滚动条确定的颜色值
05     End If
06     If Index = 1 Then          ' 当水平滚动条 ScrColor(1) 位置改变时
07         LblCorlor(1).Caption = CStr(ScrColor(1).Value)
            ' 将滚动条的 Value 值赋值给标签控件
08         LblShow.BackColor = RGB(ScrColor(0).Value, ScrColor(1).Value, ScrColor(2).Value)
            ' 将标签控件的背景色设置为滚动条确定的颜色值
09     End If
10     If Index = 2 Then          ' 当水平滚动条 ScrColor(2) 位置改变时
11         LblCorlor(2).Caption = CStr(ScrColor(2).Value)
            ' 将滚动条的 Value 值赋值给标签控件
12         LblShow.BackColor = RGB(ScrColor(0).Value, ScrColor(1).Value, ScrColor(2).Value)
            ' 将标签控件的背景色设置为滚动条确定的颜色值
13     End If
14 End Sub

```

## 【运行结果】

保存程序，并按快捷键【F5】运行程序，拖动滚动条上的滑块即可更改标签中的颜色显示。



## 9.12 程序中的闹钟——定时器控件



本节视频教学录像: 7 分钟

我们编写程序的时候,有时需要定时执行某些功能,比如每天定时备份数据、每隔两个小时整理一次数据等,这时可以使用 Visual Basic 中提供的定时器控件来完成这些任务。

### 9.12.1 定时器控件的主要属性


定时器控件 (Timer) 最重要的一个属性就是时间间隔属性 (Interval)。时间间隔属性以 ms 为单位,取值范围为 0 ~ 65535。用户可以通过属性窗口中的【Interval】文本框进行设置。

### 9.12.2 定时器控件的主要事件

定时器控件只有一个事件,即 Timer 事件。定时器控件经过预设的时间间隔,将激发定时器的 Timer 事件。

### 9.12.3 定时器控件应用示例

**【范例 9-11】用定时器控件制作一个电子表,在这个电子表中可以显示当前的日期、具体的时间等。**

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标 ,然后单击【打开】按钮。

(2) 在 Form1 中添加 1 个图像框 (PictureBox) 控件,将该控件的名称属性设置为“PicTime”。

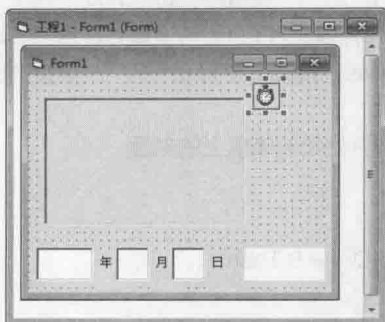
(3) 在 Form1 中添加 3 个文本框 (Text Box) 控件,各文本框的属性设置如下表所示。

名称	Caption	控件作用
TxtYear	“ ”	显示当前日期的年份值
TxtMth	“ ”	显示当前日期的月份值
TxtDay	“ ”	显示当前日期为几号

(4) 在 Form1 中添加 4 个标签 (Label) 控件,各标签的属性设置如下表所示。

名称	Caption	BorderStyle	控件作用
Lbl1	年	0	显示前面的年份值
Lbl2	月	0	显示前面的月份值
Lbl3	日	0	显示前面的日期值
LblTime	“ ”	1	显示当前时间

(5) 在 Form1 中添加 1 个定时器控件。将 Form1 中的各个控件拖动到合适的位置, 效果如图所示。



(6) 在 Form1 窗体中没有控件的任意空白处右击, 在弹出的快捷菜单中选择【查看代码】选项, 在弹出的代码窗口中添加公有变量。

```
01 Dim day, year, month, ddate, ttime As String ' 定义时间变量
02 Dim alf(0 To 11) ' 定义数组
03 Dim rr ' 定义变量
04 Dim nHourLen, nMinLen, nSecLen As Integer ' 定义时针、分针、秒针长度变量
05 Const Pi = 3.1415926 ' 定义 Pi 常量
```

(7) 在 Form1 窗体的 Load 事件中输入以下代码。

```
01 Private Sub Form_Load()
02 Timer1.Interval = 1000 ' 设置计时器每隔 1 秒变换一次
03 End Sub
```

(8) 双击 Timer 控件, 进入代码窗口, 输入以下代码 (代码 9-11-1.txt)。

```
01 Private Sub Timer1_Timer()
02 rr = PicTime.Height / 2
03 ddate = Format(Now, "mm:dd:yy") ' 格式化日期变量
04 ttime = Format(Now, "hh:mm:ss") ' 格式化时间变量
05 month = Left(ddate, 2) ' 读取月份值
06 day = Mid(ddate, 4, 2) ' 读取天
07 year = Right(ddate, 2) ' 读取年份值
08 hh = Left(ttime, 2) ' 读取时针值
09 mm = Mid(ttime, 4, 2) ' 读取分针值
10 ss = Right(ttime, 2) ' 读取秒针值
11 TxtYear.Text = "20" & year ' 显示当前年份
12 TxtMth.Text = month ' 显示当前月份
13 TxtDay.Text = day ' 显示当前几号
14 LblTime.Caption = Time ' 显示当前事件值
15 nWidth = PicTime.Width - 40 ' 设置表盘半径
16 nHourLen = nWidth * 4 / 18 ' 设置时针半径
17 nMinLen = nWidth * 6 / 18 ' 设置分针半径
```

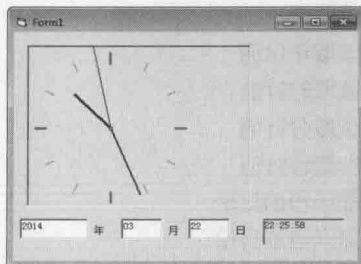
```

18 nSecLen = nWidth * 8 / 18      ' 设置秒针半径
19 alfsec = ((ss - 15) / 30) * Pi  ' 设置秒针每次转动的角度
20 alfmin = ((mm + ss / 60 - 15) / 30) * Pi  ' 设置分针每次转动的角度
21 alhour = ((hh + mm / 60 + ss / 3600 - 15) / 6) * Pi  ' 设置时针每次转动的角度
22 PicTime.Refresh ' 刷新图片
23 For l = 0 To 11 ' 利用循环开始绘制表盘上的刻度
24     alf(l) = l * 30 * Pi / 180
25     PicTime.DrawWidth = 1
26     If l = 0 Or l = 3 Or l = 6 Or l = 9 Then
27         PicTime.DrawWidth = 3
28     End If
29     PicTime.Line (rr + (rr - 100) * Cos(alf(l)), rr + (rr - 100) *
30     * Sin(alf(l)))-(rr + (rr - 300) * Cos(alf(l)), rr + (rr - 300) *
31     * Sin(alf(l))), RGB(255, 0, 255)
32 Next l
33 PicTime.DrawWidth = 3      ' 设置时针宽度
34 PicTime.Line (rr, rr)-(rr + nHourLen * Cos(alhour), rr + nHourLen * Sin(alhour))
    ' 绘制时针
35 PicTime.DrawWidth = 2      ' 设置分针宽度
36 PicTime.Line (rr, rr)-(rr + nMinLen * Cos(alfmin), rr + nMinLen * Sin(alfmin))
    ' 绘制分针
37 PicTime.DrawWidth = 1      ' 设置秒针宽度
38 PicTime.Line (rr, rr)-(rr + nSecLen * Cos(alfsec), rr + nSecLen * Sin(alfsec))
    ' 绘制秒针
39 PicTime.DrawWidth = 5      ' 设置中心点的大小
40 PicTime.PSet (rr, rr), RGB(255, 0, 255) ' 绘制中心点
41 End Sub

```


## 【运行结果】

保存程序，并按快捷键【F5】运行程序，效果如图所示。

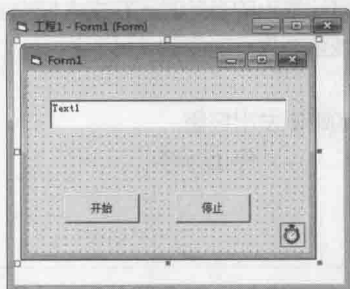


## 【拓展训练 9-9】

输入一个分钟时间值，将其转换为秒，然后进行倒计时显示。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在 Form1 窗体中添加两个命令按钮控件、1 个文本框控件和 1 个定时器控件, 将 Command1 命令按钮控件的 Caption 属性设置为“开始”, 将 Command2 命令按钮控件的 Caption 属性设置为“停止”, 最终效果如图所示。



(3) 在 Form1 窗体中没有控件的任意空白处右击, 在弹出的快捷菜单中选择【查看代码】选项, 在弹出的代码窗口中添加公有变量。

```
Option Explicit
Dim M As Single, S As Long ' 定义公有变量
```

(4) 在 Form1 窗体的 Load 事件中输入以下代码。

```
01 Private Sub Form_Load()
02     Timer1.Interval = 1000 ' 设置每隔 1 秒时间变一次
03     Timer1.Enabled = False ' 初始化时定时器控件不进行操作
04     Text1.Text = "清空文本文字并输入时间, 单击开始按钮开始倒计时" ' 初始化文本框中的内容
05 End Sub
```

(5) 在 Form1 窗体中双击【开始】按钮, 进入代码窗口输入以下代码 (拓展代码 9-9-1.txt)。

```
01 Private Sub command1_Click()
02     If Text1.Text = "" Then Exit Sub ' 如果时间为空, 退出程序
03     M = CSng(Val(Text1.Text)) ' 获取将要倒计时的时间值
04     If M <= 0 Then Text1.Text = "": Exit Sub ' 如果时间小于等于 0, 则退出
05     S = M * 60 ' 将分钟计算为秒
06     If S = 0 Then Text1.Text = "": Exit Sub ' 再次判断计算成秒后是否为 0, 为 0 退出
07     Text1.Text = Str(S) ' 文本框中的文本内容显示为秒
08     Timer1.Enabled = True ' 定时器控件开始倒计时
09 End Sub
```

(6) 在 Form1 窗体中双击【停止】按钮, 进入代码窗口输入以下代码。

```
01 Private Sub command2_Click()
02     End ' 退出程序
```

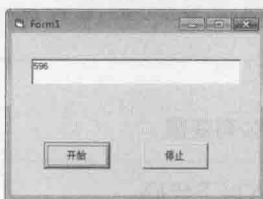


## 03 End Sub

(7) 在 Form1 窗体中双击定时器控件, 进入代码窗口输入以下代码。

```
01 Private Sub Timer1_Timer()
02     S = S - 1      ' 让秒减 1
03     Text1.Text = Trim(Str(S)) ' 将计算出的当前时间在文本框中显示出来
04     If S = 0 Then  ' 如果时间减至 0 提示时间到
05         Print " 时间到!!! "
06         Beep      ' 让计算机的喇叭发出报警
07         Timer1.Enabled = False    ' 停止计时
08     End If
09 End Sub
```

(8) 保存程序, 并按快捷键【F5】运行程序。清空文本文字, 输入数字 10, 单击【开始】按钮。




## 9.13 文件系统控件



本节视频教学录像: 7 分钟

文件系统是为了存储和管理数据及文件以便于查找和访问的组织方法。在我们使用软件的过程中, 经常会遇到需要访问计算机所存储的文件的情况, 如打开一个文件、保存一个文件等。文件系统控件即可满足这些需要。Visual Basic 中提供有 3 种文件系统控件, 分别是驱动器列表框 (DriveListBox) 控件、目录列表框 (DirListBox) 控件和文件列表框 (FileListBox) 控件。


### 9.13.1 驱动器列表框控件

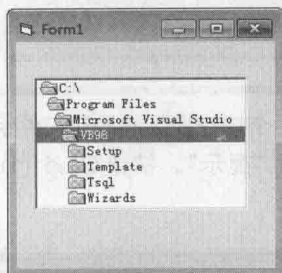
驱动器列表框控件  用来显示用户系统中所有有效磁盘驱动器的列表, 通过它可以从任一可用驱动器的磁盘文件列表中打开文件。



驱动器列表框控件中的 Drive 是比较常用的属性，该属性用于返回或设置运行时选择的驱动器，默认返回当前驱动器。


### 9.13.2 目录列表框控件

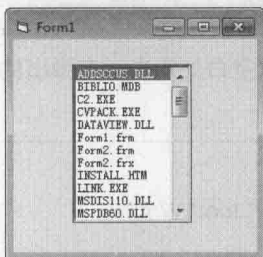
目录列表框控件  用来显示和设置文件夹的路径。



目录列表框控件比较常用的属性有 Path 属性和 ListIndex 属性等。其中，Path 属性的功能是返回或设置运行时选择的路径，默认路径为当前路径。ListIndex 属性的功能是返回或设置控件中当前被选择的项目索引号。目录列表框中的每一个目录都可以通过 ListIndex 属性来标识。

### 9.13.3 文件列表框控件

文件列表框控件  用来列出当前目录中的部分或者全部文件。



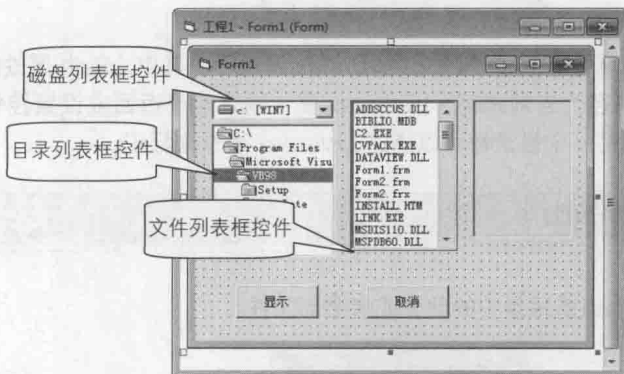
### 9.13.4 文件系统应用示例

**【范例 9-12】** 将驱动器列表框控件、目录列表框控件和文件列表框控件联合使用，设计一个图片文件浏览器。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在 Form1 窗体中添加 1 个图像框 (PictureBox) 控件，设置该控件的名称属性为“PicShow”。
- (3) 在 Form1 窗体中分别添加 1 个磁盘列表框 (DriveListBox) 控件、目录列表框 (DirListBox) 控件以及文件列表框 (FileListBox) 控件。



(4) 在 Form1 窗体中添加 1 个包含两个按钮控件的按钮控件组。按钮控件组的名称属性为“Cmd1”。将按钮 Cmd1(0) 的 Caption 属性设置为“显示”，按钮 Cmd1(1) 的 Caption 属性设置为“取消”。将各个控件按下图所示排列。



(5) 在 Form1 窗体中没有控件的任意空白处双击，在弹出的代码窗口中为 Form1 窗体的 Load 事件输入以下代码。

```
01 Private Sub Form_Load()  
02   File1.Pattern = "*.jpg;*.gif;*.bmp;*.ico;*.wmf" ' 只显示指定类型的图形文件  
03 End Sub
```

(6) 在 Form1 窗体中双击名称为 Cmd1 的命令按钮，进入代码窗口，输入以下代码 (代码 9-12-1.txt)。

```
01 Private Sub Cmd1_Click(Index As Integer)  
02   If Index = 0 Then ' 用户单击显示图像按钮时  
03     PicShow.Cls ' 将图像框中显示的图像清除  
04     PicShow.Picture = LoadPicture(Dir1.Path & "\ " & File1.FileName) ' 在图像框中显示选中的文件  
05   End If  
06   If Index = 1 Then ' 用户单击退出按钮时  
07     End ' 退出程序  
08   End If  
09 End Sub
```

(7) 在 Form1 窗体中双击名称为 Dirve1 的命令按钮，进入代码窗口输入以下代码。

```
01 Private Sub Drive1_Change()  
02 Dir1.Path = Drive1.Drive '将驱动器列表中选中的当前驱动器赋给目录列表路径  
03 End Sub
```

(8) 在 Form1 窗体中双击名称为 Dir1 的命令按钮，进入代码窗口输入以下代码。

```
01 Private Sub Dir1_Change()  
02 File1.Path = Dir1.Path '将文件列表框的路径值设置为目录列表框选中路径值  
03 End Sub
```

## 【运行结果】

保存程序，并按快捷键【F5】运行程序。选择所需的文件路径，并单击【显示】按钮，可以查看文件效果。



## 9.14 控件数组



本节视频教学录像：4 分钟

在应用程序中，往往要使用一些类型相同、功能相似的控件，可以将这种同一类型的控件定义成一个控件数组。当界面上需要若干个控件执行大致相同的操作时，控件数组很有用。

### 9.14.1 控件数组的概念

控件数组由一组相同类型的控件组成，它们共有一个相同的控件名称，具有相同的属性设置，共享同样的事件过程。

特点：

(1) 这些控件由控件数组的下标来区别，而下标由每个控件的唯一标识索引号 (Index) 来标识，通过 Index 属性即可区别控件数组元素，Index 从 0 开始取值，如：控件数组 Text1(2) 表示控件数组 Text1 的第 3 个元素。

(2) 同一个控件数组的所有元素, 具有相同的 Name 属性值, 但对于其他属性则没有规定必须完全相同。

① 控件数组的所有元素, 共享同样的事件过程, 故控件数组适用于若个控件执行操作相似的场合。

② 为区分触发某个事件是由哪个控件触发的, VB 会把该控件的下标索引号 (Index) 传递给过程。

如: 设命令按钮控件数组名为 Command, 其单击事件过程为:

```
01 Sub Command_Click(Index as Integer)
02 ...
03 End Sub
```

经过参数传递判断 Index 的值, 便知道用户按下哪一个按钮, 也就知道可以对该按钮进行相应编程。

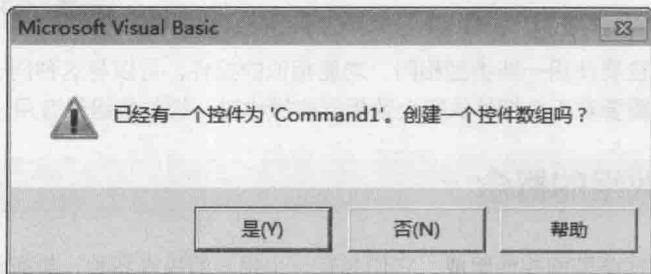
```
01 Sub Command_Click(Index as Integer)
02 ...
03 if index=4 then
04     label1.caption= " 第五个按钮被点击 "
05 end if
06 ...
07 End Sub
```

### 9.14.2 控件数组的创建

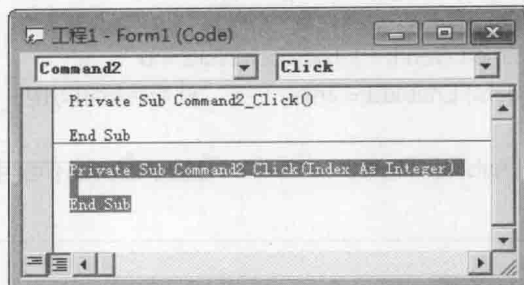
通常, 建立控件数组有两种途径: 在设计时建立和运行时添加。下面就这两种途径来说明其建立步骤。在进行窗体设计时, 建立控件数组的步骤如下。

(1) 拖动控件到窗体上并进行属性设置。

(2) 选中该控件进行“复制”和“粘贴”操作, 系统提示“是否建立控件数组”, 选择“是”即可。多次粘贴就可以创建多个控件元素。如下图所示是对一个 CommandButton 命令按钮控件进行“复制”和“粘贴”操作得到的提示。



(3) 进行事件过程的编程。同样以上图为例, 当用户单击“是”按钮, 选择建立控件数组, 和单击“否”按钮选择不创建控件数组, 其事件代码框是不一样的。如下图所示, 如果 Command2 是控件数组中的一个元素, 其单击事件后带有参数 Index, 否则不带任何参数。



控件数组除了可在窗体设计时建立，也可在程序运行时建立。其在运行时的建立同样只需在窗体上画出某空间即可，设置该空间的 Index 值为 0，表示该控件为数组。单个控件的 Index 值如下面左图所示，可以看到，其值为空；控件数组中的元素如下面右图所示，可以看到，其 Index 值为 0，即控件数组的第 1 个元素。



在编程中通过 Load 方法添加其余若干个元素，也可以通过 Unload 方法删除某个添加的元素。例如，在运行中添加一个 Label 控件，实现代码为 Load Label1 (2)，添加后，在窗体上看不到该创建的控件，需要设置其 Visible 属性为 True，并设置相应的位置后才可见。同样地，删除该控件，只需使用 Unload Label1 (2) 语句即可。

### 9.14.3 控件数组的使用

控件数组有在设计时设置好的，也有在运行中创建的。控件数组一方面使得程序简洁、令代码易于维护，另一方面能使程序具有灵活性。可见，科学地使用控件数组可使编程工作的效率提高。

#### 1. 运行中设置控件数组的属性

设窗体上有若干个以 Command1 命名的命令按钮，现要求：点击其中一个按钮后，该按钮不可用，而其他的按钮均可用。以下几行代码可以实现这个要求，比一个一个地设置高效得多。

```
01 Private Sub Command1_Click(Index As Integer)
02     Dim i As Integer          ' 计数器
03     Dim comNum As Integer     ' 按钮的索引号
04     comNum = 0
05     For i = 0 To Command1.Count - 1
```



```

06    comNum = comNum + 1
07    If comNum > Command1.Count - 1 Then comNum = 0
08    Command1(comNum).Enabled = True    ' 让所有按钮可用
09    Next
10    Command1(Index).Enabled = False    ' 让被单击按钮不可用
11 End Sub

```

## 2. 运行中添加和卸载控件数组

窗体上已有一个文本框 Text1, 程序需要在运行时动态地创建若干个文本框, 可这样实现:

(1) 首先, 设计时给 Text1 的 Index 属性设置为 "0", 这一步很重要——有了索引号才能创建控件数组。

(2) 编写代码: 之前请给工程添加两个命令按钮, Name 属性取缺省值, Caption 属性分别为: 添加、卸载。

```

01 Private Sub Command1_Click()
02     Dim txtNum As Integer    ' Text1 的 Index 号
03     Dim Num As Integer      ' 赋给各 TextBox 的值
04     txtNum = 0              ' 初值
05     Num = 1 ' 初值
06     Text1(0).Text = "Text" & Num    ' 第一个 Text1 的值
07     Dim i As Integer        ' 计数器
08     For i = 0 To 4          ' 添加五个 TextBox
09         txtNum = txtNum + 1
10         Num = Num + 1
11         Load Text1(txtNum)    ' 加载文本框
12         Text1(txtNum).Top = Text1(txtNum - 1).Top + 450 ' 设置位置
13         Text1(txtNum).Text = "Text" & Num    ' 加载内容
14         Text1(txtNum).Visible = True    ' 令其可见不能漏
15     Next
16     Command1.Enabled = False
17     Command2.Enabled = True
18 End Sub
19 Private Sub Command2_Click()
20     Dim i As Integer, N As Integer
21     N = 0
22     For i = 1 To Text1.Count - 1
23         N = N + 1
24         Unload Text1(N)
25     Next
26     Command1.Enabled = True
27     Command2.Enabled = False
28 End Sub

```

## 9.15 高手点拨



本节视频教学录像：9 分钟

用户在学习 Visual Basic 的标准控件时应注意以下问题。

### 1. 遗漏对象名称

在 VB 程序设计时，初学者常犯的一个错误是遗漏对象的名称，特别是在使用列表框时。例如，如果要引用列表框（List1）中当前选定的项目，List1.List(ListIndex) 是错误的。即使当前焦点在 List1 上，VB 也不认为 ListIndex 是 List1 的属性，而是一个变量。所以正确的引用方式是：List1.List(list1.ListIndex)。

### 2. 列表框的 Columns 属性

列表框的 Columns 属性决定列表框是水平滚动还是垂直滚动以及如何显示列中的项目。如果水平滚动，则 Columns 属性决定显示多少列，下表是一个水平滚动两列显示的列表框。

列数	属性
0	项目安排在一列中，且列表框竖直滚动
1-n	项目安排在多个列中，先填第一列，再填第二列……列表框水平滚动且显示指定数目的列

在程序运行期间，该属性是只读的，也就是说，在程序运行时不能将多列列表框变为单列列表框或将单列列表框变为多列列表框。

### 3. Form\_Load 事件内无法用绘图方法在窗体或 PictureBox 控件上输出图形

VB 忽略在不可见对象上使用图形的方法。在 Form\_Load 事件过程中窗体是不可见的，当窗体为 AutoRedraw 属性设置为 False 时，Form\_Load 事件中图形方法在窗体上不产生输出。同样，在 Form\_Load 事件过程中 PictureBox 控件也不可见，如果用图形方法向 PictureBox 控件输出也将被忽略。有两种方法可解决此问题。

方法一，将绘图程序代码放在其他事件内。通常在 Paint 事件中完成绘图，当对象在显示、位移、改变大小和使用 Refresh 方法时，都会发生 Paint 事件。

方法二，将对象的 AutoRedraw 属性设置为 True 时，在该对象上任何以图形方法绘制的图形都将在内存中建立一个备份，在改变对象大小或重新显示的情况下，将自动从备份中调出图形产生重画过程。例如，要在 Form\_Load 事件中使用绘图方法在 PictureBox 控件上输出图形，则应设置 PictureBox.AutoRedraw 为 True，而不是窗体的 AutoRedraw 为 True。当窗体的 Form\_Load 事件完成后，PictureBox 将产生重画过程，从备份中调出图形。

### 4. VB 坐标系中旋转正向

在 VB 坐标系中，逆时针方面为正，各绘图方法都参照此坐标系。计算对象的坐标点时务必注意这一点。

### 5. 如何清除已绘制的线条

Line 控件在窗体上移动时，原位置上不会留下图形痕迹。如果用 Line 方法来代替 Line 控件，则每

次在新位置上画直线前,需要清除原来位置上的线条。清除原来位置上的线条,可将 DrawMode 属性设置为 Xor 模式,在原位置上重画一次直线,即可清除原来的线条。

#### 6. 如何判定对象是否越出窗体的边界

当对象在窗体上移动时,对象是否越出窗体的上边界或左边界,不能用对象的 Top<0 或对象的 Left<0 来判断,对象的 Top<0 (Left<0) 仅表示该控件对象的上(左)边界越出窗体的上(左)边界,而要使整个控件越出窗体的上(左)边界,还需要加上控件的高度(宽度)。

#### 7. 位块传送产生“无效的过程调用”错误的原因

PaintPicture 方法中位块传送的最小宽(高)度为 8Twip 单位,如果所取区域小于下限数,将产生“无效的过程调用”错误。可在程序中加入“On Error Resume Next”错误处理语句,忽略错误。PaintPicture 方法中的操作模式只能用于位图类型图像。当传递其他图像类型时将一个值给该参数会造成“无效的过程调用或参数”错误。这是设计的原因。要避免这个错误,对于除位图外的图像,将操作模式参数置为空。

#### 8. PaintPicture 方法处理动画时无动感

如果使用 PaintPicture 方法处理动画,由于处理时间较长,可能看不到动感效果。为了能形成动感,需要在使用该方法后立即调用 DoEvents 方法,将控制权交给 Windows,再返回程序,此时可刷新图形对象内的画面。

## 9.16 实战练习

### 一、思考题

1. 试简述列表框中 list.list 属性、list.listindex 属性和 list.index 属性的区别。
2. Visual Basic 中提供了几种文件系统控件,分别是什么?

### 二、上机题

在窗体中添加一个滚动条控件、一个文本框控件和一个按钮控件,实现单击按钮控件后,根据文本框中输入的数值改变滚动条的位置。

# 第10章



本章视频教学录像：44 分钟

## 扩展你的需求 ——ActiveX 控件、工具栏和状态栏

在第 9 章中已介绍了 Visual Basic 6.0 标准控件的使用。除了标准控件，Visual Basic 6.0 还提供有大量的扩展控件，使用这些控件，可以更加丰富应用程序的功能。在 Visual Basic 6.0 中要使用扩展控件，首先要把控件添加到工具箱中。本章介绍如何在 Visual Basic 6.0 中使用扩展控件，并介绍几种常用的扩展控件的使用方法。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握添加扩展控件的方法
- ☐ 掌握图像列表控件
- ☐ 掌握工具栏控件
- ☐ 熟悉状态栏控件
- ☐ 熟悉树状视图控件
- ☐ 掌握选项卡控件
- ☐ 熟悉进度条控件

## 10.1 ActiveX 控件的使用



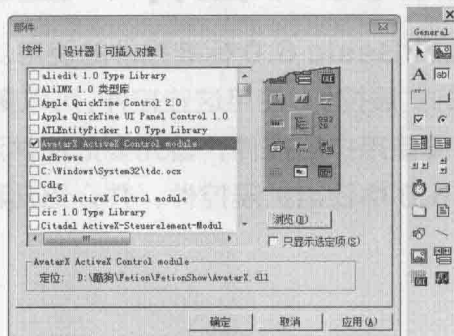
本节视频教学录像: 6 分钟

ActiveX 控件是有第三方开发或者用户自己开发的控件。事实上, ActiveX 控件只是在 Visual Basic 6.0 中使用的 ActiveX 部件之一。除此之外, Visual Basic 6.0 支持的 ActiveX 部件还有 ActiveX 文档、ActiveX DLL 和 ActiveX EXE 等。

### 10.1.1 ActiveX 控件的添加

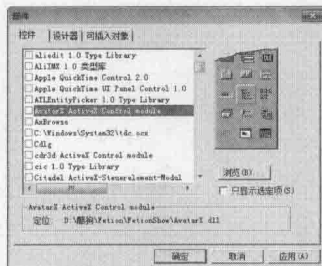
ActiveX 控件并不在 Visual Basic 6.0 的控件工具栏上, 那么在使用这些 ActiveX 控件之前, 需要先将它们添加进来。选择【工程】>【部件】命令, 打开如图所示对话框。

选中需要添加的 ActiveX 控件前的复选框, 再单击【应用】按钮, 将选中的 ActiveX 控件添加到了控件工具栏, 使用方法和标准控件是一样的。



### 10.1.2 ActiveX 控件的删除

删除 ActiveX 控件与添加操作类似。选择【工程】>【部件】命令, 打开如图所示对话框。找到要删除的 ActiveX 控件, 将其前面的复选框取消即可。



### 10.1.3 ActiveX 控件的注册

不同的 ActiveX 组件, 它们注册有着不同的方法, 现在就针对 ActiveX 控件、ActiveX DLL、ActiveX EXE 三种不同的组件分别给出解决方案。

## 1. ActiveX 控件的注册

ActiveX 控件与开发平台无关，在一种编程语言中开发出来的 ActiveX 控件，几乎不做任何修改，便可以在另一种编程语言中使用。但是 ActiveX 控件被开发出来以后，要想被正确使用，首先必须将控件文件复制到硬盘中，然后在 Windows 中进行注册。未在 Windows 中注册过的 ActiveX 控件是不能被使用的。一般注册 VB 6.0 ActiveX 控件的方法有如下几种。

(1) 使用 Regsvr32.exe 程序对 VB ActiveX 控件进行注册。该文件位于 Windows 目录的 system 子目录下。使用方法如下：点击“开始”，在弹出的菜单中再点击“运行”，在出现的对话框中输入以下命令。

< 控件路径和 ActiveX 控件文件名 > 注册一个 ActiveX 控件。此处 regsvr32 的路径名可以省略。而且一般可将被注册的 ActiveX 控件复制到 \windows\system 下，这样我们也不用在注册时输入控件的路径了。

(2) 如果想要解除对某一个 ActiveX 控件的注册，只需要在 regsvr32 后面加一个参数 “/u”，即 < 被注册过的 ActiveX 控件文件名 >。

在 VB 6.0 中可以点击界面上的：“工程”，在弹出的下拉菜单中，点击“部件”一项，随后出现了一个新的界面。在此界面上利用“浏览”按钮，找到并选中需要注册的控件，确定后便注册到“部件”界面的“控件”栏里。

利用上面两种方法进行控件注册后，便可以让部件开发人员在实际的编程中，使用该控件了。

(3) 使用安装程序制作软件——InstallShield。使用 Regsvr32.exe 程序来注册 ActiveX 控件，以及利用“浏览”来注册，虽然简单，但是都需要手工注册，在不用时，还需要手工解除，所以对于一个使用了该控件的应用程序来说并不实用。一般大型的应用软件都需要一个安装程序，在安装程序中解决 ActiveX 控件注册就非常实用了。使用 InstallShield 可以制作出专业级的安装程序，还可以注册其中的 ActiveX 控件；而且卸载软件时，可以自动注销以前注册的 ActiveX 控件。其做法就是按照 InstallShield 的向导，进行安装程序的制作，在进行到最后一步，点击“Finish”后，“InstallShield”将进行下一步的“详细定制”。选择“File Groups”选项，将其中包含需要自注册 ActiveX 控件文件项的“Self-Registered”属性改为“yes”。

(4) 安装过程中的自注册。ActiveX 控件在安装的时候必须被注册，方可以在应用程序中被调用。往往利用编程工具自带的安装制作工具可以达到这个目的。在 VB 6.0 中本身自带的创建安装程序的工具软件：Package & Deployment Wizard。只要将 ActiveX 控件包含在发布的文件中间，Package & Deployment Wizard 将根据需要自动将该控件打成自注册文件的属性。万一，我们没有成功的话，可以修改安装程序的安装文件列表 setup.lst，将相应宏中的参数设置为 DLLSelfRegister。如果不需要自注册，可以将该项删除（注意逗号要保留）。

例如下面是作者自己编的一个 ActiveX 控件，利用上述方法创建安装程序后，其后面的参数变为：

[Setup1 Files]

File1=@clock1.ocx,\$(WinSysPath),\$(DLLSelfRegister),\$(Shared),2/19/01 9:27:30 AM,36864,1.0.0.0

其中，\$(DLLSelfRegister) 就是标明自注册的宏参数设置。可以手工加或删。

(5) 补充一点。有些公司开发的 ActiveX 控件注册需要利用附带的专门的工具软件。运行该工具，就可以将相应的 ActiveX 控件注册。这仍然属于手工注册，这些 ActiveX 控件被发布时，一般享有版权，同时会有专门的说明。

## 2. ActiveX DLL 的注册

ActiveX DLL 的注册与 ActiveX 控件的注册基本上相似，上述用于 ActiveX 控件注册的方法基本都



适用于 ActiveX DLL 的注册。在此可以参照上述有关 ActiveX 控件注册的方法进行注册。

### 3. ActiveX EXE 的注册

作为一种进程外运行的组件, ActiveX EXE 的注册方法异于 ActiveX 控件和 ActiveX DLL 的注册, 并且在运用中有一定的难度。在此, 将它们的注册方法总结如下。

(1) “浏览”的方法。在 VB6.0 编程界面上, 进入“工程”, 在弹出的下拉菜单中间选择“引用”, 然后在出现的新界面上点击“浏览”, 找到并选中需要注册的组件, 确定后, 便可以在列表中间看到对应项了。

(2) 直接运行的方法。注册进程外组件 ActiveX EXE 时, 只要在 VB 6.0 的环境中运行一下该组件代码, 便可以实现注册了。这时 ActiveX EXE 的信息被加入到 Windows 注册表中。但是, 必须注意, 此信息只是在 VB 开发环境中运行此程序时被临时加入。当程序停止时, 有关如何访问这些对象的信息便从系统中清除掉。

(3) 利用安装制作工具。比如在 VB 6.0 中, 可以利用 VB 6.0 本身自带的创建安装程序的工具软件: Package & Deployment Wizard。在创建安装程序的过程中, 自动将其变为自注册。若不然, 同样可以修改安装程序的安装文件列表 setup.lst, 将相应宏中的参数设置为 EXESelfRegister。如果不需要自注册, 可以将该项删除 (注意逗号要保留)。

例如下面是作者自己编的一个进程外组件, 利用上述方法创建安装程序后, 其后面的参数变为:

[Setup1 Files]

File1=@CommThread.exe,\$(WinSysPath),\$(EXESelfRegister),\$(Shared),12/25/00 8:47:44 PM,57344,1.0.0.0



其中, 加粗的一项 \$(EXESelfRegister) 就是标明自注册的宏参数设置。可以手工加或删。

(4) 如果打算把进程外服务器安装到另外一台机器上自注册, 而且脱离编程环境时, 可以先将文件拷贝到另外机器上, 执行时在命令行上加上参数 “/regserver”, 如果取消注册, 可以在命令行上加上 “/unregserver”。使用 /regserver 命令选项注册进程外 COM 服务器时, 可以不去理会手工启动时程序工作的正常方式。Sub Main 或 Class\_Initialize 程序都不会被调用, 只有由当该编程环境包括在 EXE 文件中的注册逻辑才会运行。

## 10.2 图像列表控件



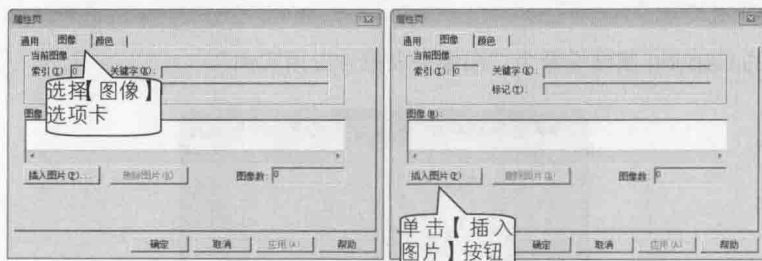
本节视频教学录像: 5 分钟

图像列表 (ImageList) 控件  不能单独使用, 它的主要功能是向其他公共控件提供图像。一种比较常见的使用方式是将图像列表控件和图像 (Image) 控件 、命令按钮 (CommandButton) 控件  等一起使用。

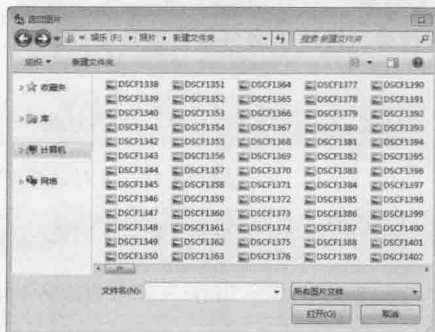
### 10.2.1 向图像列表控件添加图片

在设计过程中, 添加图像的操作需要通过设置图像列表控件来实现, 具体操作步骤如下。

- ① 在图像列表控件上右击, 选择【属性】菜单命令, 弹出【属性页】对话框, 选择【图像】选项卡。
- ② 单击【插入图片】按钮。



③ 在弹出的【选定图片】对话框中选定图片文件，然后单击【打开】按钮即可。



## 10.2.2 图像列表控件与其他控件关联

上面已经讲到，ImageList 控件用于向其他控件提供图片。向 ImageList 控件添加图片后，还需要将其他控件与 ImageList 控件关联起来。

在 Visual Basic 6.0 中，能使用 ImageList 控件图片的控件以及其相应的属性如表所示。


控件名称	可与 ImageList 关联的相应属性
ImageCombo 控件	ComboItemImage 属性、OverlayImage 属性
ListView 控件	ListItemImage 属性、Ico 属性
Toolbar 控件	ButtonImage 属性、ButtonHotImageList 属性、ButtonDisabledImageList 属性
TabStrip 控件	TabImage 属性

关于 ImageList 控件与其他控件关联的具体步骤，在后面会详细讲解。

## 10.2.3 图像列表控件的应用实例

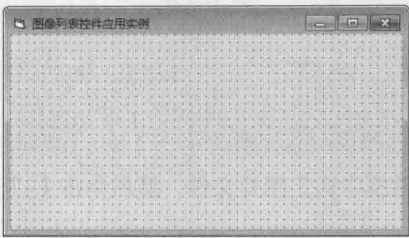
下面通过一个范例来学习图像列表控件的使用方法。

### 【范例 10-1】应用图像列表控件，设置 ImageList 控件与 Image 控件关联。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单

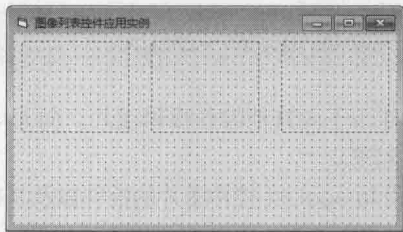
击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“图像列表控件应用实例”。



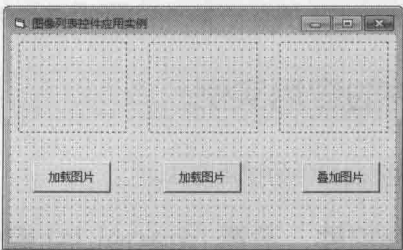
(3) 在窗体中添加 3 个 Image 控件，按照下表分别设置其属性。

名称	功能
Img1	显示【打开】对话框中指定的图像
Img2	显示【打开】对话框中指定的图像
Img3	显示 Img2 中图像叠加到 Img1 图像上的结果



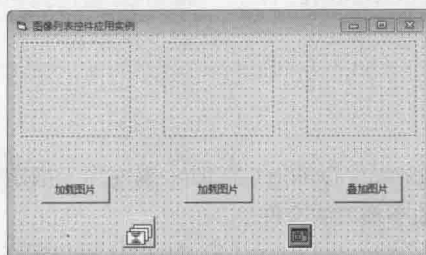
(4) 在窗体中添加 3 个 CommandButton 控件，按照下表分别设置其属性。

名称	Caption	功能
CmdOpen1	加载图片	单击该按钮弹出【打开】对话框，为 Img1 添加图像
CmdOpen2	加载图片	单击该按钮弹出【打开】对话框，为 Img2 添加图像
CmdOverlay	叠加图片	将 Img2 中的图像叠加到 Img1 图像上



(5) 在窗体中添加 1 个 ImageList 控件和 1 个通用对话框（CommonDialog）控件（需要在【部件】

对话框中的【控件】选项卡中选择【Microsoft Common Dialog Control 6.0】复选框，将通用对话框控件添加到工具箱中），并将各控件排列整齐，如图所示。



(6) 在 Form1 中没有控件的任意空白处右击，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，添加以下代码。

```
01 Private Sub Form_Load()  
02 Dim imgX As ListImage '定义 imgX 为 ListImage 中的一个对象  
03 End Sub
```

(7) 双击第 1 个“加载图片”按钮，在弹出的代码窗口中添加以下代码（代码 10-1-1.txt）。

```
01 Private Sub CmdOpen1_Click() '加载图片按钮 1 的鼠标单击事件代码  
02 CommonDialog1.Filter = "图形文件 (*.jpg;*.gif;*.bmp;*.ico;*.wmf) |*.jpg;*.gif;*.bmp;*.ico;*.wmf"  
'指定打开文件的扩展名  
03 CommonDialog1.ShowOpen '显示打开对话框  
04 Img1.Stretch = True '自动调节图片的大小  
05 Img1.Picture = LoadPicture(CommonDialog1.FileName)  
'按照打开对话框指定的文件名和路径打开图片  
06 Set imgX = ImageList1.ListImages.Add(1, , Img1.Picture)  
'将打开的图像添加到 ImageList1 控件中并指定索引值为 1  
07 End Sub
```

(8) 双击第 2 个“加载图片”按钮，在弹出的代码窗口中添加以下代码（代码 10-1-2.txt）。

```
01 Private Sub CmdOpen2_Click() '加载图片按钮 2 的鼠标单击事件代码  
02 CommonDialog1.Filter = "图形文件 (*.jpg;*.gif;*.bmp;*.ico;*.wmf) |*.jpg;*.gif;*.bmp;*.ico;*.wmf"  
'指定打开文件的扩展名  
03 CommonDialog1.ShowOpen '显示打开对话框  
04 Img2.Stretch = True '自动调节图片的大小  
05 Img2.Picture = LoadPicture(CommonDialog1.FileName)  
'按照打开对话框指定的文件名和路径打开图片  
06 Set imgX = ImageList1.ListImages.Add(2, , Img2.Picture)  
'将打开的图像添加到 ImageList1 控件中并指定索引值为 2  
07 End Sub
```

(9) 双击“叠加图片”按钮，在弹出的代码窗口中添加以下代码。

```

01 Private Sub CmdOverlay_Click() ' 叠加按钮的鼠标单击事件代码
02 ImageList1.MaskColor = vbWhite ' 设置 ImageList1 的屏蔽颜色为白色
03 Img3.Picture = ImageList1.Overlay(2, 1) ' 将 Img2 叠加到 Img1 上
04 End Sub

```

## 【代码详解】

在第(7)步第 03 行代码中, CommonDialog1 的 filter 属性用于指定在弹出的【打开】对话框中显示的文件类型, 这样可以过滤掉其他不相关的文件。

Img1.Picture = LoadPicture(CommonDialog1.FileName) 语句把在【打开】对话框中选定并打开的图片在 img1 控件中加载显示。

第(7)步中的代码可以与第(8)步进行类比。

在第(9)步第 03 行代码中, Img3.Picture = ImageList1.Overlay(2, 1) 语句使用 Overlay 叠加方法, 将第(7)和第(8)步添加到 ImageList1 中的图片叠加, 并在 img3 控件中加载显示。

## 【运行结果】

按快捷键【F5】运行程序, 分别单击【加载图片】按钮, 将“Sample\ch05\范例 5-1”文件夹内的两个图片依次加载到程序中, 然后单击【叠加图片】按钮, 即可把所加载的两张图片叠加显示, 效果如图所示。

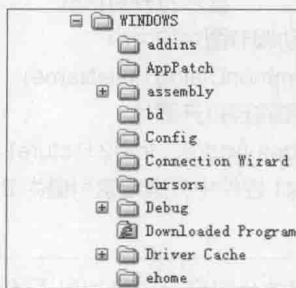


## 10.3 树状视图控件——统筹全局的好工具



本节视频教学录像: 4 分钟

树状视图 (TreeView) 控件在显示具有根节点的分层逻辑结构的列表时非常方便。例如, Windows 资源管理器就是一个非常典型的树状视图。



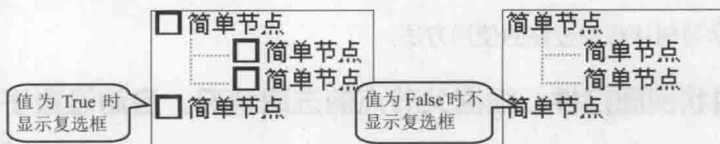
## 10.3.1 树状视图控件的主要属性、事件和方法

### 1. 树状视图控件的主要属性

下面介绍树状视图一些比较独特的属性。

#### (1) 复选框属性 (Checkboxes)

该属性用来设置是否显示复选框，该属性有两个备选值：True 和 False。值为 True 时在树的每一项旁边显示复选框；值为 False 时不显示复选框。



可以通过属性窗口中的 Checkboxes 下拉列表设置该属性的值。

#### (2) 风格属性 (Style)

该属性用来设置图形类型 (图像、文本、+/- 号、直线) 以及出现在树状视图控件中每个节点对象上的文本的类型。用户可以通过属性窗口中的【Style】下拉列表设置该属性。



### 2. 树状视图控件的事件

#### (1) 节点选择事件 (NodeCheck)

当一个节点对象被选择或者被取消选择时会触发该事件。语法是：

---

```
Event NodeCheck(ByVal Node As ComctlLib.Node)
```

---

#### (2) 节点单击事件 (NodeClick)

当一个节点对象被单击时，便会触发该事件。在单击节点对象之外的其他树状视图控件时将会触发标准的鼠标单击事件。语法是：

---

```
Private Sub object_NodeClick(ByVal node As Node)
```

---

### 3. 树状视图控件的方法

#### (1) 获取可视对象数量方法 (GetVisibleCount)

该方法将返回固定在树状视图控件内部区域的节点对象数量。语法是：



```
object.GetVisibleCount
```

## (2) Add 方法 (节点集合)


该方法在树状视图控件的节点集合中添加一个节点对象。语法是:

```
object.Add(relative, relationship, key, text, image, selectedimage)
```

## 10.3.2 树状视图控件的应用实例

下面通过一个范例来学习树状视图控件的使用方法。

### 【范例 10-2】应用树状视图控件，根据父节点的选取情况，自动设置子节点。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“树状视图控件使用实例”。

(3) 在窗体中添加 1 个树状视图 (Tree View) 控件，将此树状控件的 Checkboxes 属性设置为“True”。

(4) 将树状视图控件按如图所示进行调整。



(5) 在 Form1 中没有控件的任意空白处右击，在弹出的快捷菜单中选择【查看代码】选项，在弹出的代码窗口中添加以下代码 (代码 10-2.txt)。

```
01 Private Sub Form_Load()  
02     Call InitTreeData      ' 初始化控件的内容  
03     TreeView1.Nodes(1).Selected = True    ' 根节点为当前节点  
04     TreeView1.Nodes(1).EnsureVisible      ' 根节点可见  
05 End Sub  
06 Private Sub InitTreeData() ' 初始化控件的内容  
07     Dim i As Integer      ' 定义变量  
08     TreeView1.Nodes.Add , "根目录", "根目录"          ' 添加根目录  
09     For i = 1 To 5 ' 添加 1 级目录  
10         TreeView1.Nodes.Add "根目录", twChild, "1 级目录" & CStr(i), "1 级目录" & CStr(i)  
11     Next i  
12     For i = 1 To 5 ' 1 级目录 1 中添加 2 级目录
```

```

13     TreeView1.Nodes.Add "1 级目录 1", twwChild, "2 级目录 1" & CStr(i), "2 级目录 " & CStr(i)
14     Next i
15     For i = 1 To 5 '1 级目录 3 中添加 2 级目录
16         TreeView1.Nodes.Add "1 级目录 3", twwChild, "2 级目录 2" & CStr(i), "2 级目录 " & CStr(i)
17     Next i
18     For i = 1 To 5 '1 级目录 5 中添加 2 级目录
19         TreeView1.Nodes.Add "1 级目录 5", twwChild, "2 级目录 3" & CStr(i), "2 级目录 " & CStr(i)
20     Next i
21     For i = 1 To 5 '2 级目录 11 中添加 3 级目录
22         TreeView1.Nodes.Add "2 级目录 11", twwChild, "3 级目录 " & CStr(i), "3 级目录 " & CStr(i)
23     Next i
24     TreeView1.Nodes(1).Expanded = True '将节点展开
25 End Sub
26 Private Sub TreeView1_NodeCheck(ByVal Node As MSComctlLib.Node)
    '选取节点
27     gCheckChildrenBySelf TreeView1, Node.Index, Node.Checked
28     gCheckParentBySibling TreeView1, Node.Index
29 End Sub
30 Private Sub gCheckChildrenBySelf(TreeView1 As TreeView, ByVal curIndex As Integer, ByVal bCh
As Integer) '根据自身选取情况, 确定全选或取消其下所有子项
31     Dim n As Integer
32     If TreeView1.Nodes(curIndex).Children <= 0 Then
33         Exit Sub
34     Else
35         n = TreeView1.Nodes(curIndex).Child.Index
36         Do While n <> TreeView1.Nodes(curIndex).Child.LastSibling.Index
37             TreeView1.Nodes(n).Checked = bCh
38             gCheckChildrenBySelf TreeView1, n, bCh
39             n = TreeView1.Nodes(n).Next.Index
40         Loop
41         TreeView1.Nodes(n).Checked = bCh
42         gCheckChildrenBySelf TreeView1, n, bCh
43     End If
44 End Sub
45 Private Sub gCheckParentBySibling(TreeView1 As TreeView, ByVal curIndex As Integer)
    '根据同层、同父节点的选取情况, 确定是否选取其父节点(乃至更上层), 直至根节点
46     Dim n As Integer
47     Dim bHaveChecked As Boolean
48     If TreeView1.Nodes(curIndex).FirstSibling.Index = 1 Then
49         Exit Sub
50     Else
51         bHaveChecked = False
52         n = TreeView1.Nodes(curIndex).FirstSibling.Index

```

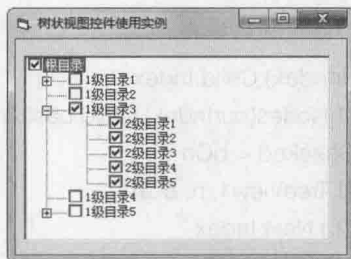
```

53 Do While n <> TreeView1.Nodes(curIndex).LastSibling.Index
54 Exit Do
55     If TreeView1.Nodes(n).Checked = True Then
56         bHaveChecked = True
57     End If
58     n = TreeView1.Nodes(n).Next.Index
59 Loop
60 If TreeView1.Nodes(n).Checked = True Then
61     bHaveChecked = True
62 End If
63 If bHaveChecked = True Then
64     TreeView1.Nodes(curIndex).Parent.Checked = vbChecked
65 Else
66     TreeView1.Nodes(curIndex).Parent.Checked = vbUnchecked
67 End If
68 gCheckParentBySibling TreeView1, TreeView1.Nodes(curIndex).Parent.Index
69 End If
70 End Sub

```

## 【运行结果】


按快捷键【F5】运行程序，最终效果如图所示。



## 10.4 选项卡控件



本节视频教学录像：5 分钟

选项卡 (TabStrip) 控件  就像我们常用的文件夹上的标签一样，选择某个标签就可打开相应的页面。选项卡控件很适合界面元素比较多以至于一个页面没有足够的面积容纳的情况，因为只要添加一个标签，就增加了一倍的界面面积。

### 10.4.1 选项卡控件的主要属性

(1) 多行属性 (MultiRow)

该属性用来确定选项卡是否可以多行显示。用户可以通过属性窗口中的【MultiRow】下拉列表设置多行属性的值。

### (2) 方位属性 (Placement)

本属性返回或设置一个值，指定选项卡将会显示在控件的哪一端。用户可以通过【属性】窗口中的【Placement】下拉列表设置方位属性的值。



### (3) 样式属性 (Style)

该属性返回或设置该控件的样式和外观。用户可以通过属性窗口中的【Style】下拉列表设置样式属性的值。



上述属性的设置都可以参照多重选择属性的设置，通过右击选项卡控件，在弹出的快捷菜单中选择【属性】选项，在弹出的【属性页】对话框中通过【通用】选项卡设置。




#### 提示

TabStrip 控件不是一个容器，本身无法包含页面。要包含实际的页面和大小，必须使用 Frame 控件或其他容器。

## 10.4.2 选项卡控件的应用实例

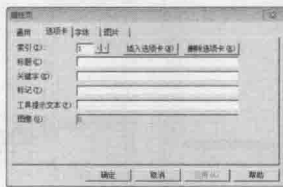
**【范例 10-3】应用选项卡控件，在选取不同的选项卡后，框架和标签也会随之变化。**

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

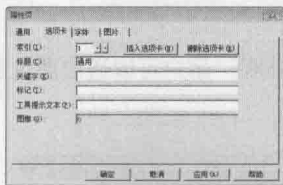
(2) 将 Form1 的 Caption 属性设置为“选项卡应用实例”。



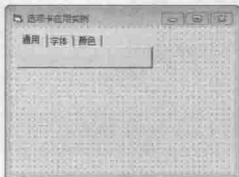
(3) 在窗体上创建一个选项卡控件，在选项卡控件上右击，在弹出的快捷菜单中选择【属性】选项，弹出【属性页】对话框，选择【选项卡】选项卡。



(4) 单击【插入选项卡】按钮，在【索引】为“1”的选项卡上设置【标题】为“通用”。



(5) 按照上一步的操作，再插入两个选项卡，将【索引】为“2”的【标题】设置为“字体”，将【索引】为“3”的【标题】设置为“颜色”，完成后的效果如图所示。

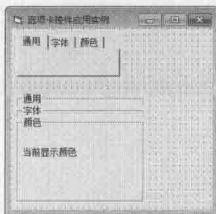


(6) 在窗体上创建一个包含 3 个 Frame 控件的控件组（采用控件复制粘贴的方法创建控件组），并按下表设置各控件的属性。

名称	Caption	Index
Frame1	通用	0
Frame1	字体	1
Frame1	颜色	2



(7) 在 Frame1 ( 0 ) 上绘制一个 Label 控件, 设置 Label 控件的 Caption 属性为 “当前显示的是通用选项卡”; 在 Frame1 ( 1 ) 上绘制一个 Label 控件, 设置 Label 控件的 Caption 属性为 “当前显示的是字体选项卡”; 在 Frame1 ( 2 ) 上绘制一个 Label 控件, 设置 Label 控件的 Caption 属性为 “当前显示的是颜色选项卡”。将各控件依次排列, 效果如图所示。



(8) 双击窗体空白处, 在弹出的代码窗口中添加以下代码 ( 代码 10-3.txt )。

```
01 Private Sub Form_Load() ' 初始化 TabStrip 控件的位置和大小
02     Dim i As Integer
03     TabStrip1.Top = 0      ' 使 TabStrip 控件大小随着窗体变化
04     TabStrip1.Left = 0
05     TabStrip1.Width = Me.ScaleWidth
06     TabStrip1.Height = Me.ScaleHeight
07     For i = 0 To 2 ' 设置 Frame 的位置和大小
08         Frame1(i).Top = TabStrip1.ClientTop
09         Frame1(i).Left = TabStrip1.ClientLeft
10         Frame1(i).Width = TabStrip1.ClientWidth
11         Frame1(i).Height = TabStrip1.ClientHeight
12     Next i
13     Frame1(0).ZOrder 0    ' 将通用属性页放在最顶层
14 End Sub
```

(9) 双击选项卡控件, 在弹出的代码窗口中添加以下代码。

```
01 Private Sub TabStrip1_Click() ' 选择不同的 Tab 时, 显示不同的属性页
02     Dim i As Integer
03     i = TabStrip1.SelectedItem.Index - 1
04     Frame1(i).ZOrder 0
05 End Sub
```

## 【代码详解】

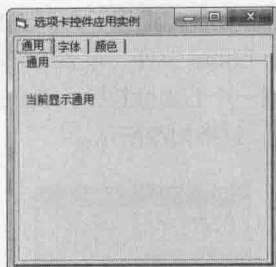
第(8)步代码首先将选项卡 TabStrip1 的位置居于窗体左上角, 并且设置宽高值与窗体一致, 然后通过 For 循环将 3 个 Frame 控件的大小调整为与 TabStrip1 相同。

第(9)步代码根据选择的选项卡的索引值来调整相对应的 Frame 控件的显示与隐藏。

## 【运行结果】

按快捷键【F5】运行程序, 最终效果如图所示。





## 10.5 进度条控件



本节视频教学录像: 5 分钟

软件执行每一个任务都需要一定的时间。为了让用户了解目前任务完成的情况, 使用进度条控件是一个很好的方法, 这样可以让用户对于当前程序进展、已完成进度和剩余进度等有比较直接的了解。

### 10.5.1 进度条控件的主要属性和方法

#### 1. 进度条控件的属性

##### (1) 最小值属性 (Min)

最小值属性用于返回或设置进度条控件的最小值, 默认值为 0。用户可以通过属性窗口中的【Min】文本框设置该属性的值。

##### (2) 最大值属性 (Max)

最大值属性用于返回或设置进度条控件的最大值, 默认值为 100。用户可以通过属性窗口中的【Max】文本框设置该属性的值。



#### 2. 进度条控件的方法


下表是进度条控件的 4 种常用方法和它们对应的功能。

方法名称	功能
Drag	用于控件的开始、结束和取消拖动操作
Move	移动控件
OLEDrag	引起部件初始化 OLE 拖放操作
ZOrder	将指定的控件放置在其图层的 z- 顺序的前端或后端

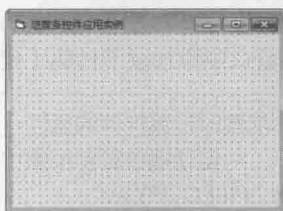
## 10.5.2 进度条控件的应用实例

下面通过一个实例来学习进度条控件的使用方法。

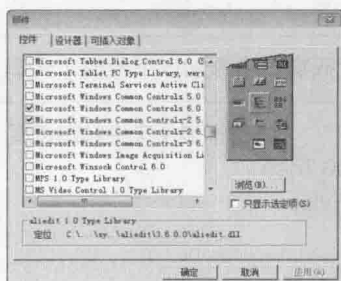
### 【范例 10-4】应用进度条控件，设计一个根据时间而变化的进度条。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

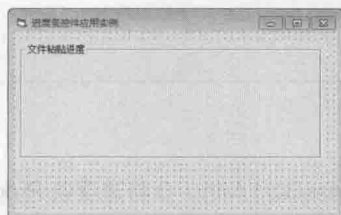
(2) 将 Form1 的 Caption 属性设置为“进度条控件应用实例”。



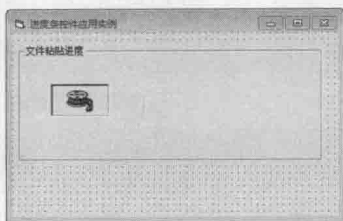
(3) 选择【工程】>【部件】菜单命令，在弹出的【部件】对话框中。选择【控件】选项卡，选中【Microsoft Windows Common Controls - 2.5.0 (sp2)】和【Microsoft Windows Common Controls 6.0】复选框，单击【确定】按钮，将动画 (Animation) 控件添加到工具箱中。



(4) 在窗体上添加一个 Frame 控件，将 Caption 属性设置为“文件粘贴进度”。



(5) 在 Frame1 控件上添加一个动画 (Animation) 控件, 将随书光盘中的 “Sample\ch05\ 范例 5-5\filecopy.avi” 文件复制到该程序所在的文件夹下。



(6) 在窗体上添加 1 个定时器 (Timer) 控件和 1 个进度条 (ProgressBar) 控件, 并将添加的控件排列如下图所示。



(7) 双击窗体空白处, 在弹出的代码窗口中添加以下代码 (代码 10-4-1.txt)。

```
01 Private Sub Form_Load() ' 窗体加载时设置控件的属性
02     Animation1.AutoPlay = True ' 自动播放复制录像
03     Animation1.Open App.Path & "FILECOPY.avi"
04     ProgressBar1.Min = 0 ' 设置 ProgressBar1 参数
05     ProgressBar1.Max = 100
06     ProgressBar1.Value = 0
07     Timer1.Interval = 500 ' 设置 Timer1 时间间隔
08 End Sub
```

(8) 双击计时器控件, 在弹出的代码窗口中添加以下代码 (代码 10-4-2.txt)。

```
01 Private Sub Timer1_Timer() ' 模拟显示文件粘贴进度
02     If ProgressBar1.Value = 100 Then
03         Animation1.AutoPlay = False
04         Animation1.Close
05     End
06 Else
07         ProgressBar1.Value = ProgressBar1.Value + 1
08     End If
09 End Sub
```

## 【代码详解】

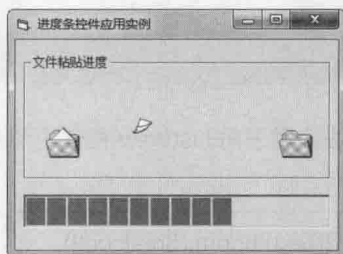
在第(7)步第 04 行代码中, ProgressBar1.Min = 0 将进度条 ProgressBar1 的最小值设为 0。在第 05

行代码中, `ProgressBar1.Max = 100`, 将最大值设为 100, 第 06 行代码 `ProgressBar1.Value = 0` 将初始值设为 0。

在第(8)步第 02 行代码中, 使用 `If` 判断语句, 如果进度条 `ProgressBar1` 的值为 100 则退出程序, 否则, 将进度条 `ProgressBar1` 的值加 1。

## 【运行结果】

按快捷键【F5】运行程序, 最终效果如图所示。



## 【拓展训练 10-1】进度控件的应用 2。

如果想使进度条初始值为 50, 并且装载的速度更快些, 应该怎么办呢? 只需修改 `ProgressBar1.Value` 和 `Timer1.Interval` 的值即可。具体修改的代码如下。

---

```
ProgressBar1.Value = 50
```

---

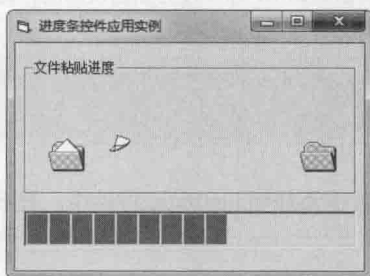
以及:

---

```
Timer1.Interval = 50
```

---

再次运行, 就可以看到修改后的效果。



## 10.6 视图控件 (ListView)



本节视频教学录像: 4 分钟

`ListView` 控件可以用来显示各项带图标列表, 也可以用来显示带有子项的列表, Windows 操作系统的资源管理器中文件夹窗口就是最好的应用例子。视图 `ListView` 位于“部件”对话框的 `Microsoft Windows Common 6.0 (SP6)` 选项中。

## 10.6.1 ListView 控件简介

ListView 控件可使用四种不同视图显示项目。通过此控件，可将项目组成带有或不带有列标头的列，并显示伴随的图标和文本。可使用 ListView 控件将称作 ListItem 对象的列表条目组织成下列 4 种不同的视图之一：大（标准）图标；小图标；列表；报表。

View 属性决定在列表中控件使用何种视图显示项目。还可用 LabelWrap 属性控制列表中与项目关联的标签是否可换行显示。另外，还可管理列表中项目的排序方法选定项目的外观。

## 10.6.2 添加数据

Add 方法是 ListItems 集合的方法，用于向 ListView 控件中添加 ListItem 对象，其语法格式如下。相关参数如下表所说明。

Object. ListItems.Add([Index],[key][Text],[Icon],[SmallIcon])

Add 方法中各参数的说明

语句类型	作用
object	必要参数。对象表达式，其值是 ListItem 集合
index	可选参数。指定在何处插入 ListItem 对象的整数。若未指定索引将 ListItem 对象添加到 ListItems 集合的末尾
Key	可选参数。唯一的字符串表达式，用来访问集合成员
Text	可选参数。与 ListItem 对象控件关联的字符串
Icon	可选参数。当 ListView 控件设为图标视图时，此参数设置从 ImageList 控件中选定的要显示的图标
SmallIcon	可选参数。当 ListView 控件设为小图标时，此参数设置从 ImageList 控件中选定的要显示的图标

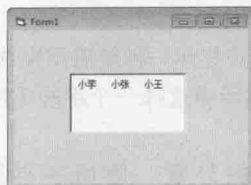
例如下面示例使用 Add 方法将数据添加到 ListView 控件中。代码如下。

```

01 Private Sub Form_Load()
02 ListView1.ListItems.Add 1,"employee1","小李" '添加员工小李
03 ListView1.ListItems.Add 2,"employee1","小张" '添加员工小张
04 ListView1.ListItems.Add 3,"employee1","小王" '添加员工小王
05 End Sub

```

程序运行效果如图所示。



### 10.6.3 创建报表视图

用 ListView 控件创建报表视图，首先要使用 ColumnHeader 对象的 Add 方法给报表添加一个表头，然后使用 ListSubItem 对象的 Add 添加内容。这里需要说明的是，ListSubItem 对象是否存在及其数目，都取决于 ColumnHeader 对象是否存在及其数目。也就是说，如果没有 ColumnHeader 对象，就不能创建任何 ListSubItem 对象。进一步说，ColumnHeader 对象的数目决定了可为 ListItem 对象设置的 ListSub 对象的数目。

### 10.6.4 创建大图标视图

用 ListView 控件创建大图标视图需要设置 ListView 控件的 View 属性值为 0~1vwIcon。

## 10.7 日期 / 时间控件 (DateTimePicker)

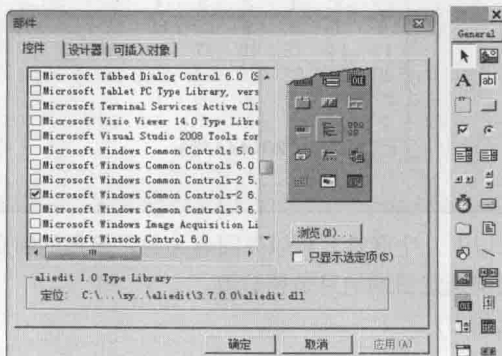


本节视频教学录像：5 分钟

DateTimePicker 控件一般用于让用户可以从日期列表中选择单个值。使您可以提供格式化的日期字段，使得进行日期选择很容易。运行时，单击控件边上的下拉箭头，会显示为两个部分：一个下拉列表，一个用于选择日期。

### 10.7.1 认识 DateTimePicker 控件

VB 默认的控件栏中是没有 DTPicker 日期控件的，添加过程：**【工具】>【部件】>【控件】>【Microsoft Windows Common Controls-2.6.0】>【应用】**命令。





DateTimePicker 控件,有以下两种操作模式。

下拉式日历模式(缺省)——允许用户显示一种能够用来选择日期的下拉式日历。

时间格式模式——允许用户在日期显示中选择一个字段(例如:月、日、年等),按下控件右边的上下箭头来设置它的值。

可以自定义控件的下拉式日历的外观。使用各种颜色属性,例如 CalendarBackColor、CalendarForeColor、CalendarTitleBackColor、CalendarTitleForeColor 和 CalendarTrailingForeColor。

允许创建属于你自己的颜色方案。



**提示**

DateTimePicker 控件是 ActiveX 控件组的一部分,包含在 MSCOMCT2.OCX 文件中。要在应用程序中使用 DateTimePicker 控件,必须将 MSCOMCT2.OCX 文件加入到工程之中。选择一个字段(例如:月、日、年等),按下控件右边的上下箭头来设置它的值。

## 10.7.2 设置和返回日期

DTPick 控件的 Value 属性设置为当前日期。如果在代码中更新了 DateTimePicker 的 Value 属性,控件会自动更新并反映出新的设置。Value 属性返回一个原始的日期值,或者空值。例如下面的代码。

```
01 Private Sub Form_load()  
02 DTPicker1.Value="2013-10-24" '设置日期  
03 End Sub
```

### 【运行结果】

单击工具栏中的【启动】按钮,即可看到程序运行结果。



DateTimePicker 控件具有以下几个属性,可以返回有关显示日期的特定信息。

- (1) Month 属性返回包含当前选定日期的月份整数值。
- (2) Day 属性返回当前选定的日。

(3) DayOfWeek 属性返回一个值，指出所选日期是星期几。(值根据 vbDayOfWeek 常量定义的值决定。)

(4) Year 属性返回包含当前选定日期的年份整数值。

(5) Week 属性返回包含当前选定日期的星期序号。

(6) 使用 Change 事件来确定用户何时更改了控件中的日期值。

(7) 使用 CheckBox 来选择无日期。

### 10.7.3 实时读取 DTPicker 控件中的日期

使用 DTPicker 控件的 change 事件可以确定用户何时更改了该控件中的日期值。在该事件中使用 Value 属性，便可实时读取 DTPicker 控件中的日期。例如：将适时读取的日期显示在立即窗口中，代码如下。

```
01 Private Sub DTPicker1_Change()  
02 Debug.print DTPicker1.Value ' 在立即窗口中显示日期  
03 End Sub
```

### 10.7.4 使用 CheckBox 属性来选择无日期

使用 CheckBox 属性能够指定控件是否返回日期。默认情况下，CheckBox 的值为 False，并且控件总是返回一个日期。要让用户能够指定无日期，可以将 CheckBox 属性设置为 True (例如，如果使用 DateTimePicker 控件输入工程的完成日期而该工程还没有完成)。如果 CheckBox 属性设为 True，那么在控件日期和时间左边的编辑部分中将出现一个小的复选框。如果这个复选框没有被选中，那么 Value 属性返回一个空值。如果选中了这个复选框，那么控件通过 Value 属性返回当前显示日期。

### 10.7.5 使用日期和时间的格式

DateTimePicker 提供了很强的灵活性，以便在控件中将日期和时间的显示格式化。可以使用所有标准的 Visual Basic 格式化字符串，也可以使用回调字段来创建自定义格式。

Format 属性决定了控件如何格式化原始日期值。可以从预定义的格式化选项选择一个，或使用控件的自定义格式化功能。



**提示**

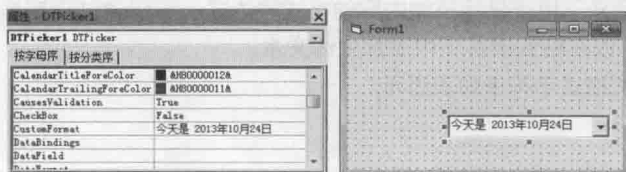
在 CustomFormat 属性定义的日期或者时间格式化前，要先将 Format 属性设置为 3-dtpCustom。

CustomFormat 属性定义了用于显示控件内容的格式表达式。可以通过指定格式字符串来告诉控件如何将日期输出格式化。可以在格式字符串中添加主体文本。

DTPicker 控件支持的格式字符串如表所示。

格式字符串	意义
d	1 或 2 位的日
dd	2 位日, 1 位值时前面加 0 (即显示 01)
ddd	3 字符, 星期缩写
dddd	星期全名
h	12 小时格式 1 或 2 位小时
hh	12 小时格式 2 位小时, 1 位值前面加 0
H	24 小时格式 1 或 2 位小时
HH	24 小时格式 2 位小时, 1 位值前面加 0
m	1 或 2 位分钟
mm	2 位分钟, 1 位值前面加 0
M	1 或 2 位月份
MM	2 位月份, 1 位值前面加 0
MMM	3 个字符, 表示月份缩写
MMMM	月份全名
s	1 或 2 位的秒
ss	2 位的秒, 1 位值时前面加 0
t	1 个字母 AM/PM (上午或者下午) 的缩写
tt	2 个字母 AM/PM (上午或者下午) 的缩写
y	1 位年份 (如 2013 显示 3)
yy	年份的最后两位
yyy	完整的年份
yyy	回调字段, 使程序员可以控制显示字段, 可以使用一系列多个 "X" 来表示唯一的回调字段

可以在格式字符串中添加主体文本。例如, 如果想设置 DTPicker 控件按照“今天是: 2013 年 10 月 24 日”的格式显示当前日期。那么需要设置 CustomFormat 属性的格式字符串为: 今天是: '2013' 年 "10" 月 "24 日"。其设置和运行结果如下图所示。



## 10.7.6 使用 DTPicker 控件计算日期或天数

DTPicker 控件还可以用于日期和天数的计算。例如，在酒店住宿登记模块中在客人退房时自动计算客人住宿日期。计算退宿日期和住宿天数使用 DTPicker 控件非常方便。通过加减一个整数得到需要的日期；通过两个日期控件相减就可以得到相差的天数。代码如下。

```
01 d=DTPicker1.Value-DTPicker2.Value ' 计算天数，d 代表天数
02 DTPicker1.Value=DTPicker2.Value+d ' 计算日期
```

## 10.8 工具栏控件



本节视频教学录像：6 分钟

工具栏是软件的重要组成部分，工具栏把菜单中一些最常用的功能以按钮的形式显示在一个栏目中，以供用户快速调用。

### 10.8.1 工具栏控件的主要属性和事件

#### 1. 添加工具栏（Toolbar）控件

工具栏控件  包含在 Microsoft Windows Common Controls 6.0 中，所以要使用工具栏控件，要将其添加到工具箱中，具体添加方法在上一节中已经讲述过，这里不再重复。

#### 2. 工具栏控件的常用属性

工具栏控件的常用属性为外观风格属性（Style），该属性用于返回或设置一个值来控制工具栏的外观。该属性有 0 和 1 两个值，为 0 表示工具栏将采用默认风格，为 1 表示工具栏将采用平面样式。用户可以通过该控件所对应的属性窗口中的【Style】下拉列表设置该属性。



#### 3. 工具栏常用的事件

##### (1) 按钮单击事件（ButtonClick）。

当用户单击 Toolbar 控件内的 Button 对象时将触发该事件。语法是：

```
Private Sub object_ButtonClick(ByVal button As Button)
```

因为用户可以用【自定义工具栏】对话框重新安排按钮对象, 所以 Index 属性值有时并不总是指示按钮的位置, 而用 Key 属性值检索 Button 对象则是一种更合理的方式。

(2) 按钮菜单单击事件 (ButtonMenu-Click)。

当用户单击一个 ButtonMenu 对象时将触发该事件。语法是:

```
Private Sub object_ButtonMenuClick([index As Integer,]ByVal ButtonMenu As ComctlLib.ButtonMenu)
```

(3) 改变事件 (Change)。

控件的内容只要发生改变, 就会触发该事件。语法是:

```
Private Sub object_Change([index As Integer])
```

## 10.8.2 工具栏控件的应用实例

下面通过一个实例来学习工具栏控件的使用方法。在本实例中, 我们要为前面章节中编写好的程序“文本框控件应用实例”添加一个工具栏。

### 【范例 10-5】应用工具栏控件, 为窗体添加工具栏。

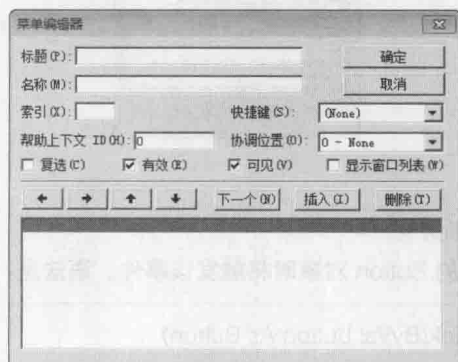
第 1 步: 创建菜单

(1) 打开随书光盘“Final\ch04\范例 4-2”文件夹中的工程文件。

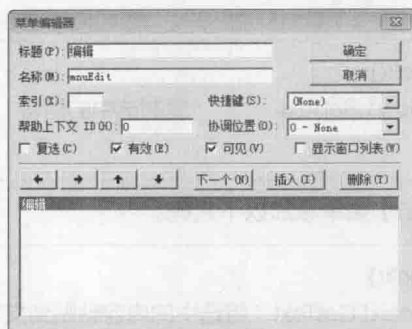
(2) 将 Form1 的 Caption 属性设置为“工具栏控件应用实例”, 并调整窗体大小和控件的位置。将文本框控件的名称设置为“TxtShow1”。



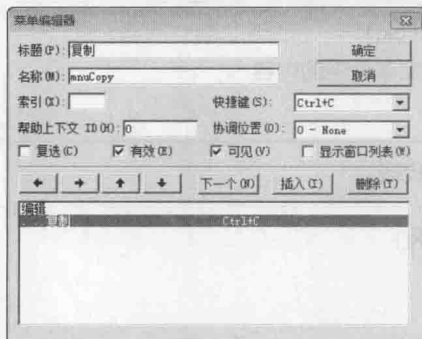
(3) 选择【工具】>【菜单编辑器】菜单命令, 打开【菜单编辑器】对话框。



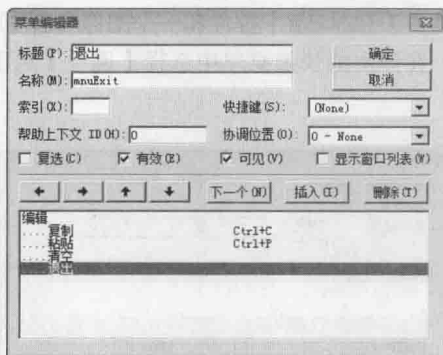
(4) 在【标题】文本框中输入“编辑”，在【名称】文本框中输入“mnuEdit”，创建一个名为“编辑”的菜单。



(5) 单击【下一个】按钮，在【标题】文本框中输入“复制”，在【名称】文本框中输入“mnuCopy”，在【快捷键】下拉列表中选择“Ctrl+C”，并单击 ➔ 按钮，创建一个名为“复制”的子菜单。



(6) 参照上一步的操作，并按照下表所示创建【编辑】菜单的其余子菜单。



子菜单标题	子菜单名称	快捷键
粘贴	mnuPaste	Ctrl + P
清空	mnuEmpty	无
退出	mnuExit	无

## 第2步: 为菜单添加代码

(1) 单击窗体上添加的【编辑】菜单, 在下拉菜单中选择【复制】命令, 在弹出的代码窗口中输入以下代码。

```
01 Private Sub mnuCopy_Click()  
02     Clipboard.SetText TxtShow1.SelText      '复制选中的内容  
03 End Sub
```

(2) 参照上面的步骤, 为【粘贴】菜单添加以下代码。

```
01 Private Sub mnuPaste_Click()  
02     TxtShow1.SelText = Clipboard.GetText '将选中的内容粘贴到文本框中  
03 End Sub
```

(3) 为【清空】菜单添加以下代码。

```
01 Private Sub mnuEmpty_Click()  
02     TxtShow1.Text = "" '清空文本框中的内容  
03 End Sub
```

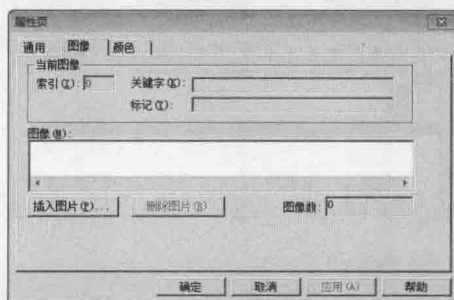
(4) 为【退出】菜单添加以下代码。

```
01 Private Sub mnuExit_Click()  
02     End '退出程序  
03 End Sub
```

## 第3步: 创建工具栏

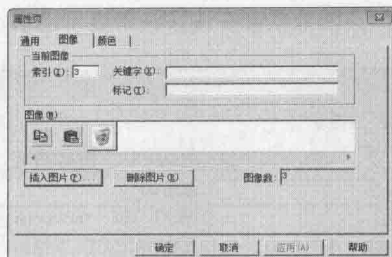
(1) 在 Form1 中添加一个工具栏 (Tool Bar) 控件和一个图像列表 (ImageList) 控件。

(2) 在图像列表控件上右击, 在弹出的快捷菜单中选择【属性】选项, 弹出【属性页】对话框, 选择【图像】选项卡。

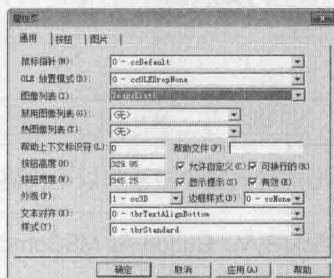


(3) 单击【插入图片】按钮, 弹出【选定图片】对话框, 把随书光盘中“Sample\ch05\范例5-2\工具栏控件的应用”中的3个图标文件依次添加进去, 单击【确定】按钮。

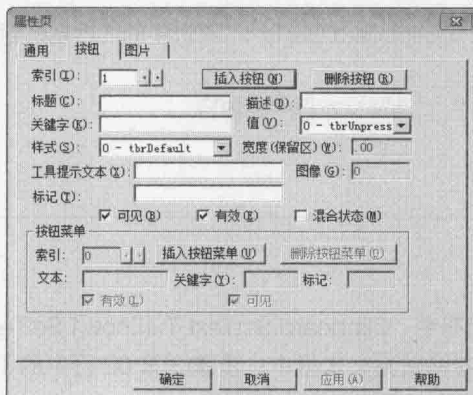




(4) 在工具栏控件上右击，在弹出的快捷菜单中选择【属性】选项，弹出工具栏控件的【属性页】对话框。在【图像列表】下拉列表中选择前面创建的 ImageList1，把工具栏控件与图像列表控件链接起来。

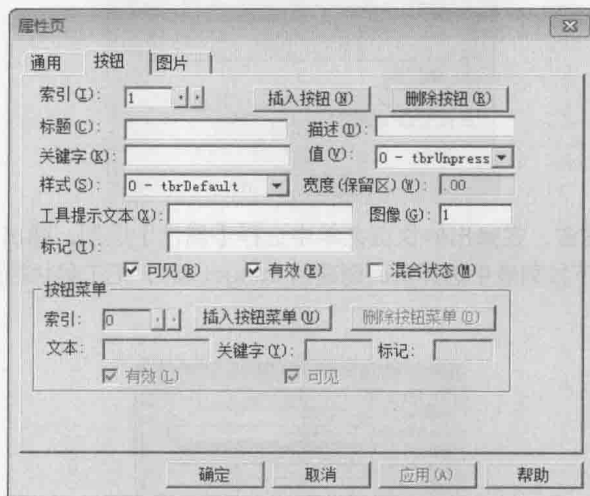


(5) 选择【属性页】对话框中的【按钮】选项卡，单击【插入按钮】按钮，窗体的工具栏中将会添加一个空白按钮。



(6) 参照步骤(5)的操作，添加 3 个按钮。在【属性页】对话框中的【图像】文本框中，输入前面图

像列表控件中保存的图像的索引值, 该图像就会在该按钮中显示出来。



(7) 双击添加的工具栏控件, 在弹出的代码窗口中添加以下代码 (代码 10-5.txt)。




```
01 Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
02     Select Case Button.Index
03         Case 1
04             Clipboard.SetText TxtShow1.SelText '复制选中的内容
05         Case 2
06             TxtShow1.SelText = Clipboard.GetText '将选中的内容粘贴到文本框中
07         Case 3
08             TxtShow1.Text = "" '清空文本框中的内容
09     End Select
10 End Sub
```

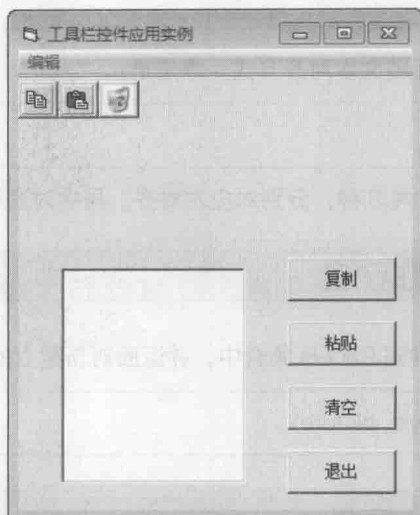
## 【代码详解】

在第 2 步步骤(1)第 02 行代码中, Clipboard.SetText TxtShow1.SelText 将选中的文本内容复制到剪贴板中; 步骤(2)则将剪贴板内容粘贴到文本框中; 步骤(3)第 02 行中的 TxtShow1.Text = "" 语句将文本框 TxtShow1 的 Text 属性赋值为空, 即清空文本框内容。

第 3 步步骤(7)代码为工具栏的 3 个按钮添加了与第 2 步的菜单相对应的命令, 所以不管执行的是菜单命令还是按钮命令, 其结果是一致的。

## 【运行结果】


按快捷键【F5】运行程序, 最终效果如图所示。单击程序界面上的【复制】按钮、选择【编辑】>【复制】菜单命令或者直接单击工具栏中的【复制】按钮, 都可以把在文本框选中的文本复制到剪贴板。同样, 【粘贴】和【清空】命令也可以通过直接单击工具栏中对应的和按钮来完成。



## 10.9 状态栏控件



本节视频教学录像: 2 分钟

状态栏 (StatusBar) 控件  通常用来显示窗体当前运行的状态, 及一些特定的关键信息。例如打开 Word 2003 后, 在下方状态栏上会显示当前光标所在的行和列, 以及关于位置的其他信息。

### 10.9.1 状态栏控件的属性

#### 1. 显示风格属性 (Style)

该属性用于设置状态栏控件显示的样式。用户可以通过属性窗口中的 Style 下拉列表框设置本属性的值。该属性有 0 和 1 这两个值, 为 0 时状态栏采用默认样式, 为 1 时状态栏采用单行显示的样式。



## 2. 对齐属性 (Alignment)

该属性用于设置一个对象的标题文本对齐方式。语法是:

```
object.Alignment [= number]
```

其中, number 值有 0、1、2 共 3 种, 分别对应左对齐、居中对齐和右对齐。

## 10.9.2 状态栏控件的方法

Add 方法是将 Panel 对象添加到 Panels 集合中, 并返回对新建立的 Panel 对象的引用。语法是:

```
object.Add(index, key, text, style, picture)
```

## 10.9.3 状态栏控件的事件

状态栏控件中比较特殊的事件如下。

(1) 单击面板事件 (PanelClick)。

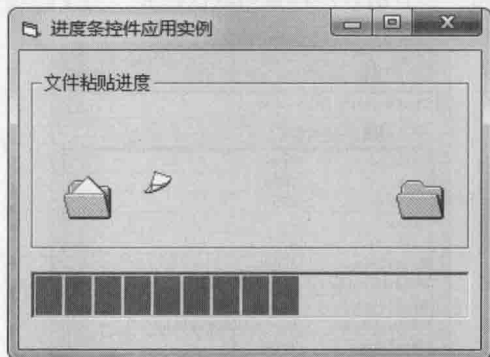
单击面板事件和标准的鼠标单击事件相似, 但单击面板事件是在任何一个 Panel 对象上按下, 然后释放鼠标按钮时才出现。语法是:

```
Private Sub object_PanelClick(ByVal panel As Panel)
```

(2) 双击面板事件 (PanelDbClick)。

双击面板事件和标准的鼠标双击事件相似, 但双击面板事件是两次在 Panel 对象上按下, 然后释放鼠标时才出现。语法是:

```
Private Sub object_PanelDbClick(ByVal panel As Panel)
```



## 10.10 高手点拨



本节视频教学录像：2 分钟

### 1. 常用 ActiveX 控件

ActiveX 控件	ActiveX 部件
TabStrip 页框	Windows 通用控件 Microsoft Windows Common Controls 6.0
ToolBar 工具栏	Windows 通用控件 Microsoft Windows Common Controls 6.0
ProgressBar 进程条	Windows 通用控件 Microsoft Windows Common Controls 6.0
TreeView 分层显示	Windows 通用控件 Microsoft Windows Common Controls 6.0
ListView 排列显示	Windows 通用控件 Microsoft Windows Common Controls 6.0
ImageList 图象列表	Windows 通用控件 Microsoft Windows Common Controls 6.0
Slider 滑块	Windows 通用控件 Microsoft Windows Common Controls 6.0
ImageCombo	Windows 通用控件 Microsoft Windows Common Controls 6.0
CommonDialog	Microsoft Common Dialog Control 6.0
MMControl1 多媒体	Microsoft Multimedia Control 6.0
MediaPlayer 媒体播放器	Microsoft Media Player

### 2. 在工具栏中添加复选（切换）按钮

工具栏中的复选按钮是指当按钮被按下以后就保持被按下的状态，只有下次再按才会弹起来，这就是工具栏的复选切换状态。要实现这样的效果，必须将它的 STYLE 属性设为 tbrCheck，这个设置可以在工具栏的属性页中完成。方法是右击工具栏并选择 Properties 选项以打开属性页，单击属性页的 Buttons 选项卡，选择要用的按钮，将它的形式 STYLE 设为 tbrCheck 即可。

### 3. 怎样做出像 IE 一样的平面工具栏

IE、Word 等流行软件的工具栏在通常状态下是平面的，只有当鼠标移过时才会突起，这样的效果通过 VB 工具栏本身是无法实现的，虽然可以用贴图的方法来模拟这种效果但却十分麻烦，简便的方法是通过调用 WIN32 API 函数来实现。其思路是用 SendMessage 函数向工具栏发送设置显示样式 TB\_SETSTYLE 的消息来改变工具栏的显示效果。

# 10.11 实战练习

## 一、思考题

- 1. 扩展控件和标准控件有什么区别？
- 2. 要设置进度控件 ProgressBar1 的最小值为 10、最大值为 80、初始值为 25，代码应该怎样写？

## 二、上机题

在 Visual Basic 6.0 中设计一个应用程序，要求完成以下功能。

(1) 窗体标题为“工具栏控件应用”，窗体中包含 1 个文本框控件，1 个工具栏控件，工具栏控件共 4 个按钮。

(2) 工具栏控件 4 个按钮的属性如表所示。

按钮标题	索引	按钮功能
黑体	1	设置字体为黑体
斜体	2	设置字体为斜体
重置	3	重置字体样式
退出	4	退出程序

(3) 单击【黑体】按钮，文本框中的文本字体显示为黑体；单击【斜体】按钮，文本框中的文本字体显示为斜体；单击【重置】按钮，文本框中的文本字体显示为正常字体；单击【退出】按钮，退出程序。

# 第11章



本章视频教学录像：21 分钟

## 鼠标、键盘的另类编程应用——鼠标、键盘事件


鼠标和键盘是用户和计算机交互最常使用的工具，应用程序的大部分功能需要鼠标和键盘配合使用。在 Visual Basic 6.0 中，窗体和大部分控件都能对用户的鼠标和键盘事件做出响应。本章介绍鼠标和键盘事件的相关知识。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握“鼠标按键按下”事件
- ☐ 掌握“鼠标按键释放”事件
- ☐ 熟悉“移动鼠标”事件
- ☐ 掌握“键盘按键”事件
- ☐ 熟悉“键盘按下”事件
- ☐ 熟悉“键盘弹起”事件



# 11.1 鼠标事件

 本节视频教学录像：10 分钟

Visual Basic 应用程序能够根据不同的鼠标事件和键盘事件做出相应的响应，例如窗体、图片框等都能检测鼠标指针的位置，并可判定其按键情况，还能响应鼠标按键与【Shift】、【Ctrl】或【Alt】等键的各种组合。

常见的鼠标事件有“单击”事件 (Click)、“双击”事件 (DblClick)、“鼠标按键按下”事件 (MouseDown)、“鼠标按键释放”事件 (MouseUp)、“移动鼠标”事件 (MouseMove) 等。

在前面的章节里我们已经多次看到“单击”事件 (Click) 和“双击”事件 (DblClick)，因此下面只介绍后 3 种鼠标事件。


## 11.1.1 “鼠标按键按下”事件 (MouseDown)

“鼠标按键按下”事件 (MouseDown) 是指当鼠标任意按键被按下时，将触发的事件。语法是：

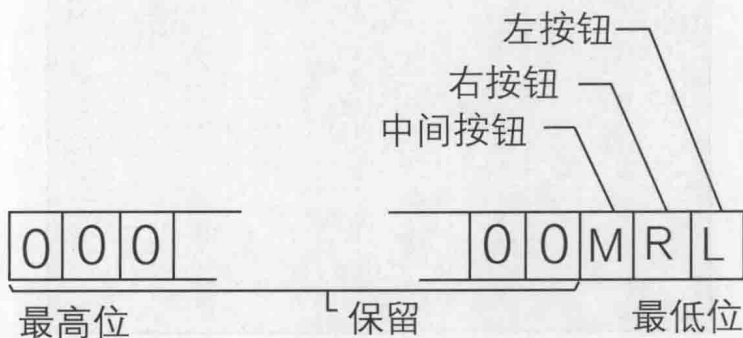
```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

语句中的“Button”用以返回一个值来表示鼠标哪个按键被按下了，Button 取值所对应的鼠标按键如表所示。

名称	值	对应按下的按键
vbLeftButton	1	左键
vbRightButton	2	右键
vbMiddleButton	4	中间键
	0	未按下任何键
vbLeftButton+ vbRightButton	3	左键和右键
vbLeftButton+ vbMiddleButton	5	左键和中间键
vbRightButton+ vbMiddleButton	6	右键和中间键
vbRightButton+ vbLeftButton+vbMiddleButton	7	按下 3 个键



**提示** button 参数的每位代表一个状态或条件。这些值被表示成整数。如图所示，3 个最低位分别表示鼠标的左按键、右按键和中间按键。每一位的默认值为 0(False)。如果未按下任何按钮，则三位的二进制值为 000。如果按下左按钮，则二进制值（或模式）变为 001，左按钮的位值由 0(False) 变为 1(True)。



语句中的“Shift”用以返回一个值表示当按键被按下时，【Ctrl】、【Shift】和【Alt】3个键的状态，具体含义如表所示。

名称	值	描述
vbShiftMask	1	【Shift】键被按下
vbCtrlMask	2	【Ctrl】键被按下
vbAltMask	4	【Alt】键被按下
	0	未按下任何键
vbShiftMask+ vbCtrlMask	3	同时按下【Shift】和【Ctrl】键
vbShiftMask+ vbAltMask	5	同时按下【Shift】和【Alt】键
vbAltMask+ vbCtrlMask	6	同时按下【Ctrl】和【Alt】键
vbShiftMask+ vbCtrlMask+ vbAltMask	7	同时按下【Shift】、【Ctrl】和【Alt】键




#### 提示

鼠标事件用来识别和响应各种鼠标状态，把这些状态看作独立的事件，不要把鼠标与 Click 事件和 DblClick 事件混为一谈。按下鼠标按钮并释放时，Click 事件只能把此过程识别为单击操作。鼠标事件不同于 Click 事件和 DblClick 事件之处还在于，鼠标事件能够区分各鼠标按钮与【Shift】、【Ctrl】、【Alt】键。

下面通过一个实例来学习“鼠标按键按下”事件（MouseDown）的应用方法。

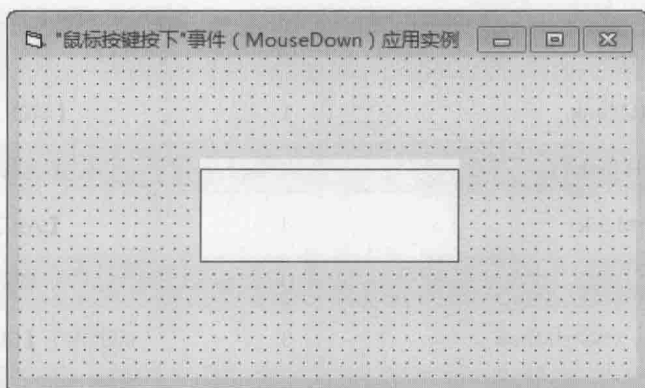
### 【范例 11-1】应用“鼠标按键按下”事件（MouseDown）的应用，按下鼠标按键后，会在窗体中显示相应的按键信息。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“鼠标按键按下”事件（MouseDown）应用实例。



(3) 在窗体中添加 1 个 Frame 控件，将其 Appearance 属性设定为 “0 - Flat”，并调整其大小和位置，如下图所示。



(4) 右击窗体空白处，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，添加以下代码 (代码 11-1.txt)。

```
01 Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
02     Select Case Button          ' 根据鼠标按键的不同设定不同的 Caption 值
03         Case 0
04             Label1.Caption = " 没有按键 "
05         Case 1
06             Label1.Caption = " 您按下了鼠标左键 "
07         Case 2
08             Label1.Caption = " 您按下了鼠标右键 "
09         Case 3
10             Label1.Caption = " 您按下了鼠标左键和右键 "
11         Case 4
12             Label1.Caption = " 您按下了鼠标中间键 "
13         Case 5
14             Label1.Caption = " 您按下了鼠标左键和中间键 "
15         Case 6
16             Label1.Caption = " 您按下了鼠标右键和中间键 "
```

```

17 Case 7
18 Label1.Caption = "您按下了鼠标的三个键"
19 End Select
20 End Sub

```

标签控件的 Appearance 属性默认为 1 或 3D 的，但这样在程序运行时，标签和窗体之间的边界就不可见。将其设定为 0 或 Flat 的话，标签显示为白底，便于与窗体区分。

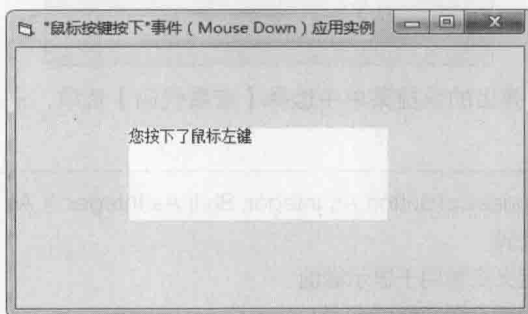
当然也可以修改标签控件的 BorderStyle 属性为 1 - Fixed Single，这样也能达到区分的效果。

## 【代码详解】

代码中的“Button”在运行时根据鼠标按键的不同，返回从 0 到 7 之间的一个值分别表示不同的结果。本例通过一个 Select 语句，先判断 Button 返回的值，然后根据不同的值在标签中显示不同的结果。

## 【运行结果】

按快捷键【F5】运行程序，在窗体中点击鼠标按键，标签控件中就会显示对应的按键结果。例如按下鼠标左键，则标签显示“您按下了鼠标左键”。



## 11.1.2 “鼠标按键释放”事件 (MouseUp)


“鼠标按键释放”事件 (MouseUp) 是指当鼠标任意按键被释放时，将触发该事件。语法是：

```
Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

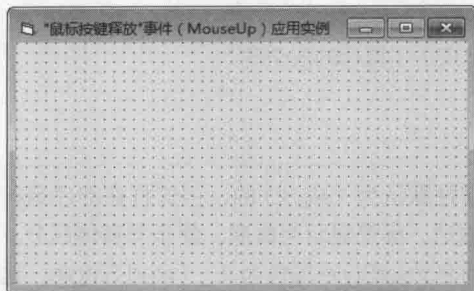
语句中的参数值和“鼠标按键按下”事件 (MouseDown) 一致，这里不再赘述。

下面通过一个实例来学习“鼠标按键释放”事件 (MouseUp) 的应用方法。

### 【范例 11-2】应用“鼠标按键释放”事件 (MouseUp)，当松开鼠标时，显示按键的相应信息。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“鼠标按键释放”事件 (MouseUp) 应用实例。



(3) 在窗体中新建一个标签控件, 将其 Caption 属性设置为空, BorderStyle 属性设置为“1”, 将该标签拖放到合适的位置并调整大小, 效果如下图所示。

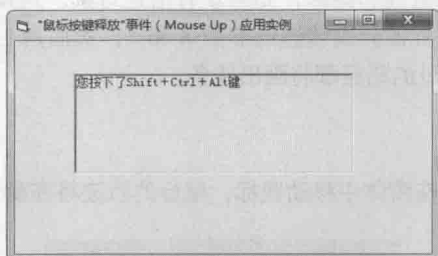


(4) 右击窗体空白处, 在弹出的快捷菜单中选择【查看代码】选项, 进入代码窗口, 添加以下代码 (代码 11-2.txt)。

```
01 Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
    ' 鼠标按键释放事件代码
02 Dim text As String ' 定义变量用于显示键值
03 Select Case Shift ' 根据不同变量值选择
04 Case 1 ' 变量值为 1, 即 Shift 键被按下
05 text = "Shift 键" ' 显示用户按下的键值
06 Case 2 ' 变量值为 2, 即 Ctrl 键被按下
07 text = "Ctrl 键" ' 显示用户按下的键值
08 Case 3 ' 变量值为 3, 即同时按下 Shift 和 Ctrl 键
09 text = "Shift + Ctrl 键" ' 显示用户按下的键值
10 Case 4 ' 变量值为 4, 即 Alt 键被按下
11 text = "Alt 键" ' 显示用户按下的键值
12 Case 5 ' 变量值为 5, 即同时按下 Shift 和 Alt 键
13 text = "Shift + Alt 键" ' 显示用户按下的键值
14 Case 6 ' 变量值为 6, 即同时按下 Ctrl 和 Alt 键
15 text = "Ctrl + Alt 键" ' 显示用户按下的键值
16 Case 7 ' 变量值为 7, 即同时按下 Shift、Ctrl 和 Alt 键
17 text = "Shift + Ctrl + Alt 键" ' 显示用户按下的键值
18 End Select
19 Label1.Caption = "您按下了 " & text ' 将标签显示的内容赋值
20 End Sub
```

## 【运行结果】

按快捷键【F5】运行程序，按住【Ctrl】、【Alt】或者【Shift】3 个按键中的任意一个或多个不松开，然后在窗体空白处单击鼠标任意键并释放，屏幕上就会显示当前键盘按下的键。



通过从上面上个事件的学习，我们可以看出“鼠标按键按下事件”和“鼠标按键释放”不同。它们的区别在于：“鼠标按键按下”事件在按下鼠标按键后就立即触发，而“鼠标按键释放”事件则必须是按下鼠标按键并弹起后才会触发。

所以，如果同时设定了“鼠标按键按下”事件和“鼠标按键释放”事件，那么在点击鼠标时，“鼠标按键按下”事件会在“鼠标按键释放”事件之前发生。


### 11.1.3 “移动鼠标”事件 (MouseMove)

“移动鼠标”事件 (MouseMove) 是指当移动鼠标时，将触发该事件。语法是：

```
Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

下面通过一个实例来学习“移动鼠标”事件 (MouseMove) 的应用方法。

#### 【范例 11-3】应用“移动鼠标”事件 (MouseMove)，在移动鼠标时，进行相应的操作。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“移动鼠标”事件 (MouseMove) 应用实例。



(3) 右击窗体空白处，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，添加以下代码。

```
01 Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
02 Line -(X, Y) '使用 Line 方法画线
```

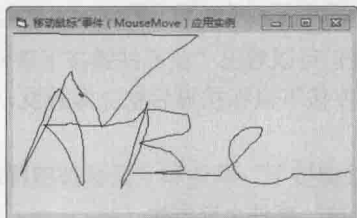
```
03 End Sub
```

## 【代码详解】

Line 方法用于在对象上绘制直线和矩形, 如果没有指定对象, 则具有焦点的窗体将作为对象。“移动鼠标”事件将鼠标当前所在坐标值分别传递给参数 X 和 Y, 然后用 Line 方法以 X 和 Y 为值画线。所以程序运行后, 鼠标在窗体内经过的路径都将画出线条。

## 【运行结果】

按快捷键【F5】运行程序, 在窗体中移动鼠标, 鼠标的轨迹将在窗体中显示出来。



## 11.2 键盘事件



本节视频教学录像: 9 分钟

键盘事件同鼠标事件一样, 都是用户与程序之间进行交互操作的主要元素。利用键盘事件可以响应多种键盘操作, 也可以解释、处理 ASCII 字符。

### 11.2.1 “键盘按键”事件 (KeyPress)

当用户按下某个字符键的时候将会触发该事件。

语法是:


```
Private Sub object_KeyPress(KeyAscii As Integer)
```

当该事件被触发后, 被按的那个键所对应的 ASCII 码值被传递给事情过程的 KeyAscii 参数。需要注意的是, 只有获得焦点的对象才可以捕获键盘事件, Keyascii 为返回一个标准数字 ANSI 键代码的整数, 当 KeyAscii 参数值为 0 时, 便取消按键, 这样对象便接收不到字符了。

那么什么是焦点呢, 在 Visual Basic 中, 可以把焦点理解为接收用户鼠标或键盘输入的能力。只有当对象具有焦点时, 才可以接收用户的输入。我们平时在使用 Windows 操作系统的时候, 经常会打开多个窗口以完成不同的任务, 但是在任何一个时间, 只有一个窗口获得焦点, 这个获得焦点的窗口就是当前正在操作的窗口。如果没有焦点这个概念, 当我们在键盘中输入信息或者单击鼠标的时候, 程序将无法判断要将这些信息传递给哪一个窗口或哪一个对象。

**【范例 11-4】应用“键盘按键”事件 (KeyPress), 将按键字符转换为大写字符。**



(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 窗体的 Caption 属性设置为“键盘按键”事件 (KeyPress) 应用实例。



(3) 双击工具箱中的【文本框】(TextBox) 控件按钮，在窗体中添加一个文本框，并将文本框的 Text 属性设置为空，然后把文本框拖放到合适的位置并调整其大小，如图所示。



(4) 右击窗体空白处，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，添加以下代码。

```
01 Private Sub Text1_KeyPress(KeyAscii As Integer)
02     KeyAscii = Asc(UCase(Chr(KeyAscii))) ' 将按键转换为大写字符
03 End Sub
```

## 【代码详解】

Chr 函数用于返回其后面所接字符码所代表的字符，如 Chr(65) 将返回 A。

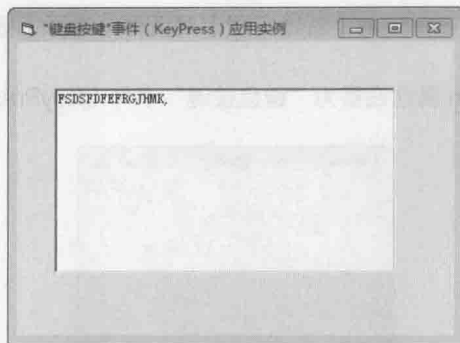
Ucase 函数将其后面所接字符串全部转换为大写。

Asc 函数将返回所接字符串的首字母的字符值，即 ASCII 值。

程序运行时：当按下键盘上得某个键时，所按下键的 ASCII 值首先赋予参数 KeyAscii，然后 Chr 函数以按下的键的 ASCII 值返回字符值给 Ucase 函数，接着 Ucase 函数将其转换为大写，最后 Asc 函数返回转换后的大写字母的 ASCII 值，再次赋予参数 KeyAscii，所以按下的键全被转换为大写后再输出。

## 【运行结果】

按快捷键【F5】运行程序。无论在文本框中输入大写字母还是小写字母，都会显示为大写字母，如图所示。



## 11.2.2 “键盘按下”事件 (KeyDown)

“键盘按下”事件 (KeyDown) 是指当对象拥有焦点并且按下键盘按键时将会触发的事件。语法是:

```
Private Sub Form_KeyDown(keycode As Integer, shift As Integer)
Private Sub object_KeyDown([index As Integer,]keycode As Integer, shift As Integer)
```

参数 keycode 将返回被按键物理位置代码; shift 参数返回一个整数, 该整数反映了【Shift】键、【Ctrl】键和【Alt】键的状态。shift 参数为 1、2 和 4 时, 分别表示【Shift】键、【Ctrl】键和【Alt】键被按下。




“键盘按键”事件和“键盘按下”事件的区别在于: 按下键盘上的任意一个键, 都会触发“键盘按下”事件; 而要触发“键盘按键”事件, 则必须按下的是对应标准 ASCII 字符的按键。

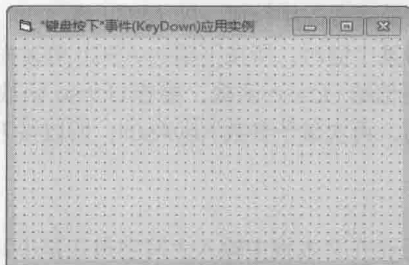
### 提示

下面通过一个实例来学习“键盘按下”事件 (KeyDown) 的应用方法。

### 【范例 11-5】应用“键盘按下”事件 (KeyDown), 判断用户按下的按键情况。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 将 Form1 窗体的 Caption 属性设置为“键盘按下”事件 (KeyDown) 应用实例。



(3) 右击窗体空白处, 在弹出的快捷菜单中选择【查看代码】选项, 进入代码窗口, 添加以下代码 (代码 11-5.txt)。

```
01 Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)      ' 键盘按下事件
02     Select Case Shift        ' 根据按键情况不同, 显示不同的信息
```

```

03      Case 1
04      Print " 您刚才按下的是 SHIFT 键 "
05      Case 2
06      Print " 您刚才按下的是 CTRL 键 "
07      Case 4
08      Print " 您刚才按下的是 ALT 键 "
09      Case 3
10      Print " 您刚才同时按下了 SHIFT 和 CTRL 键 "
11      Case 5
12      Print " 您刚才同时按下了 SHIFT 和 ALT 键 "
13      Case 6
14      Print " 您刚才同时按下了 CTRL 和 ALT 键 "
15      Case 7
16      Print " 您刚才同时按下了 SHIFT、CTRL 和 ALT 键 "
17      End Select
18 End Sub

```

## 【运行结果】

按快捷键【F5】运行程序。分别按下【Ctrl】键、【Shift】键、【Alt】键，或同时按下其中的两个或 3 个键，相应的按键结果将显示在窗口中，如图所示。



### 11.2.3 “键盘弹起”事件 (KeyUp)

“键盘弹起 (KeyUp)”事件是指当对象拥有焦点并且松开键盘按键时将会触发该事件。语法是：

```

Private Sub Form_KeyUp(keycode As Integer, shift As Integer)
Private Sub object_KeyUp ([index As Integer], keycode As Integer, shift As Integer)

```

参数 keycode 将返回被按键物理位置代码；shift 参数返回一个整数，该整数反映了【Shift】键、【Ctrl】键和【Alt】键的状态。shift 参数为 1、2 和 4 时，分别表示【Shift】键、【Ctrl】键和【Alt】键被按下后又松开。

需要注意的是：KeyPress 和 keycode 并不总是相同的，KeyPress 传递的参数是按键代表的字符，

因此有大小写之分；而 keycode 传递的参数是键盘上该按键的物理位置的代码，因此没有大小写之分。



**提示**

键盘事件彼此并不相互排斥。按下键盘某一键时将生成 KeyDown 和 KeyPress 事件，而松开此键后则会生成 KeyUp 事件。当用户按下一个 KeyPress 不能检测的键时将触发 KeyDown 事件，而松开此键后则会生成 KeyUp 事件。



本节视频教学录像：2 分钟

## 11.3 高手点拨

Visual Basic 应用程序能够响应多种键盘事件和鼠标事件。在 Visual Basic 中，提供了三种键盘事件：一是 KeyPress，按下对应某 ASCII 字符的键；二是 KeyDown，按下键盘的任意键；三是 KeyUp，释放键盘的任意键。同时，也提供了三种鼠标事件：一是 MouseUp，释放任意鼠标键按钮；二是 MouseDown，按下任意鼠标键按钮；三是 MouseMove，移动鼠标指针到屏幕新位置。通过响应这些键盘及鼠标事件，应用程序能对其位置及状态的变化做出响应操作。

## 11.4 实战练习

1. 在窗体上添加一个标签控件，编写代码，使程序运行时按下鼠标左键，标签变为绿色；释放鼠标时，标签变为蓝色。

2. 在 Visual Basic 6.0 中设计一个应用程序，要求完成以下功能。

(1) 窗体标题为“鼠标事件应用”。

(2) 利用“移动鼠标”事件在窗体中画图，并且每次触发“移动鼠标”事件都在当前鼠标坐标处绘制 1 个圆圈。

提示：Circle 方法用于绘制圆、椭圆或弧形，语法为：

---

Circle (X,Y) R

---

X, Y 为所绘制图形的坐标，R 为半径。

---

# 第12章



本章视频教学录像：43 分钟

## 程序与用户的交互——菜单和对话框设计

灵活地运用菜单栏可以使设计出来的软件在操作上更加的方便和灵活，同时，也可以显出程序员更加的专业。适当地运用对话框，可以为用户提供良好的人机交互。本章介绍菜单栏的应用、输入对话框的设计、消息对话框的设计及通用对话框的应用等内容，并通过具体的实例讲解，使读者学习起来更加轻松。

### 本章要点（已掌握的在方框中打钩）

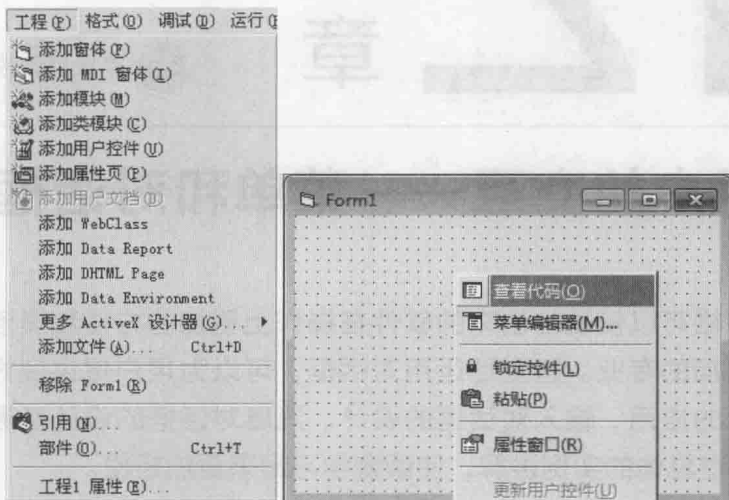
- ☐ 了解菜单编辑器
- ☐ 掌握下拉式菜单设计
- ☐ 掌握弹出式菜单设计
- ☐ 熟悉自定义菜单设计
- ☐ 掌握模式对话框和无模式对话框
- ☐ 熟悉预定义对话框设计
- ☐ 了解通用对话框设计

## 12.1 魅力化妆师——菜单设计



本节视频教学录像: 18 分钟

菜单是软件的重要组成部分。菜单由于具有使用方便和占用界面空间小等特点, 因此在各种软件中被广泛使用。在我们使用的 Visual Basic 开发环境中, 常用的菜单有下拉式菜单和弹出式菜单两种。其中, 下拉式菜单一般固定在程序界面中的某个位置, 而弹出式菜单则经常在右击对象的时候弹出。



下面以“龙马酒店管理系统”为例, 简单说明一下菜单的组成。



**菜单栏:** 菜单栏位于标题栏的下方, 可以包括一个或多个菜单标题。

**菜单项:** 菜单项由主菜单项和子菜单项组成。如图中的【住宿管理】就是一个主菜单项, 而【退房结账】则是一个一级子菜单项, 如果系统需要, 还可以设置二级子菜单项或多级子菜单项。

**分隔符:** 分隔符主要用于对同类的菜单项进行分组显示。

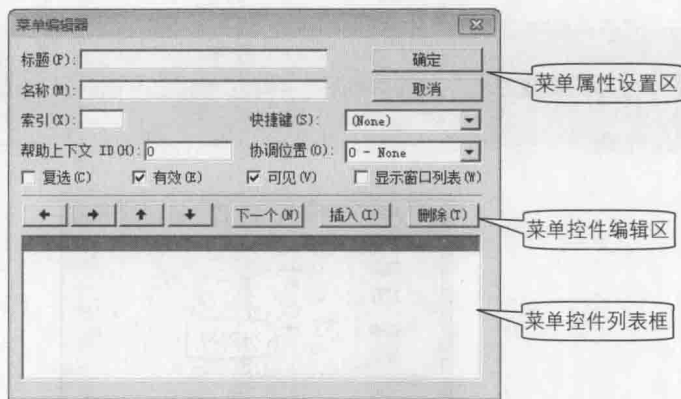
**快捷键:** 快捷键用于在键盘上执行快速启动命令。

## 12.1.1 菜单编辑器

在 Visual Basic 中，可以使用菜单编辑器来设计菜单。可以采用以下 3 种方式打开【菜单编辑器】对话框。

- (1) 选择【工具】菜单中的【菜单编辑器】菜单项。
- (2) 在窗体中右击，在弹出的快捷菜单中选择【菜单编辑器】菜单项。
- (3) 按快捷键【Ctrl+E】。

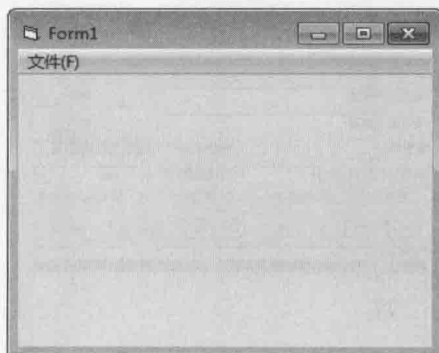
通过以上任一命令打开【菜单编辑器】对话框，其中主要分为菜单属性设置区、菜单控件编辑区和菜单控件列表框等 3 部分。



- (1) 菜单属性设置区：用来对现有菜单进行设置和编辑。
- (2) 菜单控件编辑区：用来设置和调整各个菜单项目之间的关系，以及添加和删除菜单项目。
- (3) 菜单控件列表框：用来查看各个菜单项目之间的关系。

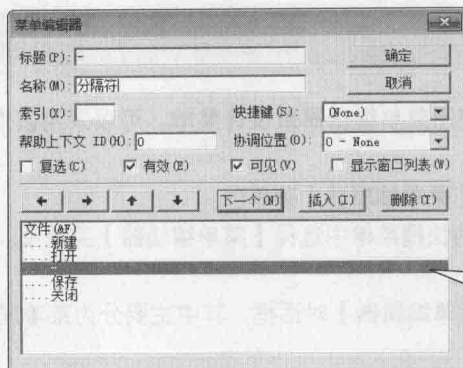
### 1. 菜单属性设置区

(1)【标题】属性。【标题】属性用于设置菜单项的标题，即显示在控件上的文本。除了可以设置显示的文本外，还可以通过在括号内利用“&”加上一个字母设置菜单项的键盘访问键，如“文件(&F)”，表示用户可以通过快捷键【Alt + F】打开该菜单。菜单项上并不显示“&”字符，而是将快捷键对应的字符以圆括号括起来并加上下划线。



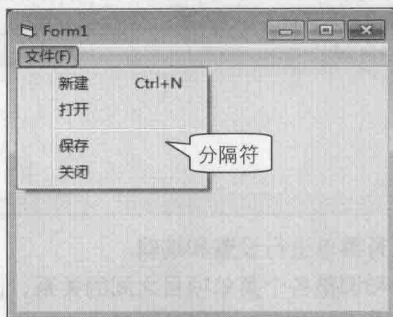
如果某一菜单中的内容比较多，可以将其分组，组与组之间添加一个标题属性为“-”的菜单项，这样在画面上会显示一条横线。





## 技巧

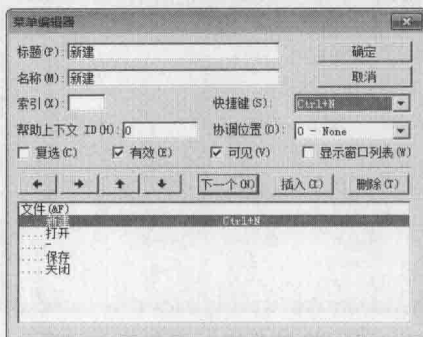
添加一个标题属性为“-”的菜单项后，菜单组与组之间就多了个用以分隔菜单项的横线。



(2)【名称】属性。【名称】属性是菜单项的控件名。在编写代码的时候如果要引用该菜单项，就需要引用【名称】属性中的控件名。

(3)【索引】属性。和控件数组类似，使用数组来管理菜单项也是比较方便的。【索引】属性就是表示当前菜单项在菜单数组中的位置。

(4)【快捷键】属性。为了提高软件的使用效率，可以为比较常用的菜单项设置快捷键，比如设置【新建】的快捷方式为【Ctrl+N】，这样当我们按键盘上的【Ctrl+N】组合键的时候，就能够实现用鼠标选择该菜单项所产生的相同效果。



(5)【帮助上下文 ID】属性。【帮助上下文 ID】属性是为了方便我们在软件的帮助文档中快速定位当

前所选项目的帮助材料。

(6)【协调位置】属性。该属性用于决定是否在容器窗口中显示菜单以及如何显示菜单。

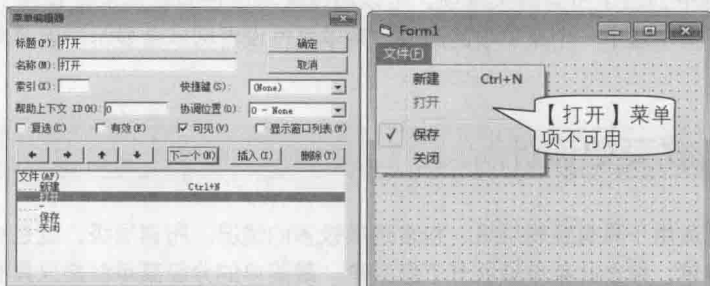
(7)【复选】属性。该属性用于决定是否在菜单项左侧显示复选标记。



**注意**

如果在某一菜单项设置中选中【复选】复选框，则当该菜单项被选中时，旁边就会出现一个选中标记“√”。

(8)【有效】属性。有时我们希望只选择与目前操作相关的某些菜单项目，这个时候就可以使用【有效】属性。



**提示**





【有效】属性是布尔值。当【有效】属性为 True 时，当前菜单项可以响应鼠标单击事件；当【有效】属性为 False 时，当前菜单项为灰色并不响应鼠标单击事件。

(9)【可见】属性选项用于决定当前菜单项是否可见。如果我们取消选择【可见】属性，该菜单项就不会显示在菜单中。



2. 菜单控件编辑区

菜单控件编辑区用来设置和调整各个菜单项目之间的关系，以及添加和删除菜单项目。主要功能如表所示。

功能按钮	功能
向左箭头 	每次单击将选定的菜单项提升一个等级，最多可以创建 4 个等级
向右箭头 	每次单击将选定的菜单项降低一个等级，最多可以创建 4 个等级
向上箭头 	每次单击将选定的菜单项在同级菜单中向上移动一个位置
向下箭头 	每次单击将选定的菜单项在同级菜单中向下移动一个位置
【插入】按钮	在列表中选定的菜单项上方添加一个菜单项
【下一个】按钮	将光标移动到下一个菜单项
【删除】按钮	删除选中的菜单项


3. 菜单控件列表框

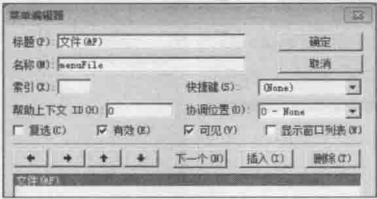
菜单控件列表框中列出了所有的菜单项，可以通过菜单控件列表框来查看各个菜单项目之间的关系。在菜单控件列表框中选中一个菜单项的时候，菜单属性设置区就会显示所选菜单项的各个属性，以供用户修改和设置。

12.1.2 下拉式菜单设计

下拉式菜单特别适用于具有逻辑层级、包含选项较多的情况。所谓层级，就是将各种功能分类，将类别列表做成上级菜单，单击此菜单项展开下级菜单。最简单的分级菜单就是只具有一个层级的单级菜单。

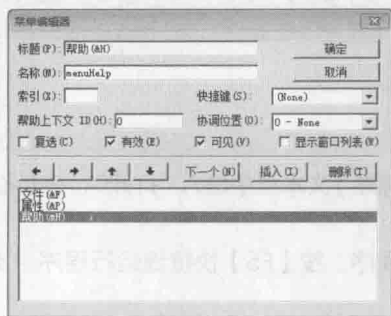
**【范例 12-1】制作 3 个单级菜单，这 3 个单级菜单分别为“文件”、“属性”和“帮助”，并为这 3 个单级菜单设置快捷键。**

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。
- (3) 在【标题】文本框中输入“文件(&F)”，在【名称】文本框中输入“menuFile”，创建一个【文件】菜单。



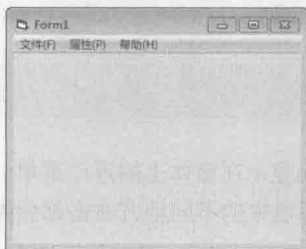
(4) 单击【下一个】按钮，在【标题】文本框中输入“属性(&P)”，在【名称】文本框中输入“menuProp”，创建一个【属性】菜单。

(5) 重复上面的步骤，再创建一个【帮助】菜单，其中【标题】为“帮助(&H)”，【名称】为“menuHelp”，单击【确定】按钮。



## 【运行结果】

保存程序，按快捷键【F5】运行程序，查看效果，如图所示。




提示

因为这是单级菜单，没有给它们做下级菜单。很显然，单级菜单并不总是能够满足我们的需求。如果只是单级菜单，那它和按钮控件就没有什么区别了。下拉菜单的魅力就在于能够通过一级级的下拉菜单容纳非常多的内容，而当我们不需要使用它的时候，它又会自动收回，只占用很少的界面空间。


## 【拓展训练 12-1】

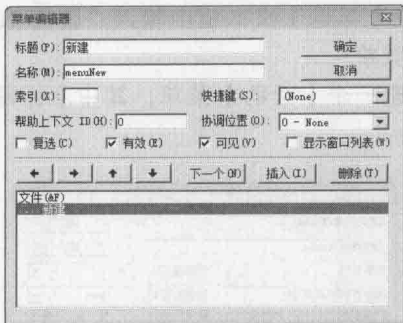
制作一个多级菜单，即在一级菜单（文件）下面包含有两个二级菜单（新建和打开）。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。

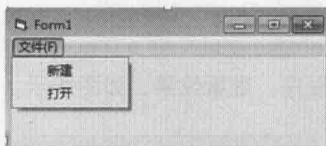
(3) 在【标题】文本框中输入“文件(&F)”，在【名称】文本框中输入“menuFile”，创建一个【文件】菜单。

(4) 单击向右箭头  1 次，在【标题】文本框中输入“新建”，在【名称】文本框中输入“menuNew”，创建一个【新建】二级菜单。



(5) 单击【下一个】按钮，在【标题】文本框中输入“打开”，在【名称】文本框中输入“menuOpen”，创建一个【文件】二级菜单。

(6) 单击【确定】按钮，保存程序，按【F5】快捷键运行程序。当单击【文件】按钮时，将会弹出下拉菜单选项，如图所示。




### 12.1.3 弹出式菜单设计

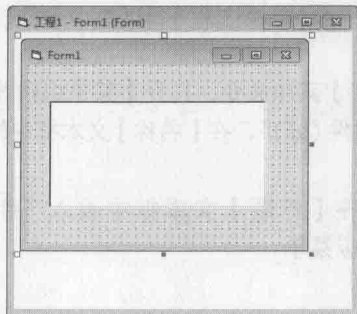
弹出式菜单是一种独立于菜单栏而显示在窗体上的浮动菜单。弹出式菜单广泛应用在和程序上下文相关的菜单中，在 Visual Basic 开发环境中的不同地方右击都会弹出不同的菜单。这些弹出的菜单都是和用户正在使用的功能紧密结合的。

**【范例 12-2】设置一个弹出式菜单，当单击鼠标右键时，在弹出的快捷菜单中呈现出所设置的菜单项。**

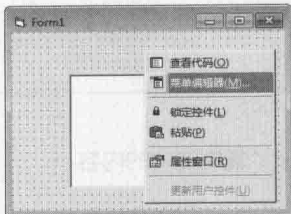
#### 1. 界面设计

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

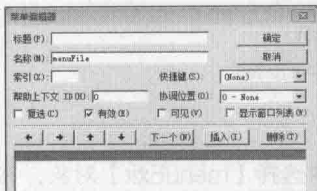
(2) 在 Form1 窗体中添加一个文本框控件，并将文本框控件的 Text 属性设置为空。



(3) 在窗体的空白处右击，在弹出的快捷菜单中选择【菜单编辑器】菜单项。

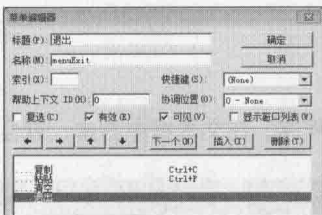


(4) 在弹出的【菜单编辑器】对话框的【名称】文本框中输入“menuFile”，撤选【可见】复选框，创建一个空白菜单。



(5) 为刚创建的菜单项创建子菜单，参数设置如表所示。

标题	名称	快捷键
复制	menuCopy	Ctrl + C
粘贴	menuPaste	Ctrl + P
清空	menuEmpty	无
退出	menuExit	无



(6) 单击【确定】按钮。

## 2. 编写代码

(1) 在 Form1 窗体的空白处右击，在弹出的快捷菜单中选择【查看代码】菜单项，在弹出的代码窗口的对象下拉列表中选择【Text】对象，在过程下拉列表中选择【MouseDown】事件，并输入以下代码。

```

01 Private Sub Text1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
02   If Button = vbRightButton Then ' 判断是否右键单击
03     PopupMenu menuFile ' 指定的当前坐标位置显示弹出式菜单
04   End If
05 End Sub
    
```



(2) 在代码窗口的对象下拉列表中选择【menuCopy】对象, 在过程下拉列表中选择【Click】事件, 并输入以下代码。

```
01 Private Sub menuCopy_Click()  
02   Clipboard.SetText Text1.Text '复制选中的内容  
03 End Sub
```

(3) 在代码窗口的对象下拉列表中选择【menuEmpty】对象, 在过程下拉列表中选择【Click】事件, 并输入以下代码。

```
01 Private Sub menuEmpty_Click()  
02   Text1.Text = "" '清空文本框中的内容  
03 End Sub
```

(4) 在代码窗口的对象下拉列表中选择【menuExit】对象, 在过程下拉列表中选择【Click】事件, 并输入以下代码。

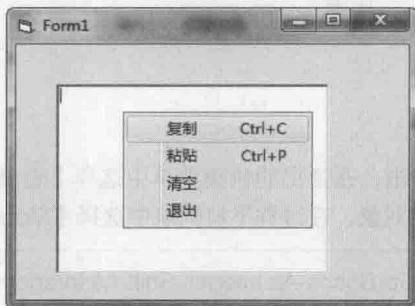
```
01 Private Sub menuExit_Click()  
02   End '退出程序  
03 End Sub
```

(5) 在代码窗口的对象下拉列表中选择【menuPaste】对象, 在过程下拉列表中选择【Click】事件, 并输入以下代码。

```
01 Private Sub menuPaste_Click()  
02   Text1.Text = Clipboard.GetText '将选中的内容复制到文本框中  
03 End Sub
```

## 【运行结果】

保存程序, 按快捷键【F5】运行程序。在文本框中右击, 就会弹出设计的弹出式菜单。




**注意**

在文本框中右击时, 首先弹出系统快捷菜单, 再在文本框另一空白处右击, 则会弹出设计的弹出式菜单。

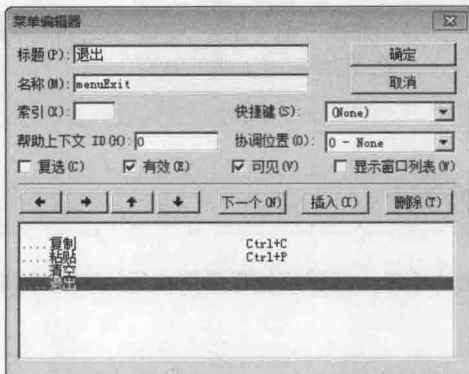


## 【拓展训练 12-2】

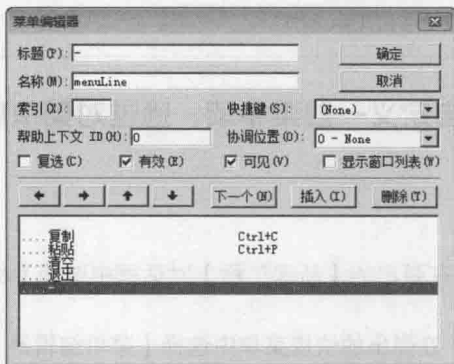
为下拉菜单列表设置一条分组线，用于区分不同含义的菜单名。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 按照【范例 12-2】“1. 界面设计”中步骤(2)~(5)的操作，创建一个菜单项和一个子菜单。



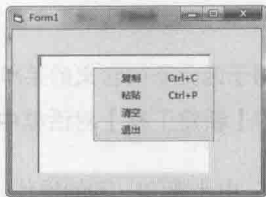
(3) 单击【下一个】按钮，在【标题】文本框中输入“-”，在【名称】文本框中输入“menuLine”，创建一条分组线。



(4) 在 Form1 的空白处右击，在弹出的快捷菜单中选择【查看代码】菜单项，在弹出的代码窗口的对象下拉列表中选择【Text】对象，在过程下拉列表中选择【MouseDown】事件，并输入以下代码。

```
01 Private Sub Text1_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
02     If Button = vbRightButton Then
    ' 判断是否右键单击
03     PopupMenu menuFile
    ' 指定的当前坐标位置显示弹出式菜单
04     End If
05 End Sub
```

(5) 单击【确定】按钮，保存程序，按【F5】快捷键运行程序。在文本框中右击，就会弹出刚才所设计的弹出式菜单，可以看到在弹出式菜单的最后多出一条分组线。



技巧

弹出式菜单中的各菜单项并不能执行相关命令，因为没有对各菜单项的事件进行代码编写。同时，分组线的设计是不分下拉式菜单和弹出式菜单的。

## 12.1.4 自定义菜单设计

有的时候我们需要更加智能的菜单，能够根据程序所执行的任务或者自定义设置动态增减菜单。添加菜单的语法是：


```
Load mnuArray(mnuArray.UBound+1)
```

删除菜单的语法是：

```
UnLoad mnuDynArray(mnuArray.UBound)
```

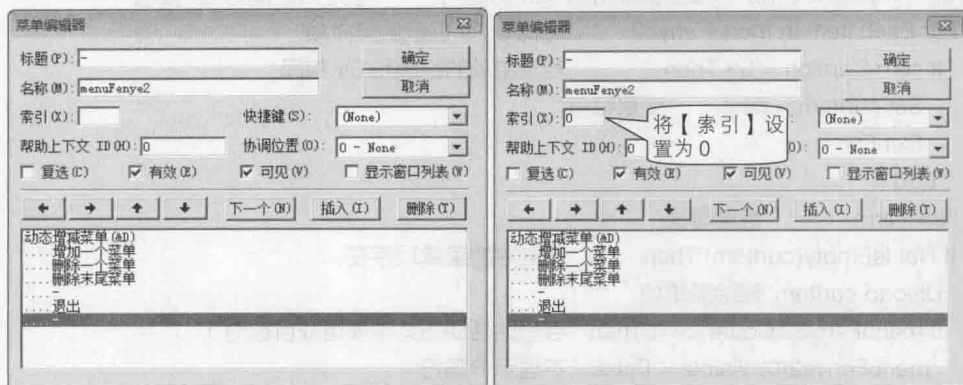
### 【范例 12-3】根据需要自定义一些菜单项，通过对菜单项的代码编写，实现自定义菜单项的特殊功能。

#### 1. 界面设计

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在窗体上的空白处右击，在弹出的快捷菜单中选择【菜单编辑器】菜单项。
- (3) 打开【菜单编辑器】对话框，按照下表所示设计菜单。

菜单等级	标题	名称
1 级菜单	动态增减菜单 (&D)	menuDadm
2 级菜单	增加一个菜单	menuDadmAdd
2 级菜单	删除一个菜单	menuDadmDel
2 级菜单	删除末尾菜单	menuDadmDelLast
2 级菜单	—	menuFenye
2 级菜单	退出	menuDadmExit
2 级菜单	—	menuFenye2

(4) 将菜单项中的最后一项【索引】属性设置为“0”。



(5) 单击【确定】按钮，在 Form1 窗体上双击，在弹出的代码窗口中输入以下代码。

```
01 Private Sub Form_Load()  
02     menuFenye2(0).Visible = False ' 不显示分隔符  
03     menuDadmDel.Enabled = False ' 删除菜单项不可用  
04     menuDadmDelLast.Enabled = False ' 删除最末菜单项不可用  
05 End Sub
```

(6) 在代码窗口的对象下拉列表中选择【menuDadmAdd】对象，在过程下拉列表中选择【Click】事件，并输入以下代码（代码 12-3-1.txt）。

```
01 Private Sub menuDadmAdd_Click()  
02     Dim i As Integer  
03     Dim Str ' 定义变量  
04     i = menuFenye2.UBound ' 变量赋值  
05     Load menuFenye2(i + 1) ' 增加菜单项  
06     Str = InputBox(" 输入所要增加的菜单项的标题 ", " 菜单项标题 ", "MenuName")  
07     ' 通过对话框为 Str 赋值  
08     If Str = "" Then Str = " 新增菜单项，索引号为 " & menuFenye2(i + 1).Index  
09     ' 若 Str 赋值为空，则 Str 为 " 新增菜单项，索引号为 "  
10     menuFenye2(i + 1).Caption = Str ' 将 Str 赋值给菜单项标题  
11     menuFenye2(i + 1).Visible = True ' 增加的菜单项可见  
12     menuFenye2(0).Visible = True ' 显示分隔符  
13     menuDadmDel.Enabled = True ' 删除菜单项可用  
14     menuDadmDelLast.Enabled = True ' 删除最末菜单项可用  
15 End Sub
```

(7) 在代码窗口的对象下拉列表中选择【menuDadmDel】对象，在过程下拉列表中选择【Click】事件，并输入以下代码（代码 12-3-2.txt）。

```
01 Private Sub menuDadmDel_Click()  
02     Dim Str
```

```

03 Dim item, curlItem      '定义变量
04 Str = InputBox(" 输入所要删除的菜单项的标题 ")      '通过对话框为 Str 赋值
05 For Each item In menuFenye2      '寻找与 Str 匹配的菜单项
06     If item.Caption = Str Then      '若菜单项的标题和 Str 相同
07         Set curlItem = item      '变量赋值
08     Exit For
09 End If
10 Next item      '继续寻找
11 If Not IsEmpty(curlItem) Then      '要删除的菜单项存在
12     Unload curlItem '删除菜单项
13     If menuFenye2.Count <= 1 Then '若删除后动态菜单项组项目数为 1
14         menuFenye2(0).Visible = False '不显示分隔符
15         menuDadmDel.Enabled = False '删除菜单项不可用
16         menuDadmDelLast.Enabled = False      '删除最末菜单项不可用
17     End If
18 Else      '要删除的菜单项不存在
19     MsgBox " 没有找到所要删除的菜单项 "      '显示警告窗口
20 End If
21 End Sub

```

(8) 在代码窗口的对象下拉列表中选择【menuDadmDelLast】对象，在过程下拉列表中选择【Click】事件，并输入以下代码（代码 12-3-3.txt）。

```

01 Private Sub menuDadmDelLast_Click()
02     Dim i As Integer '定义变量
03     i = menuFenye2.UBound '变量赋值
04     Unload menuFenye2(i) '删除最末菜单项
05     i = menuFenye2.Ubound      '变量赋值，技巧：以下代码是为了防止将设计阶段设置的菜单项删除
06     If i = 0 Then      '若无可用的菜单项
07         menuFenye2(0).Visible = False '不显示分隔符
08         menuDadmDel.Enabled = False '删除菜单项不可用
09         menuDadmDelLast.Enabled = False      '删除最末菜单项不可用
10     End If
11 End Sub

```

(9) 在代码窗口的对象下拉列表中选择【menuDadmExit】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

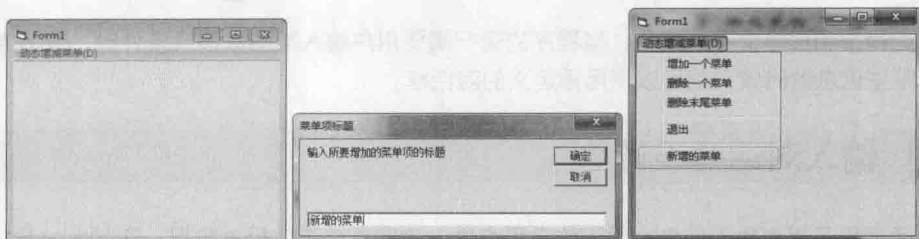
```

01 Private Sub menuDadmExit_Click()
02     Unload Me      '退出程序
03 End Sub

```

## 【运行结果】

保存程序，按快捷键【F5】运行程序。选择相应的菜单项，即可实现增加、删除菜单项的功能。



技巧

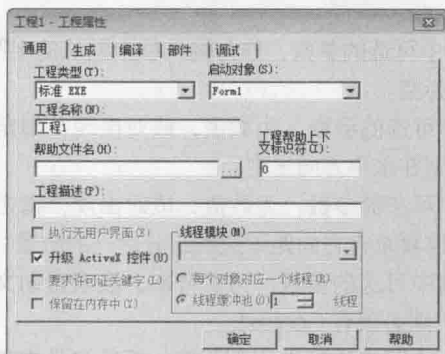
当单击【增加一个菜单】命令后，所弹出的【菜单项标题】对话框是一个输入对话框。关于输入对话框的知识，将在 12.3 小节中介绍。

## 12.2 模式对话框和无模式对话框

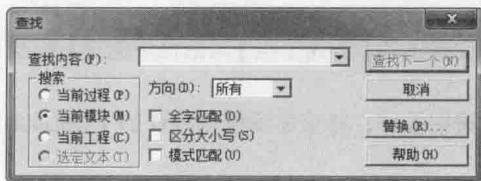


本节视频教学录像：4 分钟

模式对话框和无模式对话框是指对话框是否对软件其他部分的使用有影响。模式对话框用户必须将其关闭或者隐藏才可以继续操作软件的其他部分。在 Visual Basic 中选择【工程】>【工程 1 属性】菜单命令，之后所弹出的【工程 1 - 工程属性】对话框就是一个模式对话框。



无模式对话框的概念和模式对话框相对。无模式对话框的出现并不会影响软件其他部分的使用，用户无需关闭无模式对话框就可以正常使用软件的其他部分。在 Visual Basic 中选择【编辑】>【查找】菜单命令，之后所弹出的【查找】对话框就是一个无模式对话框。



## 12.3 预定义对话框设计



本节视频教学录像: 10 分钟

对话框是用户和程序交互的窗口, 当程序的运行需要用户输入某些数值或条件时, 或者当程序需要向用户提示某些信息的时候, 就可以使用预定义的对话框。

### 12.3.1 输入对话框设计

输入对话框提示用户输入信息, 用于传递用户输入的信息或查找相关数据。在 Visual Basic 中, 常使用 `InputBox()` 函数进行输入对话框的设计。

`InputBox` 函数的语法如下。

```
InputBox(Prompt[,Title][,Default][,Xpos][,Ypos][,Helpfile,Context])
```

`InputBox` 函数中各参数的说明如下。

(1) `Prompt` 是 `InputBox` 函数中必需的参数, 作为输入框中提示信息出现的字符串, 其最大长度约为 1024 个字符, 由所使用字符的宽度决定。



如果 `Prompt` 包含多个行, 则可在各行之间用回车符 (`Chr(13)`)、换行符 (`Chr(10)`) 或回车换行符的组合 (`Chr(13)&Chr(10)`) 来分隔。

**注 意**

(2) `Title` 是 `InputBox` 函数中可选的参数, 作为输入框标题栏中的字符串。若省略该参数, 则在标题栏中显示应用程序名称。

(3) `Default` 是 `InputBox` 函数中可选的参数, 作为输入框中默认的字符串, 在没有其他输入时作为默认值。若省略该参数, 文本框则为空。


(4) `Xpos` 是 `InputBox` 函数中可选的参数, 为数值, 成对出现, 指定输入框的左边与屏幕左边的水平距离。若省略该参数, 输入框则在水平方向居中。

(5) `Ypos` 是 `InputBox` 函数中可选的参数, 为数值, 成对出现, 指定输入框的上边与屏幕上边的距离。若省略该参数, 输入框则在屏幕垂直方向距下边约三分之一的位置。

(6) `Helpfile` 是 `InputBox` 函数中可选的参数, 为字符串, 表示帮助文件, 用该文件为输入框提供上下文相关的帮助。若有 `Helpfile`, 则必须有 `Context`。

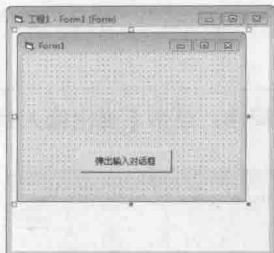
(7) `Context` 是 `InputBox` 函数中可选的参数, 为数值, 帮助文件中某帮助主题的上下文编号。若有 `Context`, 则必须有 `Helpfile`。

**【范例 12-4】**在窗体上添加一个命令按钮, 运行程序后, 当单击该命令按钮时能够弹出一个输入对话框。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在窗体上添加一个命令按钮控件, 将该命令按钮控件的【Caption】属性设计为“弹出输入对话框”。





(3) 在 Form1 窗体上双击【弹出输入对话框】命令按钮，打开代码窗口，输入以下代码。

```
01 Private Sub Command1_Click()  
02   Dim str ' 定义变量  
03   str = InputBox(" 请添加一个用户名 ") ' 使用 InputBox 函数弹出输入对话框  
04 End Sub
```

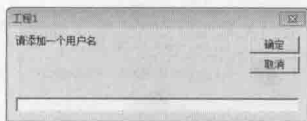


#### 提示

InputBox 函数返回的是一个字符串，若需要得到数值，则需要使用 Val 函数将字符串转换为一个值。

### 【运行结果】

保存程序，按快捷键【F5】运行程序。单击【弹出输入对话框】按钮，弹出一个输入对话框，如图所示。



#### 技巧

在弹出的输入对话框中输入完内容后，如果单击【确定】按钮或按下【Enter】键，InputBox 函数则返回文本框中的内容；如果单击【取消】按钮，此函数则返回一个长度为零的字符串。

### 【拓展训练 12-3】

在 Visual Basic 中，每执行一次 InputBox 函数只能输入一个值，如果需要输入多个值，可以多次调用 InputBox 函数。

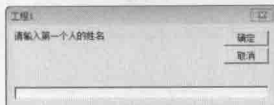
(1) 我们可以将【范例 12-4】中步骤(3)的代码进行如下修改（拓展代码 12-3.txt）。

```
01 Private Sub Command1_Click()  
02   Dim First ' 定义变量  
03   Dim Second ' 定义变量  
04   Dim Third ' 定义变量  
05   First = InputBox(" 请输入第一个人的姓名 ") ' 第 1 次输入数据  
06   Second = InputBox(" 请输入第二个人的姓名 ") ' 第 2 次输入数据
```

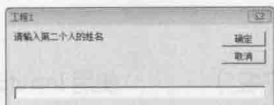


```
07 Third = InputBox(" 请输入第三个人的姓名 ") ' 第 3 次输入数据
08 End Sub
```

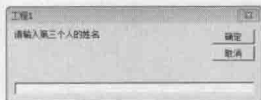
(2) 保存程序后按快捷键【F5】运行程序，单击【弹出输入对话框】按钮，弹出一个输入对话框。



(3) 输入第 1 个人的姓名后，单击【确定】按钮，继续输入。

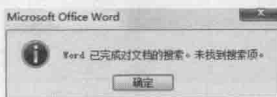


(4) 输入第 2 个人的姓名后，单击【确定】按钮，可以进行第三次姓名的输入。



## 12.3.2 消息对话框设计

消息对话框主要用于向用户显示一些比较重要的信息。例如，我们在使用 Microsoft Word 查找某个文本的时候，如果找不到，就会弹出一个消息对话框。



在 Visual Basic 中，消息对话框主要是使用 MsgBox 函数来实现的，它的主要作用就是弹出一个对话框，在其中显示指定的数据和提示信息。此外，该函数还可以返回用户在此对话框做的选择，并将返回值赋给指定变量。

MsgBox 函数的语法如下。

```
MsgBox(prompt[, buttons] [, title] [, helpfile, context])
```

MsgBox 函数中各参数的说明如下。

(1) prompt 是 MsgBox 函数中必需的参数。字符串表达式，作为显示在对话框中的消息。prompt 的最大长度大约为 1024 个字符，由所用字符的宽度决定。



**提示**

如果 prompt 的内容超过一行，则可在每一行之间用回车符 (Chr(13))、换行符 (Chr(10)) 或是回车与换行符的组合 (Chr(13) & Chr(10)) 将各行分隔开来。

(2) Buttons 是 MsgBox 函数中可选的参数。数值表达式是值的总和，指定显示按钮的数目及形式、使用的图标样式、默认按钮是什么以及消息框的强制回应等。如果省略，buttons 的默认值则为 0。

MsgBox 函数中 Button 主要参数的各个常量值如下表所示。

常量	值	相关说明
vbOKOnly	0	只显示“确定”按钮
VbOKCancel	1	显示“确定”和“取消”按钮
VbAbortRetryIgnore	2	显示“终止”、“重试”和“忽略”按钮
VbYesNoCancel	3	显示“是”、“否”和“取消”按钮
VbYesNo	4	显示“是”和“否”按钮
VbRetryCancel	5	显示“重试”和“取消”按钮
VbCritical	16	显示“关键信息”图标
VbQuestion	32	显示“警告询问”图标
VbExclamation	48	显示“警告消息”图标
VbInformation	64	显示“通知消息”图标



#### 提示

数值 (0 ~ 5) 描述了消息框中显示的按钮的类型与数目, 数值 (16, 32, 48, 64) 描述了图标的样式。

(3) Title 是 MsgBox 函数中可选的参数。在对话框标题栏中显示的字符串表达式。如果省略 title, 则将应用程序名放在标题栏中。

(4) Helpfile 是 MsgBox 函数中可选的参数。字符串表达式, 识别用来向对话框提供上下文相关帮助的帮助文件。如果提供了 helpfile, 则也必须提供 context。

(5) Context 是 MsgBox 函数中可选的参数。数值表达式, 由帮助文件的作者指定给适当的帮助主题的帮助上下文编号。如果提供了 context, 则也必须提供 helpfile。

MsgBox 函数返回值的各个常量值如表所示。


常量	值	相关说明
vbOK	1	确定
vbCancel	2	取消
vbAbort	3	终止
vbRetry	4	重试
vbIgnore	5	忽略
vbYes	6	是
vbNo	7	否



## 技巧

若在消息框中显示【取消】按钮,则按下【Esc】键与单击【取消】按钮的效果相同。

**【范例 12-5】**在窗体上添加一个控件按钮,单击该控件,会弹出一个消息对话框,根据对消息对话框的操作,会再次弹出一个消息对话框,提示读者对消息对话框进行了哪些操作。

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标 ,然后单击【打开】按钮。

(2) 在窗体上添加一个命令按钮控件,将该命令按钮控件的 Caption 属性设计为“单击我看看”。

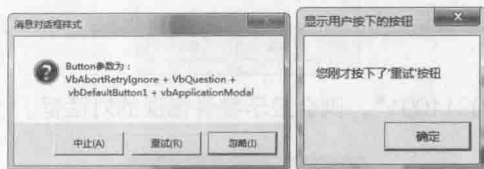


(3) 在 Form1 窗体上双击【单击我看看】命令按钮,打开代码窗口,输入以下代码(代码 12-5.txt)。

```
01 Private Sub Command1_Click()
02     Dim i As Integer
03     i = MsgBox("Button 参数为:" + Chr(13) + Chr(10) + "VbAbortRetryIgnore + VbQuestion +" +
Chr(13) + Chr(10) + " vbDefaultButton1 + vbApplicationModal", _
04     vbAbortRetryIgnore + vbQuestion + vbDefaultButton3 + vbApplicationModal, "消息对话框样式")
05     ' 设置消息对话框样式
06     Select Case i ' 返回用户在对话框中按下的按钮值
07         Case 1
08             MsgBox "您刚才按下了 ' 确定 ' 按钮", , " 显示用户按下的按钮 "
09         Case 2
10             MsgBox "您刚才按下了 ' 取消 ' 按钮", , " 显示用户按下的按钮 "
11         Case 3
12             MsgBox "您刚才按下了 ' 终止 ' 按钮", , " 显示用户按下的按钮 "
13         Case 4
14             MsgBox "您刚才按下了 ' 重试 ' 按钮", , " 显示用户按下的按钮 "
15         Case 5
16             MsgBox "您刚才按下了 ' 忽略 ' 按钮", , " 显示用户按下的按钮 "
17         Case 6
18             MsgBox "您刚才按下了 ' 是 ' 按钮", , " 显示用户按下的按钮 "
19         Case 7
20             MsgBox "您刚才按下了 ' 否 ' 按钮", , " 显示用户按下的按钮 "
21     End Select
22 End Sub
```


## 【运行结果】

保存程序，按快捷键【F5】运行程序。单击【单击我看看】按钮，弹出【消息对话框样式】消息对话框。单击【重试】按钮，弹出【显示用户按下的按钮】消息对话框。



## 【拓展训练 12-4】

在弹出的输入对话框中输入学号，如果学号与所设置的学号匹配，则弹出【登录成功】消息对话框；如果输入的学号与所设置的学号不匹配，则弹出【登录失败】消息对话框。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

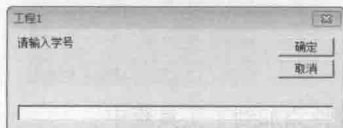
(2) 在窗体上添加一个命令按钮控件，将该按钮控件的 Caption 属性设计为“登录”。



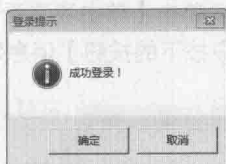
(3) 在 Form1 窗体中双击【登录】按钮，打开代码窗口，输入以下代码（拓展代码 12-4.txt）。

```
01 Private Sub Command1_Click()  
02 Dim str      '定义变量  
03 Dim str1     '定义变量  
04 str = InputBox("请输入学号") '使用 InputBox 函数弹出输入对话框  
05 If str = "0911091" Then '如果学号为 0911091，则显示成功登录  
06     str1 = MsgBox("成功登录！" + Chr(13) + Chr(10), vbInformation + vbOKCancel, "登录提示")  
07     '设置消息对话框样式  
08 Else '否则显示登录失败  
09     str1 = MsgBox("登录失败！" + Chr(13) + Chr(10), vbInformation + vbOKCancel, "登录提示")  
10 End If  
11 End Sub
```

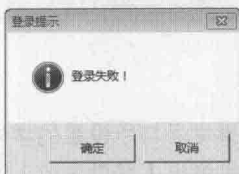
(4) 保存程序，按快捷键【F5】运行程序。单击【登录】按钮，在弹出的输入对话框中输入学号“0911091”，单击【确定】按钮。



(5) 弹出【登录提示】消息对话框。



(6) 如果输入的学号不是“0911091”，则会显示登录错误的对话框。



## 12.4 通用对话框设计



本节视频教学录像：7 分钟

通用对话框旨在提供软件设计时一些有共性的对话框，如打开文件和保存文件、颜色设置和字体设置等。

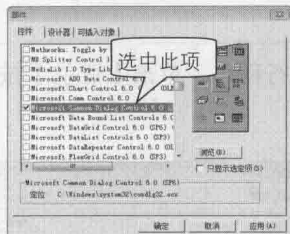
### 12.4.1 添加通用对话框控件

通用对话框是 ActiveX 控件，初次使用要将其添加到工具箱中。

(1) 在工具箱上单击鼠标右键，在弹出的快捷菜单中选择【部件】菜单项。



(2) 弹出【部件】对话框，选中【Microsoft Common Dialog Control 6.0】复选框，然后单击【确定】按钮。



(3) 这样，我们就将通用对话框控件添加到了工具箱中。




通用对话框中提供有 6 种标准对话框，分别是打开对话框、另存为对话框、颜色对话框、字体对话框、打印对话框和帮助对话框。每种对话框都有对应的方法，通过对控件方法的调用，可在应用程序中显示相应的对话框。各对话框的名称、方法和相关描述如表所示。

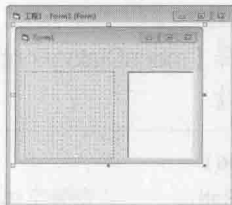
对话框	方法	描述
打开对话框	ShowOpen	用于选取要打开的文件名和路径
另存为对话框	ShowSave	用于指定文件保存的名称和路径
颜色对话框	ShowColor	用于选取或创建要使用的颜色
字体对话框	ShowFont	用于选取所需的字体属性
打印对话框	ShowPrinter	用于选取打印机及对打印机进行设置
帮助对话框	ShowHelp	用于与帮助文件进行关联

## 12.4.2 通用对话框设计实例

### 【范例 12-6】实现在下拉菜单中单击不同的命令，弹出不同的通用对话框。

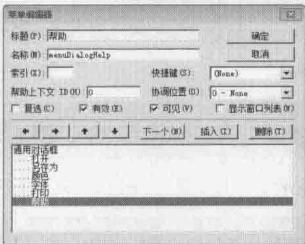
(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 在 Form1 中创建 1 个文本框控件，设置名称属性为“Txt1”，设置 Text 属性为空；再创建 1 个图像控件，设置图像控件的名称属性为“Img1”。

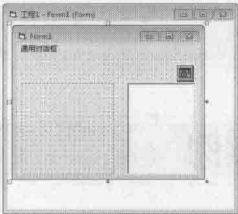


(3) 在 Form1 窗体的空白处右键单击，在弹出的快捷菜单中选择【菜单编辑器】命令，打开【菜单编辑器】对话框，然后按照下表设计菜单。

菜单级别	标题	名称
1 级菜单	通用对话框	menuCommonDialog
2 级菜单	打开	menuDialogOpen
2 级菜单	另存为	menuDialogSaveAs
2 级菜单	颜色	menuDialogColor
2 级菜单	字体	menuDialogFont
2 级菜单	打印	menuDialogPrint
2 级菜单	帮助	menuDialogHelp



(4) 单击【确定】按钮，在 Form1 窗体中添加一个通用对话框控件，然后将所有的控件排列好。



(5) 在 Form1 窗体的空白处右击，在弹出的快捷菜单中选择【查看代码】选项，在弹出的代码窗口的对象下拉列表中选择【menuDialogColor】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

```
01 Private Sub menuDialogColor_Click()  
02     CommonDialog1.ShowColor      ' 显示颜色对话框  
03     Txt1.ForeColor = CommonDialog1.Color ' 将文本框中的字体颜色设置为颜色对话框中选中的颜色  
04 End Sub
```

(6) 在代码窗口的对象下拉列表中选择【menuDialogFont】对象，在过程下拉列表中选择【Click】事件，并输入以下代码（代码 12-6-1.txt）。

```
01 Private Sub menuDialogFont_Click()  
02     CommonDialog1.Flags = cdlCFBoth      ' 安装字体  
03     CommonDialog1.ShowFont      ' 显示字体对话框  
04     Txt1.Font.Bold = CommonDialog1.FontBold ' 指定字体是否加粗
```



```

05 Txt1.Font.Name = CommonDialog1.FontName      '指定字体名称
06 Txt1.Font.Italic = CommonDialog1.FontItalic  '指定字体是否为斜体
07 Txt1.Font.Size = CommonDialog1.FontSize      '指定字体大小
08 Txt1.Font.Strikethrough = CommonDialog1.FontStrikethru '指定字体删除线
09 Txt1.Font.Underline = CommonDialog1.FontUnderline '指定字体下划线
10 End Sub

```

(7) 在代码窗口的对象下拉列表中选择【menuDialogHelp】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

```

01 Private Sub menuDialogHelp_Click()
02 CommonDialog1.HelpFile = "VB6.hlp"          '设置帮助文件
03 CommonDialog1.HelpCommand = cdlHelpContents '设置要显示的帮助主题
04 CommonDialog1.ShowHelp                      '显示帮助对话框
05 End Sub

```

(8) 在代码窗口的对象下拉列表中选择【menuDialogOpen】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

```

01 Private Sub menuDialogOpen_Click()
02 CommonDialog1.Filter = "图形文件 (*.jpg;*.gif;*.bmp;*.lco;*.wmf)|(*.jpg;*.gif;*.bmp;*.lco;*.wmf)
" '指定打开文件的扩展名
03 CommonDialog1.ShowOpen          '显示打开对话框
04 Img1.Stretch = True             '自动调节图片的大小
05 Img1.Picture = LoadPicture(CommonDialog1.FileName)
'按照打开对话框指定的文件名和路径打开图片
06 End Sub

```

(9) 在代码窗口的对象下拉列表中选择【menuDialogPrint】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

```

01 Private Sub menuDialogPrint_Click()
02 CommonDialog1.ShowPrinter      '显示打印对话框
03 End Sub

```

(10) 在代码窗口的对象下拉列表中选择【menuDialogSaveAs】对象，在过程下拉列表中选择【Click】事件，并输入以下代码。

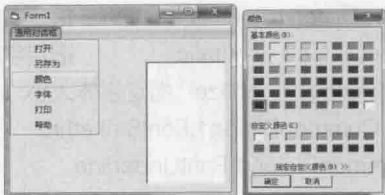
```

01 Private Sub menuDialogSaveAs_Click()
02 CommonDialog1.ShowSave        '显示另存为对话框
03 End Sub

```

## 【运行结果】

保存程序，按快捷键【F5】运行程序。单击【通用对话框】菜单按钮，在弹出的下拉菜单中选择【颜色】菜单项，弹出【颜色】对话框。



## 12.5 高手点拨



本节视频教学录像: 4 分钟

通过本章的学习, 我们知道 Visual Basic 6.0 给用户三种菜单: 窗体控制菜单、下拉菜单与快捷菜单。窗体控制菜单由窗体的 ControlBox 属性控制, 下拉菜单和快捷菜单则可以由菜单控件 (Menu) 方便地设计出来。为了创建 Menu 控件, 要使用“菜单编辑器”。

对话框主要可以分为三类: 系统预定义对话框、用户自定义对话框和通用对话框。根据对话框是否要求用户必须作出响应, 可以分为模式对话框和无模式对话框。下面给出了一些通用对话框的常见属性。

**对话框标题 (DialogTitle):** 用来给出对话框的标题内容 (缺省值为“打开”)。

**文件名称 (FileName):** 用来给出对话框中“文件名”区中文件名的初始值。用户在对话框中的文件列表框中选中的文件名也放在此属性中, 即用它能设置和返回文件名。

**初始化路径 (InitDir):** 用来设置和返回目录名。若不设置该属性, 系统将显示当前目录, 这里先设为 C:\。

**过滤器 (Filter):** 用来指定在对话框中的文件列表框中列出的文件类型。其格式为: 描述符 1| 筛选符 1| 描述符 2| 筛选符 2|……。设为 AllFiles|\*. \*|Frm 文件 \*.Frm|VB 文件 \*.Vbp。

**标志 (Flags):** 用来设置对话框中的一些选项。

**缺省扩展名 (DefaultExt):** 如果用户输入的文件名不带扩展名, 则自动将此缺省扩展名作为其扩展名。

**文件最大长度 (MaxFileSize):** 用来指出文件名最大长度 (范围为 1~2048)。

**过滤器索引 (FilterIndex):** 指出缺省的筛选符。排序为 1, 2, 3……

**取消引发错误 (CancelError):** 这是一个选择钮。选中后, 当单击打开文件对话框的取消按钮以关闭一个对话框时, 系统将显示一个报错信息的信息框。

## 12.6 实战练习

### 一、思考题

1. 对话框从整体上来分主要分哪两类? 试简述它们的区别。
2. 下拉式菜单和弹出式菜单主要应用于哪些方面?

### 二、操作题

设计一个下拉式菜单, 主要用于登录, 当登录成功后, 提示登录成功, 登录失败后, 则要提示登录失败。

# 第13章



本章视频教学录像：32 分钟

## 编程错误终结者——程序调试与错误处理

在进行程序设计时，出现错误是不可避免的，一个应用程序在不断清除错误的过程中会变得越来越稳定。程序调试与错误处理在软件开发的所有环节中贯彻始终，其工作质量将直接影响程序的最终使用价值。

在 Visual Basic 6.0 中提供有功能强大的调试工具，可以很方便地对程序进行调试。本章讲解 Visual Basic 6.0 中的常见错误类型、调试工具的使用以及常用的调试方法。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉 Visual Basic 6.0 程序中的错误类型
- ☐ 了解程序工作状态
- ☐ 掌握调试工具的使用
- ☐ 掌握常用的调试方法
- ☐ 了解 Err 对象
- ☐ 了解 On Error GoTo 语句
- ☐ 了解 Resume 语句

## 13.1 Visual Basic 6.0 程序中的错误类型



本节视频教学录像：5 分钟

在 Visual Basic 6.0 中，无论多么仔细地编写代码，都不可避免地会出现错误。严重的错误可能会使应用程序不再对命令做出响应，这时可能要重新启动应用程序，从而造成已经完成但尚未存储的代码丢失，对用户造成不可预见的损失。

为了排除这些错误，可以手动修改，但是当程序比较庞大的时候，用人工方式去消除错误就会变得非常困难并且非常低效。Visual Basic 6.0 提供有一些工具可以帮助我们更加自动化地查找、排除这些错误。

在使用 Visual Basic 编写程序和运行程序的时候，最有可能发生的错误有以下 3 种。

### 13.1.1 语法错误

语法错误是由于程序代码语句中出现了不符合 Visual Basic 语法规则的语句所引起的，例如括号不匹配。

或者在 If...Then...End If 语句中少了 End If 语句，就会提示缺少表达式。

或者过程结束时多了 End Sub 语句等。



这类错误往往是用户在编写程序代码时发生的，Visual Basic 编辑器会自动检查出来，告诉用户建议的改正方法，或者自动改正某些基本的语法错误。

### 13.1.2 逻辑错误

逻辑错误是指程序代码上没有语法错误，但由于程序的结构或算法存在问题，使得程序的运行结果与编写程序时的初衷相悖。

例如：

变量的作用域范围没有设置正确；循环条件不正确，造成死循环等。对于这类错误，一般要借助调试工具来找出程序的错误点和错误原因，以便解决。

### 13.1.3 运行时错误

没有语法错误和逻辑错误并不是就万事大吉了，软件在运行的时候各种突发状况也会导致错误。运行时错误是指发生在程序的运行环境发生改变后，往往得不到正确的运行结果的错误。例如，当出现除数为 0 时，会产生“除数为零”的错误提示；当要对磁盘文件进行操作时，计算机突然断电。对于这类

错误，在 Visual Basic 中采用错误陷阱的手段，帮助用户编制错误处理程序，从而可在最大程度上避免错误的出现。

## 13.2 程序工作状态

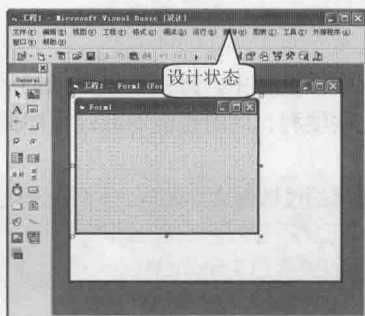


本节视频教学录像：2 分钟

在 Visual Basic 中，程序有 3 种工作状态：设计状态、运行状态和中断状态。

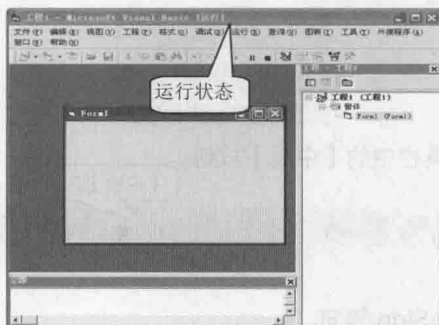
### 13.2.1 设计状态

进行编写程序代码、设计程序界面、为窗体添加控件等操作时，程序处于设计状态。



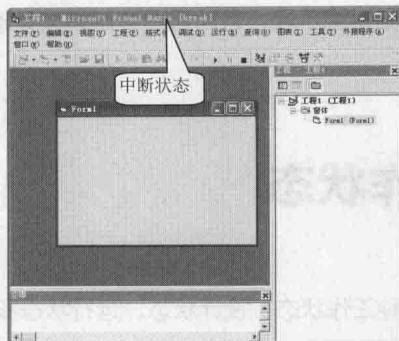
### 13.2.2 运行状态

设计好程序后，选择【运行】菜单下的【启动】菜单项或者直接按快捷键【F5】，程序开始运行，此时就从设计状态变成了运行状态。



### 13.2.3 中断状态

在程序运行的任意时刻，选择【运行】菜单下的【中断】菜单项或者直接按快捷键【Ctrl+Break】，程序将转换到中断状态。我们在进行错误处理和程序调试的时候，都要切换到中断状态。



## 13.3 程序调试



本节视频教学录像: 14 分钟

程序出错是不可避免的, 在 Visual Basic 中, 提供有一系列的运行和调试工具。掌握并熟练使用这些工具, 将会为程序的纠错提供很大的便利, 同时也能够减轻程序员的工作负担。



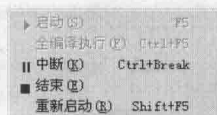
**提示**

在应用程序中查找并修改错误的过程称之为调试。为了分析应用程序的操作方式, Visual Basic 提供有几种工具。这些调试工具不但对查出错误根源特别有用, 而且还能用来尝试着改变应用程序, 或用来了解其他应用程序的工作方式。

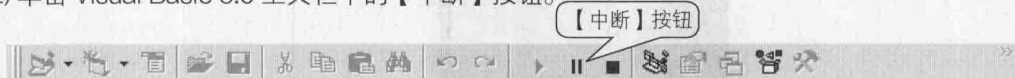
### 13.3.1 使程序进入中断状态

当我们要对程序进行调试和纠错的时候, 首先要使程序进入中断状态。在 Visual Basic 中程序进入中断状态的方式有以下 5 种。

(1) 选择 Visual Basic 6.0 的【运行】菜单下的【中断】菜单项。



(2) 单击 Visual Basic 6.0 工具栏中的【中断】按钮。



(3) 按快捷键【Ctrl+Break】。

(4) 在应用程序中设置断点和 Stop 语句。

(5) 在运行时产生的一些错误使得程序自动进入中断状态。

例如:

```
01 Private Sub Form_Load()
02   Dim Str As Integer      ' 定义 Str 为 Integer 型变量
03   Str = "Hello"          ' 为 Str 赋值
```

04 MsgBox Str '弹出窗口显示 Str 的值

05 End Sub

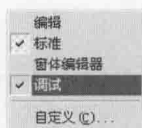
运行上面这段代码，程序将自动中断程序并弹出出错对话框。



## 13.3.2 调试工具

对于语法错误和运行时错误，系统可以自动帮助我们检测出来，但对逻辑错误，系统是无能为力的。如果仅靠自己阅读代码来查找错误，无疑是非常低效的。为了解决这类问题，Visual Basic 提供一些非常专业的调试工具。

Visual Basic 提供有一个调试工具栏，可以使程序调试更加方便。在【标准】工具栏上右击，在弹出的快捷菜单中选择【调试】菜单项，即可打开【调试】工具栏。通过单击该工具栏中的各个按钮就可以调用相应的调试工具。



【调试】工具栏中各个按钮的名称和功能如表所示。

按钮名称	按钮图标	按钮功能
切换断点		在代码中设置一个断点，程序执行到该语句时将中断，以便进行调试；将光标定位至断点语句处，再次单击该按钮，将取消断点设置
逐语句		执行下一条语句，并跟踪到过程中
逐过程		执行下一条语句，但不跟踪到过程中
跳出		执行当前过程中剩余的代码，并在调用当前过程的下一条语句处中断执行
本地窗口		显示当前过程中所有变量的值
立即窗口		当处于中断模式时，允许执行代码或查询变量当前值
监视窗口		显示选定表达式的值
快速监视		当处于中断模式时，列出选定表达式的当前值
调用堆栈		当处于中断模式时，用一个对话框来显示所有已被调用但尚未完成运行的过程

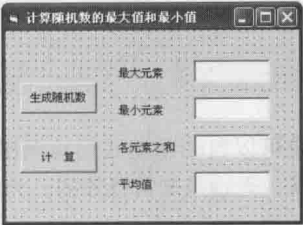



下面通过一个实例来学习【调试】工具栏中按钮的使用方法。

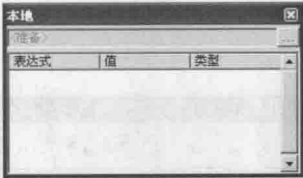
【范例 13-1】应用调试工具栏，分别使用本地窗口、立即窗口和监视窗口监测程序运行情况。


第 1 步：本地窗口

(1) 打开随书光盘中的“Sample\ch13\范例 13-1\计算随机数的最大值和最小值.vbp”文件，弹出相应的对话框。

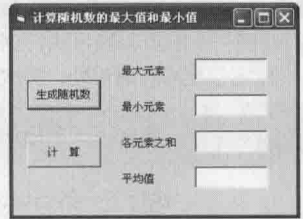


(2) 单击【调试】工具栏中的【本地窗口】按钮，弹出【本地】窗口。【本地】窗口用于显示当前过程中所有变量的值。由于当前程序没有运行，所以【本地】窗口暂时没有任何内容。

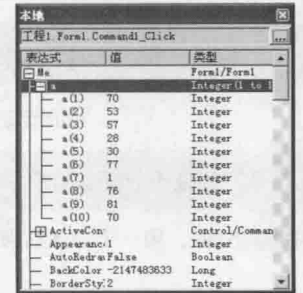



(3) 单击【调试】工具栏中的【逐语句】按钮，应用程序将进入逐句运行状态。

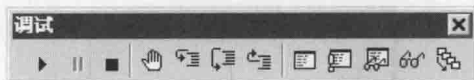
(4) 单击程序界面中的【生成随机数】按钮，程序开始产生随机数。




(5) 按快捷键【F8】，每按一下，程序就向前执行一句，同时【本地】窗口中将显示执行此句后的变量值。



(6) 单击【调试】工具栏中的【结束】按钮，结束程序的运行。



## 第 2 步：立即窗口

(1) 单击【调试】工具栏中的【立即窗口】按钮，打开【立即】窗口。

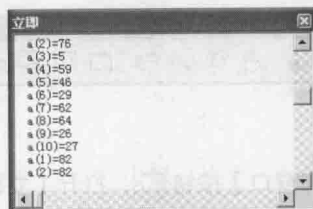



(2)【立即】窗口处于中断模式时，允许执行代码或所查询变量当前值。在使用时需要从应用程序中输出信息到【立即】窗口，使用方法是使用 Debug 语句加上 Print。所以为了使实例文件能够将变量值输出到【立即】窗口，在源代码的 Command1\_Click() 过程中添加一条语句：Debug.Print "a(" & i & ")=" & a(i)。修改后的 Command1\_Click() 过程代码如下（代码 13-1.txt）。

```
01 Private Sub Command1_Click()
02 Dim i As Integer
03 Cls
'清空窗体内容
04 Text1.Text = ""
'为 Text1 文本赋空值
05 Text2.Text = ""
'为 Text2 文本赋空值
06 Text3.Text = ""
'为 Text3 文本赋空值
07 Text4.Text = ""
'为 Text4 文本赋空值
08 Print "产生的十个随机数为"
'输出字符
09 For i = 1 To 10
10 a(i) = Int(Rnd * 100)
'产生随机数
11 Print a(i);
'输出
12 Debug.Print "a(" & i & ")=" & a(i)
'输出到立即窗口
13 Next i
14 End Sub
```


(3) 按快捷键【F5】运行程序，在程序界面中单击【生成随机数】按钮，可以看到，变量 a 的值已

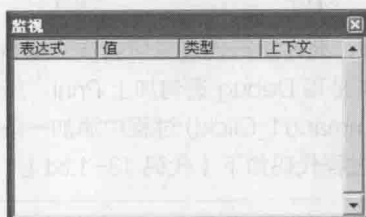
经被输出到了【立即】窗口中。



(4) 单击【调试】工具栏中的【结束】按钮，结束程序的运行。

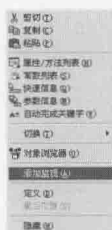
### 第3步：监视窗口

(1) 单击【调试】工具栏中的【监视窗口】按钮，打开监视窗口。




(2) 【监视】窗口用于显示选定表达式的值。在使用【监视】窗口时，需要在程序处于中断状态或设计状态时添加监视表达式。

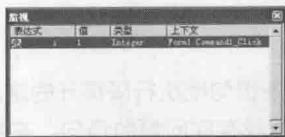
(3) 在原有 Command1\_Click() (产生随机数) 过程的源代码中选中变量 i，单击鼠标右键，在弹出的快捷菜单中选择【添加监视】菜单项。



(4) 在弹出的【添加监视】对话框中，将【过程】设置为【Command1\_Click】，将【模块】设置为【Form1】，将【监视类型】设置为【当监视值改变时中断】，然后单击【确定】按钮。



(5) 单击【调试】工具栏中的【运行】按钮，应用程序进入运行状态。单击程序界面的【生成随机数】按钮后，由于监视表达式 i 在循环中不断变化，所以在 Command1\_Click() 过程运行中不断产生中断，以便在代码窗口中观察产生随机数数组 a 的变量值。



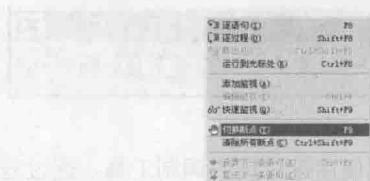
### 13.3.3 调试方法


本小节介绍 Visual Basic 中主要的两种调试方法：断点调试和跟踪调试。

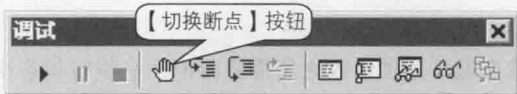
#### 1. 断点调试

在设计模式时，根据需要可在应用程序中设置一个或多个断点，当应用程序运行到断点时将暂停运行，并且进入中断模式，此时可以检查变量或参数的当前值，从而找出错误的原因。断点可以在设计时预先设置好，也可以在程序运行的过程中进入中断状态之后再加入。在代码窗口中设置断点的方法有以下 4 种。

(1) 将光标移动到要设置断点的位置，选择【调试】>【切换断点】菜单命令。

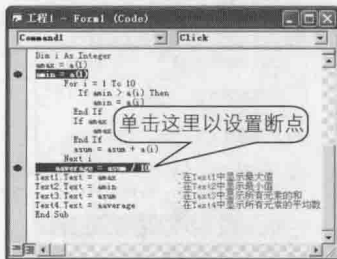


(2) 将光标移动到要设置断点的位置，单击【调试】工具栏中的【切换断点】按钮.



(3) 将光标移动到要设置断点的位置，按快捷键【F9】。

(4) 将光标移动到要设置断点的位置，单击语句左端的阴影区。某条语句被设置断点后，将被突出显示出来，如图所示。



按照设置断点的方法，在已经设置断点的语句中重复操作一次即可取消断点。当程序运行到设置断点的语句时，将会自动进入中断状态以供调试。

#### 2. 跟踪调试

当一个应用程序的运行得不到正确结果，编译器又没有报错，往往是因为程序中存在逻辑错误所致。一般情况下，当代码很多的时候，往往很难知道错误的具体位置，这时应首先估计出错的范围，再设置多个断点，以将存在问题的代码行隔离出来，然后用逐语句、逐过程和跳出等调试工具，跟踪观察

每一个语句的执行效果,找出问题所在。

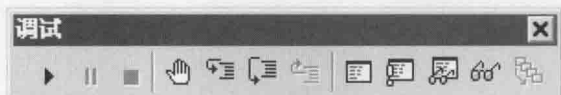
### (1) 逐行执行代码

逐行执行代码就是在中断状态下逐条语句地执行隔离开的这段代码,通过查看应用程序的接口或者是 Visual Basic 6.0 的【立即】窗口来查找存在问题的语句。在断点状态下进入中断状态后,逐行执行代码常用的方法有以下 3 种。

#### ① 选择【调试】>【逐语句】菜单命令。



#### ② 单击调试工具栏中的【逐语句】按钮.



#### ③ 按快捷键【F8】。

如果觉得逐语句调试速度太慢,还可以使用逐过程的调制工具。逐过程调试是以过程或者函数为单位进行调试的。逐过程调试的方法也使用菜单命令和工具栏命令,但是最简便的方法是使用快捷键【Ctrl+F8】。

### (2) 设置要执行的下一条语句

在调试应用程序时,经常在做了某项修改之后需要再返回去执行某条语句,以验证此项修改是否正确,这就需要设置下一条要执行的语句。设置下一条需要执行的语句的常用方法如下。

① 在中断状态下,将光标移到需要重新执行的代码行,然后选择【调试】>【设置下一条语句】菜单命令。



#### ② 在中断状态下,将光标移到需要重新执行的代码行,然后直接按快捷键【Ctrl+F9】。



#### 提示

如果已经执行错误处理程序中的代码,但不能确定要在哪里恢复运行,可以用【调试】菜单中的【显示下一条语句】菜单命令把光标设置到下一个要执行的行。

## 13.4 除虫行动—— Visual Basic 6.0 中的错误处理



本节视频教学录像：9 分钟

程序的设计不可能做到绝对完美，在实际运行过程中不可避免地会出现一些错误与问题，而且有些错误是我们无法预料或者控制的，例如计算机突然断电产生的错误。因此当出现错误的时候，我们需要知道如何去处理它。

### 13.4.1 Err 对象

Visual Basic 提供了一个用于错误捕捉的 Err 对象。当错误发生时，Err 对象可用于得到所发生的错误的错误号、错误描述等，我们可以根据这些错误信息采取相应的解决方法。下表是常见错误代码及其所代表的信息。

错误代码	所代表信息	错误代码	所代表信息
5	无效的过程调用	6	溢出
7	内存超出	9	下标越界
10	为固定数组	11	除以零
13	类型不匹配	14	字符串空间不足
16	表达式太复杂	18	发生用户中断
20	无错误恢复	28	超出栈空间
35	没有定义子程序、函数或属性	47	DLL 程序客户太多
48	调用 DLL 时的错误	49	错误的 DLL 调用约定
51	内部错误	52	文件名错误
53	文件找不到	55	文件已打开
57	I/O 设备错误	61	磁盘已满
62	输入超过文件尾	63	记录号错误
67	文件太多	68	设备没有准备好
71	磁盘尚未就绪	75	路径 / 文件访问错误
76	找不到路径	92	For 循环没有被初始化
93	非法模式字符串	94	非法使用 Null
380	无效的属性值	382	属性设置不能在运行时完成

### 13.4.2 On Error GoTo 语句

为了发生错误之后能及时处理,在应用程序中常常使用错误陷阱的方法。On Error GoTo 语句用于启动一个错误处理程序,同时可以指定该子程序在一个过程中的位置,也可用来禁止一个错误处理程序。

设置错误陷阱的命令一般放在过程的开始部分,主要有以下3种用法。

#### 1. On Error Goto 标号

当程序出现错误时,转到由标号指定的错误处理程序开始运行。

#### 2. On Error Resume Next

当程序出现错误时,转到紧接着错误语句的后一个语句开始运行,即跳过出错语句。

#### 3. On Error Goto 0

禁止当前过程中任何已启动的错误处理程序,即关闭错误陷阱。如果程序正常运行,将不执行错误处理程序,因此一般在代码的最后通过“Exit Sub”语句跳过错误处理程序,结束本过程运行。

### 13.4.3 Resume 语句

在设置了错误陷阱并捕捉到错误信息后,应用程序将转而执行错误处理程序。通常情况下,使用 Resume 命令将引起错误的语句再重新操作一遍。

Resume 命令有3种用法。

(1) Resume 0 或 Resume : 从产生错误的语句恢复运行。


(2) Resume Next : 跳过产生错误的语句,运行下面的语句。

(3) Resume 标号: 从标号指定的语句开始运行。

### 13.4.4 错误处理实例

在部分程序中,需要读取特定的文件,如果文件不存在,就会出现运行错误。下面的实例将使用本章所讲的错误处理方法解决这个问题。

#### 【范例 13-2】错误处理应用实例,调试程序的运行。

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标,然后单击【打开】按钮。

(2) 把 Form1 的 Caption 属性设置为“学生信息管理”。





(3)在窗体中添加两个命令按钮（Command Button）控件、1 个含有 4 个文本框（TextBox）控件的文本框控件组（采用复制控件再粘贴的方法）和 1 个含有 4 个标签（Label）控件的标签控件组，按照下表所示设置各控件的属性，并将控件排列成下图所示的效果。

控件名称	Caption / Index	控件作用
Command1	保存	用于向顺序文件“花名册”中添加记录
Command2	显示	用于读取顺序文件“花名册”中的记录
Label1(0)	姓名	提示记录中字段内容的意义
Label1(1)	班级	提示记录中字段内容的意义
Label1(2)	学号	提示记录中字段内容的意义
Label1(3)	专业	提示记录中字段内容的意义
Text1(0)	0	显示记录中字段的内容
Text1(1)	1	显示记录中字段的内容
Text1(2)	2	显示记录中字段的内容
Text1(3)	3	显示记录中字段的内容



(4) 双击 Command1 按钮控件，在弹出的代码窗口中输入以下代码（代码 13-2-1.txt）。

```

01 Private k As Integer      ' 设置一个记录号变量
02 Dim i As Integer  ' 存储顺序文件的程序
03 Private Sub Command1_Click()
04 Dim ans As String
05 On Error GoTo Line1      ' 如果出错，转到此位置
06 For i = 0 To 3      ' 检验信息的完整性
07 If Text1(i).Text = "" Then
08   ans = MsgBox(" 请填写完整的花名册信息！ ", vbOKOnly, " 警告 ")
09   If ans = 1 Then Exit Sub
10 End If
11 Next i

```

```

12 Open App.Path & "/ 花名册 .txt" For Append As #1 '使用 Append 模式, 用于向顺序文件中
添加记录
13 For i = 0 To 3 '保存花名册信息
14     Print #1, Text1(i).Text
15 Next i
16 Close 1
17 Command2.Caption = "无记录。单击, 从头重新显示!"
18 For i = 0 To 3 '保存后清除文本框中的内容
19     Text1(i).Text = "" '清空文本框内容
20 Next i
21 Exit Sub
22 Line1: '判断文件是否已经打开, 正在读取
23 MsgBox "正在读取文件! 当另一按钮为“无记录。单击, 从头重新显示!”时, 才能向文件中添
加记录。", vbOKOnly, "警告"
24 End Sub

```

(5) 双击 Command2 按钮控件, 在弹出的代码窗口中输入以下代码 (代码 13-2-2.txt)。

```

01 Private Sub Command2_Click() '读取顺序文件的程序
02 Dim str As String
03 On Error GoTo myerror '如果出错, 转到此位置
04 k = k + 1
05 Open App.Path & "/ 花名册 .txt" For Input As #1
06 Command2.Caption = "下一条"
07 For i = 0 To 3 '开始读取花名册第 1 条信息
08     Line Input #1, str
09     Text1(i).Text = str
10 Next i
11 Print "第"; k; "条记录"
12 Exit Sub
13 myerror:
14 If Err.Number = 53 Then
15     MsgBox "无法读取花名册 .txt 文件, 请确认文件是否存在。"
16     Command2.Caption = "显示"
17     k = 0
18     Exit Sub
19 ElseIf Err.Number = 55 Then '读取每一条花名册信息
20     If EOF(1) = False Then
21         For i = 0 To 3
22             Line Input #1, str
23             Text1(i).Text = str
24         Next i
25         Print "第"; k; "条记录"

```

```

26 Exit Sub
27 Else ' 读取完花名册后, 清除记录显示
28     For i = 0 To 3
29         Text1(i).Text = ""
30     Next i
31     Cls ' 清除记录数显示
32     Command2.Caption = "无记录。单击, 从头重新显示!"
33     Close 1
34     k = 0
35     Exit Sub
36 End If
37 End If
38 End Sub

```

## 【代码详解】

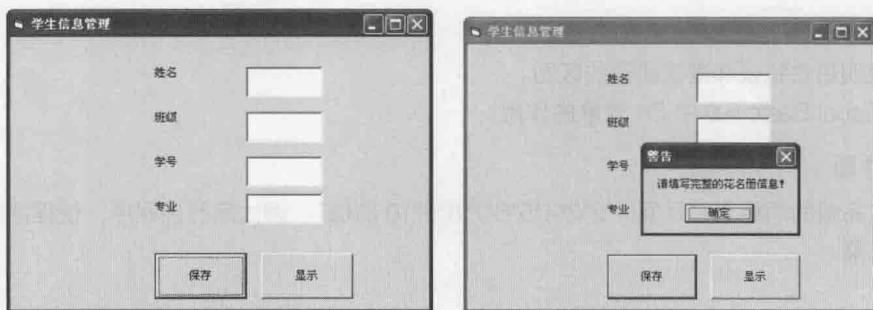
这段程序分别为 Command1 按钮和 Command2 按钮添加了错误处理过程。

在第(4)步代码中, On Error GoTo Line1 语句使得如果 Command1 按钮的点击事件 Command1\_Click 出现错误, 则跳转到 Line1 标号指示的位置, 弹出提示对话框显示“正在读取文件! 当另一按钮为“无记录。单击, 从头重新显示!”时, 才能向文件中添加记录。”, 并且用 End Sub 语句结束过程。

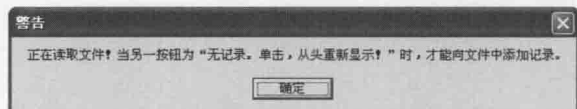
在第(5)步代码中, 如果错误代码为 53, 即“文件找不到”, 则弹出提示对话框显示“无法读取花名册.txt 文件, 请确认文件是否存在。”; 如果错误代码为 55, 即“文件已打开”, 则开始显示记录信息。

## 【运行结果】

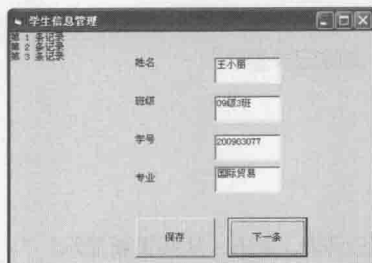
将“Sample\ch09\范例 13-2\花名册.txt”文件复制到工程所在文件夹下, 按快捷键【F5】运行程序。如果在单击程序界面上的【保存】按钮前没有填写完整信息, 系统会提示填写完整信息。



如果在单击【显示】按钮后立刻单击【保存】按钮, 系统会提示错误信息。



正确填写信息后, 单击【保存】按钮, 即可保存到“花名册.txt”中。单击【显示】按钮, 信息即可在文本框中显示, 并在窗体左侧显示记录号。



## 13.5 高手点拨



本节视频教学录像: 2 分钟

在编写包含多个模块的大型 Visual Basic 应用程序时, 错误处理代码可能变得相当复杂。请记住下面这些准则。

在进行代码的调试时, 使用 Err 对象的 Raise 方法, 可在所有错误处理程序中重新生成错误, 以解决处理程序中无针对某特定错误的代码的情况。这使得应用程序可试着沿调用列表在其他错误处理例程中更正错误。另外, 它可确保在发生代码不能处理的错误时, Visual Basic 会显示错误信息。该技巧可帮助你发现那些未被充分处理的错误。如果需要在完成错误处理后清除 Err 对象, 使用 Clear 方法, 这对于使用 On Error Resume Next 进行直接错误处理是有必要的。每当执行任意类型的 Resume 语句、Exit Sub、Exit Function、Exit Property 或任何 On Error 语句时, Visual Basic 就会自动调用 Clear 方法。

## 13.6 实战练习

### 一、思考题

1. 简要说明语法错误和逻辑错误的区别。
2. 简述 Visual Basic 6.0 中 Err 对象的作用。

### 二、操作题

程序预定完成的功能为“计算  $1*2*3*4*5*6*7*8*9*10$  的值”。调试运行该程序, 使程序能正确运行, 并得出正确结果。

# 第3篇

## 高级应用

本篇介绍数据库系统与 SQL 语言基础，Visual Basic 6.0 中的数据库编程、数据报表、API 编程、网络编程、图形图像与多媒体编程、文件系统编程以及应用程序打包等。通过本篇的学习，读者能掌握 Visual Basic 的高级应用。

第 14 章 进入数据仓库——数据库与 SQL 语言基础

第 15 章 Visual Basic 与数据库的联合——Visual Basic 6.0 中的数据库编程

第 16 章 Visual Basic 6.0 生成的报表——数据报表

第 17 章 Visual Basic 编程的核心——API 编程

第 18 章 Visual Basic 中的网络世界——网络编程

第 19 章 Visual Basic 中的视听——图形图像与多媒体编程

第 20 章 用 VB 操纵文件——文件系统编程

第 21 章 让你的程序去旅行——应用程序打包

# 第14章



本章视频教学录像：40 分钟

## 进入数据仓库——数据库与 SQL 语言基础

SQL 是一种结构化的查询语言，利用 SQL 语句可以很方便地向数据库中添加数据、修改数据、删除数据和查询数据。本章讲述 SQL 语言的基本应用，从而为编写数据库应用程序奠定坚实的基础。可以将 SQL 语言融入到 Visual Basic 高级语言中使用，这样便可利用高级语言的过程结构弥补 SQL 语言在实现复杂应用方面的不足。结构化查询语言（Structured Query Language，SQL）是最重要的关系数据库操作语言，并且它的影响已经超出数据库领域，得到其他领域的重视和采用。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉数据库的基本概念
- ☐ 掌握常用 SQL 语句
- ☐ 掌握 Select 语句的使用
- ☐ 熟悉 SQL 中的常用函数
- ☐ 掌握利用 SQL 语言修改表数据的方法

## 14.1 数据库基本概念



本节视频教学录像：6 分钟

从字面来看，可以将数据库看做是存放数据的仓库。而从本质来看，数据库是一些数据的集合，用户不仅可以方便地存储大量的数据信息，而且也可以方便地查找信息。从广义上来讲，任何包含数据信息的载体或窗口都可以被称为数据库，比如上学的时候所用的笔记本电脑，里面记载了老师所讲的内容，就可以把它当作一个数据库。当然，这样的数据库没有什么规律可循，它只是一个集合而已。简单来说本身可视为电子化的文件柜——存储电子文件的处所，用户可以对文件中的数据进行新增、截取、更新、删除等操作。

下面对数据库中的基本对象进行简单的介绍。

### 1. 表 (TABLE)

一个关系对应一个表，关系就是表中数据之间存在的联系。从直观的角度看，表就是一个二维的具有数据的表格，表的一行称为一个记录，表的一列称为一个字段。

学号	姓名	年龄	性别	班级
11001	张三	20	男	01101
11002	李四	21	女	01101
11003	王五	22	男	01101

### 2. 索引 (INDEX)

索引用于快速访问表，通过索引可以不必扫描整张表就能够查询到数据，从而优化查询速度。一个表中可以有若干个索引。例如，在一张表的若干个数据中，若想查询第 33 行的数据，在没有索引的情况下，一般是重头开始查询，直到查询到第 33 行为止。有了索引以后，就可以通过索引直接找到第 33 行的数据，而无须从头开始查询。

### 3. 主键 (PRIMARY KEY)

表中的每一条记录必须满足唯一性，这样才能确保查找到所需要的记录。一个记录就是表中的一行数据。主键的作用就是确保这种唯一性。在上表中，“学号”列可以是主键，因为它是唯一的，每个“学号”对应一个人。通过“学号”可以找到每个人，而不会引起歧义。表中一般都要定义一个主键，但不是强制的。

### 4. 外键 (FOREIGN KEY)

如果有两个表，并且这两个表的主键相同，这两个表之间就可以通过相同的主键建立关系，从而可以在两个表之间查询数据。一个表的主键相对于另一个表就是外键，外键说明当前表中的某项是引用于另一个表。



#### 提示

主键和外键是把多个表组织为一个有效的关系数据库的粘合剂。主键和外键的设计对物理数据库的性能和可用性都有着决定性的影响。必须将数据库模式从理论上的逻辑设计转换为实际的物理设计。而主键和外键的结构是这个设计过程的症结所在。一旦将所设计的数据库用于了生产环境，就很难对这些键进行修改，所以在开发阶段就设计好主键和外键就是非常必要和值得的。



## 14.2 SQL 应用



本节视频教学录像：3 分钟

SQL 是一种可以与数据库交互的结构化查询语言，是一种对数据库中的数据进行组织、管理和检索的工具。

### 14.2.1 SQL 语言的特点

SQL 虽然被称为结构化的查询语言，但实际上它可以实现数据查询、定义、添加、删除和修改等全部功能。它把关系型数据库的数据定义语言 DDL、数据操纵语言 DML 和数据控制语言 DCL (Data Contr0L Language) 集为一体，统一在一个语言中，体现了其一体化的特点。

从 SQL 的角度来说，表中的记录没有顺序，所以 SQL 语句不能按照某种特定的顺序来取出记录，只能按查询条件来读取记录，SQL 会确定实现查询的最佳方法，体现了其高度非过程化的特点。

### 14.2.2 常用 SQL 语句简介

SQL 语言以其功能强大、结构简单、易学易用等特点在关系型数据库管理系统中得到了最为广泛的应用。在 SQL 语言中最常用的语句有 Select、Insert、Update、Delete、Create、Drop 等，下表列出了它们的主要功能。

语句	功能
Select	从一个表或多个表中检索列和行
Insert	向一个表中增加行
Update	更新表中已存在的行的某几列
Delete	从一个表中删除行
Create	按特定的表模式创建一个新表
Drop	删除一张表

根据表中所描述的 SQL 语句功能来看，可以进行如下分类。

数据定义：Create 语句、Drop 语句。

数据操作：Insert 语句、Update 语句和 Delete 语句。

数据查询：Select 语句。

## 14.3 Select 语句的使用——数据库的灵魂



本节视频教学录像：13 分钟

在 SQL 中，Select 语句用于查询数据库并检索匹配指定条件的数据。

Select 语句的语法是:

```
Select [All | Distinct] Expression [ As name ]
From table1[,table2]
[Where "conditions1"]
[Group By "column-list1"]
[Order By "column-list2" [Asc | Desc] ]
```

可以看到，Select 语句中包含很多的子语句，在这些子语句中，只有 From 子语句是必需的。下表列出了常见的子语句及其功能。

Select 语句的子语句	功能
From	指定要查询的是数据库中的哪个表
Where	用于设置查询的各种条件
As	为查询中的各个字段设置别名
Group By	把查询后返回的记录分组排列
Order By	设置返回的查询结果按照所指定的列递增排列还是递减排列

下表列出了一些 Select 语句中常见的参数及说明。

参数	说明
[All Distinct]	用于限制返回的记录数量, 若没有指定该参数, 则默认值为 All
*	将查询表中的所有字段都选中
column1 [,column2]	查询表的字段名, 可以是一个或者多个, 如果是多个则以逗号隔开, 字段中包含所要查询的数据
As	字段的别名, 替换所要查询字段的名字用于显示
table1[,table2]	所查询的表的名字, 可以是一个或者多个, 如果是多个则以逗号隔开
conditions1	查询属性所使用的条件
column-list1	用于查询结果分组的条件
column-list2	查询结果排序所需要的条件
[Asc   Desc]	Asc 表示升序排列, Desc 表示降序排列

### 14.3.1 Select 子语句

Select 子语句的常用语法是:

Select [fields] FROM [table]

(1) [fields] 指字段的名称, 该字段包含了用户要获取的数据信息。如果查询结果包含多个字段, 则按列举顺序依次获取它们。

(2) [table] 指表的名称。



#### 提示

可以用一个通配符“\*”选取表中的所有字段, 将返回符合条件的数据的所有列。

执行 Select 子语句查询时, 数据库引擎会搜索指定的表, 抽出所选择的列, 选择满足条件的行, 并按指定的顺序对选出的行排序, 或将它们分组。

例如要查询数据库中名为“客户”的表中的所有记录, 可以使用下面的 SQL 语句。

```
Select * From 客户
```

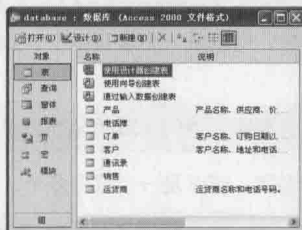
该语句返回并显示“客户”表中所有的数据。

如果不需要显示这么多的内容, 例如只需要显示“联系人姓名”、“联系人头衔”和“电话”这3列, 只需在查询代码中去掉“\*”符号并指定所需要查询的列名字段即可, 列名之间用逗号分隔。

```
Select 联系人姓名, 联系人头衔, 电话 From 客户
```

**【范例 14-1】**在数据库 database.mdb 中, 利用 Select 子句对客户表中的所有信息进行查询 (database.mdb 文件在随书光盘中的“Sample\ch14 文件夹”下)。

(1) 打开随书光盘中的“Sample\ch10\ database.mdb”文件。



(2) 单击【对象】列表中的【查询】按钮, 在打开的【查询】窗口中双击【在设计视图中创建查询】按钮, 弹出【查询 1 选择查询】对话框和【显示表】对话框。

(3) 关闭【显示表】对话框, 然后选择【视图】>【SQL 视图】菜单命令。



(4) 在弹出的【查询 1: 选择查询】对话框中输入以下 Select 子句。

Select \* From 客户

(5) 选择【查询】>【运行】菜单命令，查询出所有符合记录的数据。



### 【拓展训练 14-1】

除了可以查询所有的信息，利用 Select 语句还可以查询所需要的特定信息。

(1) 例如要查询“客户”表中的“联系人姓名”、“联系人头衔”、“电话”等 3 种信息，可以按照【范例 14-1】中的操作步骤，在弹出的【查询 1: 选择查询】对话框中输入以下 Select 子句。

Select 联系人姓名, 联系人头衔, 电话 From 客户

(2) 选择【查询】>【运行】菜单命令，即可查询出所有符合记录的数据。



## 14.3.2 From 子语句

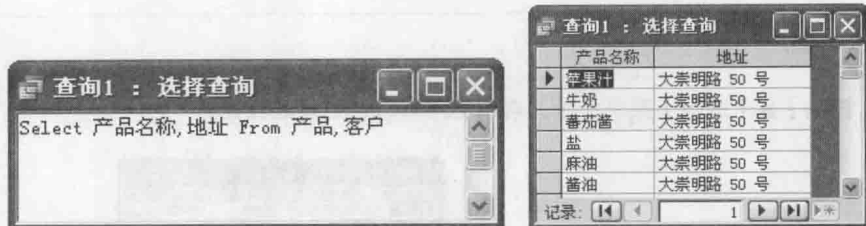
在上面的例子中我们已经使用了 From 子语句，使用 From 子语句指定搜索的表是“客户”表。From 子语句用来决定将要进行查询操作的目标表格，From 子语句是进行数据库查询操作时必不可少的语句，用来为查询字段指明目标。在上面的例子中只使用了一个表进行查询，现在我们使用 From 子语句同时指定多个表，可以在多个表中进行搜索。

**【范例 14-2】**在“产品”表和“客户”两个表中，查询“产品名称”和“地址”等信息。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1: 选择查询】对话框中输入以下语句。

Select 产品名称, 地址 From 产品, 客户

(2) 选择【查询】>【运行】菜单命令，即可查询出所有符合记录的数据。



### 14.3.3 As 子语句

As 子语句用于为查询的字段设置别名,通过设置别名能为查询提供不少方便。例如:

- (1) 当要查询的字段名字特别长或者特别复杂的时候,假如某列的名称是“Administrator”,当我们用 As 子语句将其重命名为 A 以后,就可以输入 A 来进行数据操作,从而可避免输入冗长的名字。
- (2) 当有一个现成的数据库查询公式时,若想套用这个公式查询一个数据库,只需要使用 As 语句把数据库的列名重命名为查询公式中相应的列名即可,而不用把查询公式中所有的列名都更改一遍。
- (3) 当进行多表查询时,如果有两个表中的某一列的名字相同,可以通过 As 语句将它们更直观地区别出来。

#### 【范例 14-3】利用 As 子句,将“客户”表中的“电话”字段更名为“Tel”。

- (1) 按照【范例 14-1】中的操作步骤,在弹出的【查询 1: 选择查询】对话框中输入以下语句。

```
Select 电话 As Tel From 客户
```

- (2) 选择【查询】>【运行】菜单命令,查看更改效果。



从图中可以看到,列名“电话”已更改为了“Tel”。



**提示**

使用 As 子语句为查询字段设置的别名是虚拟的,并不会影响数据库中字段的真实字段名。

### 14.3.4 Where 子语句

在查询时加入 Where 子语句为查询添加条件,则只有满足 Where 子语句条件的内容才会出现在查询结果中。因为查询需求千差万别,所以 Where 子语句中提供有各种操作符,通过这些操作符的有机组合就能够构建非常复杂的查询条件。下表列出了常用的操作符及用途。

操作符	用途
>, >=, =, <, <=, <> (不等于)	用于进行大小比较
Not (非)、And (与)、OR (或)	用于进行多条件的逻辑连接
Between...And...	指定表达式值在指定的范围内
Not Between...And...	指定表达式值不在指定的范围内
In	判断表达式值是否在列表的指定项中
Not In	判断表达式值是否不在列表的指定项中
Like, Not Like	判断值是否与指定的字符通配格式相同
Is Null, Is Not Null	判断表达式是否为空

例如希望在一个学生成绩表中查询出所有成绩大于 60 分的记录，查询语句如下。

```
Select * From 学生成绩表 Where 成绩 >60
```

### 1. And 和 Or 操作符

如果查询条件不止一个，可以通过 And 和 Or 运算符增加条件。And 和 Or 运算符的区别是，And 运算符连接的各种条件必须同时都满足，而 Or 运算符连接的条件只要有一个满足即可。

例如希望在一个学生成绩表中查询出所有成绩高于 60 分并且低于 80 分的结果，查询语句如下。

```
Select * From 学生成绩表 Where 成绩 >60 And 成绩 <80
```

如果希望在一个学生成绩表中查询出所有成绩低于 60 分或者高于 80 分的结果，查询语句如下。

```
Select * From 学生成绩表 Where 成绩 <60 Or 成绩 >80
```

### 2. In 和 Not In 操作符

如果希望查询出所有含有某个我们指定的字符的结果，可以使用 In 操作符。例如要查询数据库中名为“客户”的表中的“联系人头衔”为“市场经理”，并且只显示“联系人姓名”、“联系人头衔”和“电话”这 3 列的结果，查询语句如下。

```
Select 联系人姓名, 联系人头衔, 电话 From 客户 Where 联系人头衔 In ('市场经理')
```

联系人姓名	联系人头衔	电话
苗雅琪	市场经理	(0871) 88601531
王先生	市场经理	(061) 15553392
余小姐	市场经理	(089) 3877310
余小姐	市场经理	(087) 40322121
王先生	市场经理	(0571) 20334560

记录: 1 共有记录 6



Not In 正好和 In 操作符相反, 如果不含有 Not In 中指定的字符, 将会出现在检索结果中。

### 3. Between 操作符

Between 条件运算符用于测试一个数值是否处在 Between 关键字两边指定数值的中间。例如希望在一个学生成绩表中查询出所有成绩高于 60 分并且低于 80 分的结果, 使 Where 和 And 运算符的查询语句如下。

```
Select * From 学生成绩表 Where 成绩 >60 And 成绩 <80
```

使用 Between 运算符的查询语句如下。

```
Select * From 学生成绩表 Where 成绩 Between 60 And 80
```

### 4. 空值运算符

空值运算符将查询出字段为空值的记录。例如在前面的名为“客户”的表中, 有一些客户信息中的“传真”字段没有内容。可以通过下面的语句查询出所有“传真”字段为空的客户。

```
Select 联系人姓名, 联系人头衔, 电话, 传真 From 客户 Where 传真 Is Null
```

查询结果如图所示。

联系人姓名	联系人头衔	电话	传真
王炫皓	物主	(0321) 5553932	
谢丽秋	销售代表	(0571) 45551212	
方先生	物主	(030) 30076545	
刘先生	销售员	(030) 35557647	
王先生	市场助理	(0571) 75559857	
苏先生	物主	(027) 80534871	

## 【范例 14-4】利用 Where 子句, 从“产品表”中查询“库存量”在 80 ~ 120 之间的“产品名称”。

(1) 按照【范例 14-1】中的操作步骤, 在弹出的【查询 1: 选择查询】对话框中输入以下语句。

```
Select 产品名称 From 产品 Where 库存量 >80 And 库存量 <120
```

(2) 选择【查询】>【运行】菜单命令, 查看更改效果。

产品名称
德国奶酪
糯米
浪花奶酪
啤酒
鱿鱼
虾子



该查询结果还可以使用以下查询语句得到。

```
Select 产品名称 From 产品 Where 库存量 Between 80 And 120
```

## 【拓展训练 14-2】

利用 Where 子句，查询“客户”表中的“联系人头衔”不是“市场经理”，并且只显示“联系人姓名”、“联系人头衔”和“电话”这 3 列的结果。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1：选择查询】对话框中输入以下语句。

```
Select 联系人姓名, 联系人头衔, 电话 From 客户 Where 联系人头衔 Not In ('市场经理')
```

(2) 选择【查询】>【运行】菜单命令，查看更改效果。



## 14.3.5 Order By 子语句

Order By 子语是将检索出来的数据，按照一定的顺序进行排列。如果希望查询结果升序排列，在使用 Order By 子句的同时，在查询语句后面加上 Asc；如果希望查询结果降序排列，在使用 Order By 子句的同时，在查询语句后面加上 Desc。

例如希望在一个学生成绩表中将查询出符合条件的成绩按降序排列，可以通过下面的语句实现。

```
Select * From 学生成绩表 Order By 成绩 Desc
```

## 【范例 14-5】在“客户”表中，查询“联系人姓名”并按升序排列。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1：选择查询】对话框中输入以下语句。

```
Select 联系人姓名, 联系人头衔, 电话, 传真 From 客户 Order By 联系人姓名 Asc
```

(2) 选择【查询】>【运行】菜单命令，查看更改效果。



## 【拓展训练 14-3】

如果希望查询到的联系人信息按照“联系人姓名”降序排列，可在弹出的【查询 1：选择查询】对话框中输入以下语句。

Select 联系人姓名, 联系人头衔, 电话, 传真 From 客户 Order By 联系人姓名 Desc

选择【查询】>【运行】菜单命令后, 效果如图所示。

联系人姓名	联系人头衔	电话	传真
马强	市场经理	(091) 65558888	
赵小姐	市场助理	(010) 89551189	
余小姐	市场经理	(089) 38771310	(089) 3877451
余小姐	市场经理	(087) 40322121	(087) 40322120
余小姐	销售经理	(089) 7034214	
余小姐	助理销售代表	(055) 5555939	(055) 5553620
徐先生	销售经理	(030) 23672220	(030) 23672221
徐先生	销售经理	(030) 72035188	
徐文彬	市场助理	(030) 34202378	

### 14.3.6 Group By 子语句

Group By 子语句主要用于指出对查询结果分组的依据, 通常跟聚合函数 (Sum、Max、Min 等) 一起使用。

在“销售”表中, 可以使用 Sum 函数计算所有地区的产品销售总额, 如果希望计算每一地区各自的总销售额该怎么办? 这时就需要用到 Group By 子语句。

**【范例 14-6】**在“销售”表中, 查询“销售地”、“销售额”字段信息, 并在查询到的信息中计算“销售地”的“销售额”的总和。

(1) 按照【范例 14-1】中的操作步骤, 在弹出的【查询 1: 选择查询】对话框中输入以下语句。

Select 销售地, Sum( 销售额 ) As 销售总额 From 销售 Group By 销售地

(2) 选择【查询】>【运行】菜单命令, 查看更改效果。

销售地	销售总额
安徽	340
河北	300
河南	1800
湖北	400
湖南	610
山东	580

## 14.4 SQL 中的常用函数



本节视频教学录像: 6 分钟

SQL 拥有很多可用于计数和计算的内置函数。本节介绍 SQL 中常用的算术函数和统计函数。

### 14.4.1 算术函数

在 SQL 中最常见的 5 种基本运算符是“+”(加)、“-”(减)、“\*” (乘)、“/”(除)和“%”(求余)。还有一些算术函数 SQL 没有提供, 但是很多数据库都支持, 具体说明如表所示。

函数名	函数功能
Abs(x)	返回 x 的绝对值
Sign(x)	当 x 为负数、零、正数的时候分别返回 x 的符号 -1、0 或者 1
Mod(x,y)	返回 x 除以 y 的余数, 和 $x\%y$ 的作用一样
Floor(x)	返回小于等于 x 的最大整数
Ceiling(x) 或 Ceil(x)	返回大于等于 x 的最小整数
Power(x,y)	返回 x 的 y 次方的数值
Round(x)	返回最接近于 x 的数
Round(x,d)	返回四舍五入且小数位数是 d 位的数值
Sqrt(x)	返回 x 的平方根

### 【范例 14-7】利用算术函数, 计算在“产品”表中, 将所有产品涨价 20% 后的结果。

(1) 按照【范例 14-1】中的操作步骤, 在弹出的【查询 1: 选择查询】对话框中输入以下语句。

Select 产品名称, 单价, 单价 \*1.2 As 涨价后的价格 From 产品

(2) 选择【查询】>【运行】菜单命令, 查看更改效果。



## 14.4.2 统计函数

在 SQL 中有 5 种用于统计的函数, 具体的功能如表所示。

函数名	功能
Min	返回一列中最小的数值
Max	返回一列中最大的数值
Sum	返回一列中所有数值的总和
Avg	返回一列中所有数值的平均值
Count	返回一列中所有数值的个数
Count(*)	返回一个表中的行数

### 【范例 14-8】利用统计函数，统计“产品”表中产品的最大库存量、最大订购量和平均价格。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1：选择查询】对话框中输入以下语句。

```
Select Avg( 单价 ) As 平均价格, Max( 库存量 ) As 最大库存量, Max( 订购量 ) As 最大订购量 From 产  
品
```

(2) 选择【查询】>【运行】菜单命令，查看更改效果。



### 【拓展训练 14-4】

利用统计函数，统计“产品”表的总计录数。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1：选择查询】对话框中输入以下语句。

```
Select Count(*) as 表的总数 From 产品
```

(2) 选择【查询】>【运行】菜单命令，查看更改效果。



## 14.5 利用 SQL 语言修改表数据



本节视频教学录像：9 分钟

除了从数据库中查询数据外，也需要向数据库中插入新的数据、更新已有数据、删除某些数据。

### 14.5.1 Insert 语句

Insert 语句用于向数据库的表中插入新的数据。语法如下。

```
Insert Into table [column 1, column 2...]  
Values (Value1, Value2)
```

各参数的含义如下。

table：现存表的名称。

column：表 table 中列 / 字段的名称。

value：赋予列 / 字段的一个有效的值或表达式。

## 【范例 14-9】通过使用 Insert 语句，向“运货商”表中添加一条“木木快递”记录。

(1) 按照【范例 14-1】中的操作步骤，在弹出的【查询 1：选择查询】对话框中输入以下语句。

Select \* From 运货商

(2) 选择【查询】>【运行】菜单命令，查看更改效果。



(3) 再次选择【视图】>【SQL 视图】菜单命令，在弹出的【查询 1：选择查询】对话框中输入以下语句。

Insert Into 运货商 (公司名称) Values ('木木快递')

(4) 选择【查询】>【运行】菜单命令，在弹出的警告对话框中单击【是】按钮，将“木木快递”这条记录插入“运货商”表。



(5) 将【查询 1：选择查询】对话框中的语句清空，并输入以下语句。

Select \* From 运货商

(6) 选择【查询】>【运行】菜单命令，可以看到“木木快递”这条记录已插入“运货商”表。



## 14.5.2 Update 语句

Update 语句用于更改表中的记录。语法如下。

Update table

Set column1=value1 [,column2=value2]

Where condition

各参数的含义如下。

table：现存表的名称。

column：表 table 中列 / 字段的名称。

value: 赋予列 / 字段的一个有效的值或表达式。

condition: 请参考 Where 子句。

### 【范例 14-10】利用 Update 语句将在【范例 14-8】中插入的“木木快递”记录更改为“木木速递”。

(1) 选择【视图】>【SQL 视图】菜单命令，在弹出的【查询 1：选择查询】对话框中输入以下语句。

Update 运货商 Set 公司名称 = '木木速递' Where 公司名称 = '木木快递'

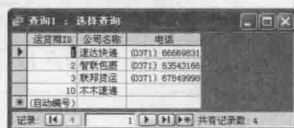
(2) 选择【查询】>【运行】菜单命令，在弹出的警告对话框中单击【是】按钮，“木木快递”记录即更改为“木木速递”。



(3) 将【查询 1：选择查询】对话框中的语句清空，然后输入以下语句。

Select \* From 运货商

(4) 选择【查询】>【运行】菜单命令，可以看到“木木快递”这条记录已更改为“木木速递”。



## 14.5.3 Delete 语句

Delete 语句用于删除一条记录。

语法如下。

Delete [From] {table | view}

[Where condition]

各参数的含义如下。

table: 现存表的名称。

view: 视图的名称。

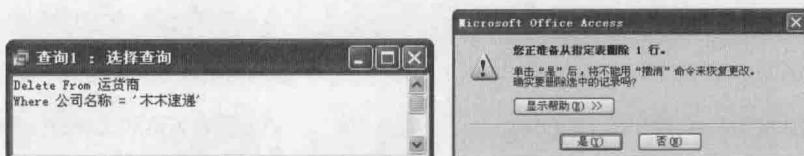
condition: 请参考 Where 子句。

### 【范例 14-11】利用 Delete 语句，将【范例 14-9】中已更改为“木木速递”记录删除。

(1) 选择【视图】>【SQL 视图】菜单命令，在弹出的【查询 1：选择查询】对话框中输入以下语句。

Delete From 运货商 Where 公司名称 = '木木速递'

(2) 选择【查询】>【运行】菜单命令，在弹出的警告对话框中单击【是】按钮，将“木木速递”这条记录从“运货商”表中删除。



(3) 将【查询 1: 选择查询】对话框中的语句清空，然后输入以下语句。

Select \* From 运货商

(4) 选择【查询】>【运行】菜单命令，可以看到“木木速递”这条记录已从“运货商”表中删除。



## 14.6 高手点拨



本节视频教学录像：3 分钟

数据库常用函数（一）：内部合计函数

函数名	含义
COUNT ( * )	返回行数
COUNT ( DISTINCT COLNAME )	返回指定列中唯一值的个数
SUM ( COLNAME/EXPRESSION )	返回指定列或表达式的数值和
SUM ( DISTINCT COLNAME )	返回指定列中唯一值的和
AVG ( COLNAME/EXPRESSION )	返回指定列或表达式中的数值平均值
AVG ( DISTINCT COLNAME )	返回指定列中唯一值的平均值
MIN ( COLNAME/EXPRESSION )	返回指定列或表达式中的数值最小值
MAX ( COLNAME/EXPRESSION )	返回指定列或表达式中的数值最大值

数据库常用函数（二）：日期与时间函数

函数名	含义
DAY ( DATE/DATETIME EXPRESSION )	返回指定表达式中的当月几号



函数名	含义
MONTH ( DATE/DATETIMEEXPRESSION )	返回指定表达式中的月份
YEAR ( DATE/DATETIME EXPRESSION )	返回指定表达式中的年份
WEEKDAY ( DATE/DATETIMEEXPRESSION )	返回指定表达式中的当周星期几
DATE ( NOT DATE EXPRESSION )	返回指定表达式代表的日期值
TODAY	返回当前日期的日期值
CURRENT[FIRST TO LAST]	返回当前日期的日期时间值
COLNAME/EXPRESSION UNITS PRECISION	返回指定精度的指定单位数
MDY ( MONTH, DAY, YEAR )	返回标识指定年、月、日的日期值
DATETIME ( DATE/DATETIME EXPRESSION ) FIRST TO LAST	返回表达式代表的日期时间值
INTERVAL ( DATE/DATETIME EXPRESSION ) FIRST TO LAST	返回表达式代表的时间间隔值
EXTEND ( DATE/DATETIME EXPRESSION, [FIRST TO LAST] )	返回经过调整的日期或日期时间值

## 14.7 实战练习

### 一、思考题

1. 简述 SQL 语言的特点。
2. 使用 As 子句有哪些简便之处?

### 二、上机题

现有一个“学生信息.mdb”数据库,在该数据库中有一个“基本信息”表,表中的各字段如表所示。

字段名	数据类型	字段大小	是否主键
学号	文本	12	否
姓名	文本	16	否
性别	文本	2	否
出生年月	日期		否
专业	文本	50	否

要求在“学生信息.mdb”数据库中,实现对“基本信息”表中的数据记录进行添加、更新、删除及查询等操作。

# 第 15 章



本章视频教学录像：59 分钟

## Visual Basic 与数据库的联合 ——Visual Basic 6.0 中的数据库编程

前面已介绍了数据库与 SQL 语言基础，在 Visual Basic 6.0 中提供有大量的控件用于数据库编程。从本章开始将介绍如何在 Visual Basic 6.0 中进行数据库应用程序开发，主要包括 Visual Basic 6.0 中的数据库控件、建立数据库的方法、数据窗体设计器的使用，以及使用 Data 控件和 ADO 控件进行数据库编程等内容。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉 Visual Basic 中的数据库控件
- ☐ 掌握建立数据库的方法
- ☐ 了解数据查询操作
- ☐ 熟悉数据窗体设计器的使用
- ☐ 熟悉 Data 控件的常用属性、方法和事件
- ☐ 熟悉 ADO 控件的常用属性、方法和事件

## 15.1 英雄相惜——Visual Basic 6.0 与数据库



本节视频教学录像: 3 分钟

当计算机中的文件越来越多时, 随着信息量的不断增大, 查找资料也会变得越来越不方便, 所以人们开始使用数据库来更加科学地存放、管理它们。

数据库 (DataBase, DB) 是一个长期存储在计算机内的、有组织的、有共享的、统一管理的数据集合。它是一个按数据结构来存储和管理数据的计算机软件系统。用 Visual Basic 作为数据库开发平台有以下优点。

(1) 简单性。Visual Basic 提供了数据控件, 利用该控件, 用户只要编写少量的代码甚至不编写任何代码就可以访问数据库, 对数据库进行浏览。

(2) 灵活性。Visual Basic 不像一般的数据库 (如 Access) 那样局限于特定的应用程序结构, 也不需要用某些指令对当前打开的数据库进行操作, 因而比较灵活。

(3) 可扩充性。Visual Basic 是一种可以扩充的语言, 其中包括在数据库应用方面的扩充。在 Visual Basic 中, 可以使用 ActiveX 控件, 也可以由第三方开发者提供。有了这些控件, 可以很容易地在 Visual Basic 中增加新功能, 扩充 Visual Basic 数据存取控制的指令系统。

### 15.1.1 Visual Basic 支持的常用数据库

Visual Basic 支持的常用数据库如下。

#### 1. Visual Basic 内部数据库

Visual Basic 中内置了一个小型的数据库。Visual Basic 内部数据库与 Microsoft Access 数据库具有相同的格式, 可以利用 Visual Basic 直接创建和操作, 小巧、灵活是它最大的特点。

#### 2. 外部数据库

在 Visual Basic 中可以访问的中小型数据库有 Microsoft Access、MySQL、BD2、dBASE、FoxPro 2.0/2.5 和 Paradox 等; 在 Visual Basic 中可以访问的大型数据库有 Oracle、Sybase、DB2、SQL Server 等。在这么多的数据库中, 选择什么数据库一方面取决于实际的需求, 另一方面则要考虑和用户现有系统的兼容, 以及日后的平滑升级和迁移。

#### 3. ODBC

从严格的意义上来说, ODBC 并不是一个数据库, ODBC 是微软公司开放服务结构 (Windows Open Services Architecture, WOSA) 中有关数据库的一个组成部分, 它建立了一组规范, 并提供了一组对数据库访问的标准 API (应用程序编程接口)。通俗一点说, ODBC 是构建在各种各样的数据库之上的一个通用数据库访问方式。ODBC 帮助我们隐藏了访问各种不同数据库的细节, 只要编制好访问 ODBC 的程序, ODBC 就会自动地对不同的数据库采用不同的访问方式。使用 ODBC 驱动程序, 可以建立与几乎任何数据库管理系统连接的应用程序。

### 15.1.2 Visual Basic 中的数据库控件

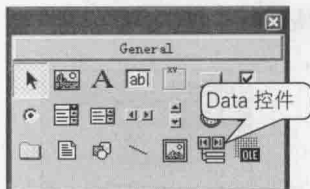
在 Visual Basic 中可以很方便地利用数据库控件访问数据库。Visual Basic 中常用的数据库控件有

以下 3 种。

### 1. 数据访问控件 (Data 控件)

在 Visual Basic 工具箱中提供了一个数据访问控件 (Data 控件)。应用程序中使用数据控件, 可以按如下步骤操作。

- (1) 执行“文件”菜单中的“新建工程”命令, 建立一个新工程。
- (2) 在工具箱中选择数据控件图标, 然后在窗体上画一个数据控件。
- (3) 通过属性窗口设置数据控件的属性。



使用 Data 控件, 不需要任何编程就可以实现对数据库的访问。通过在窗体上绘制 Data 控件, 设置其 DatabaseName 属性和 RecordSource 属性, 就可以生成一个数据库应用程序的数据源。Data 控件在工具箱中, 不需要特殊的步骤加入, 但要利用数据控件返回数据库中记录的集合, 必须通过它的属性设置。使用数据控件, 不需要编写代码就可以实现以下操作。

- (1) 与本地或远程的数据库建立连接。
- (2) 对连接的数据库执行 SQL 查询, 打开指定的数据表或定义记录集。
- (3) 把数据字段传送到各种约束控件, 并可在约束控件中显示或修改数据字段的值。
- (4) 根据约束字段中数据的变化, 添加新记录或更新数据库。
- (5) 捕获访问数据时出现的错误。
- (6) 关闭数据库。

### 2. 数据库绑定控件 (DBGrid 控件)

数据库绑定控件 DBGrid 用于显示和操作代表 Recordset 对象中的记录, 或字段的一系列行和列。它的最重要的属性是 DataSource, 用于设置一个指定 DBGrid 控件的值, 通过这个控件将当前控件连接到数据库中。

### 3. ActiveX 对象

ActiveX 数据对象 (ActiveX Data Object, ADO) 是微软公司一个用于存取数据源的 COM 组件。它提供了编程语言和统一数据访问方式 OLE DB 的一个中间层, 使得开发人员编写访问数据的代码时不用考虑数据库的实现方式。

## 15.2 数据库的建立、维护和查询



本节视频教学录像: 12 分钟

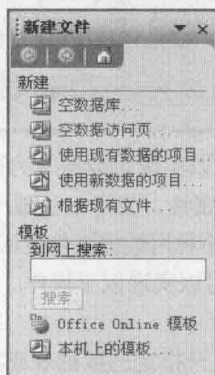
在对数据库的操作中, 数据库的增、删、改、查等可以说是最为常用的功能。下面从建立数据库开始——介绍如何实现上述操作。

## 15.2.1 建立数据库

不同类型的数据库的创建方式大同小异, 本小节以 Microsoft Access 数据库为例, 介绍如何创建一个数据库。建立 Access 数据库的方法有以下两种。

### 1. 直接在 Access 软件中新建一个数据库

在 Access 软件中新建一个数据库非常简单, 就像在 Microsoft Word 中新建一个文档一样。以 Microsoft Access 2003 为例, 单击 Access 软件中的【文件】菜单, 选择其中的【新建】菜单项, 在软件界面右侧将会出现【新建文件】窗格。

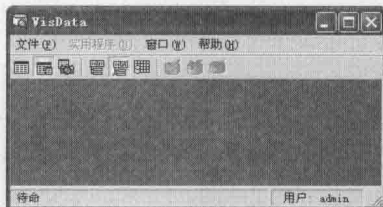


在【新建文件】窗格中, 可以选择是新建一个全新的 Access 数据库, 还是在已有的数据库基础上新建一个 Access 数据库。

### 2. 在 Visual Basic 中使用可视化数据管理器新建一个数据库

Visual Basic 中自带了一个可视化数据管理器, 使用它可以在 Visual Basic 环境中创建一个数据库。

在 Visual Basic 6.0 窗口中选择【外接程序】>【可视化数据管理器】菜单命令, 弹出可视化数据管理器【VisData】窗口。



下面通过一个实例来学习在 Visual Basic 中使用可视化数据管理器新建 Access 数据库的方法。

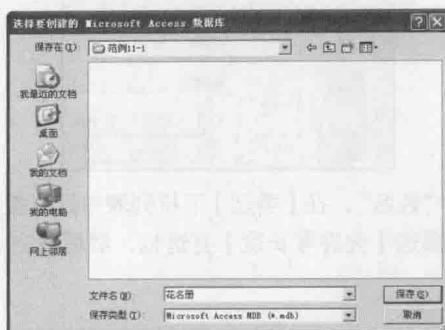
### 【范例 15-1】使用可视化数据管理器创建 Access 数据库。在 Visual Basic 6.0 中, 使用可视化数据管理器创建一个花名册数据库。

#### 第 1 步: 设计表结构

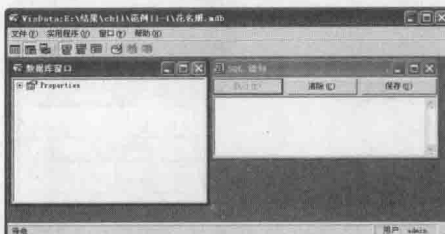
(1) 在可视化数据管理器中选择【文件】>【新建】>【Microsoft Access】>【Version 7.0 MDB (7)】菜单命令。

(2) 弹出【选择要创建的 Microsoft Access 数据库】对话框, 浏览到将要保存该数据库的位置, 在

【文件名】文本框中输入“花名册”，然后单击【保存】按钮。



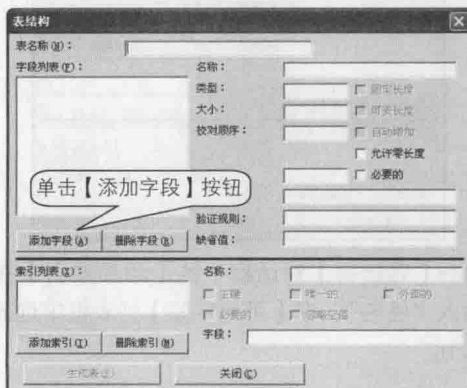
(3) 这样就创建了一个名为“花名册”的数据库。

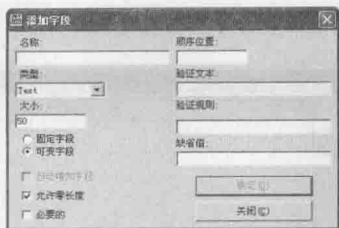


(4) 接下来在该数据库中再创建一个名为“花名册”的表。在可视化数据管理器的【数据库窗口】中的任意位置右击，在弹出的快捷菜单中选择【新建表】命令，弹出【表结构】对话框。

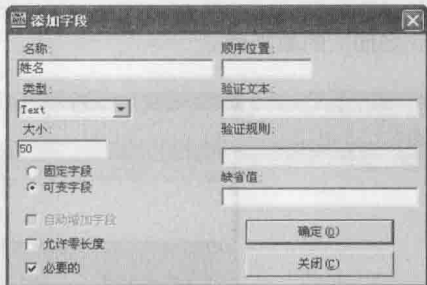


(5) 单击【添加字段】按钮，弹出【添加字段】对话框。





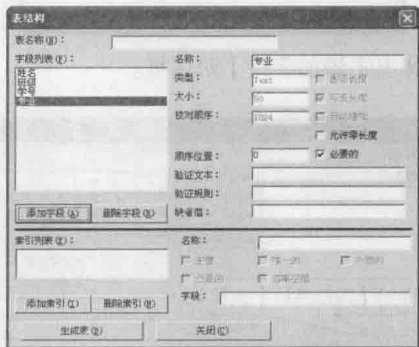
(6) 在【名称】文本框中输入“姓名”，在【类型】下拉列表表中设置类型为【Text】，选中【可变字段】单选按钮和【必要的】复选框，撤选【允许零长度】复选框，然后单击【确定】按钮，即可在表中添加一个“姓名”字段。



(7) 使用同样的方法，按照下表所示的属性值再添加“班级”、“学号”和“专业”等 3 个字段。

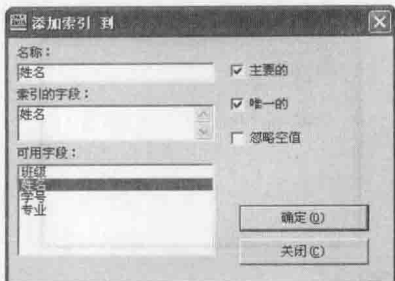
名称	类型	可变字段	允许零长度	必要的
班级	Text	是	否	是
学号	Text	是	否	是
专业	Text	是	否	是

效果如图所示。

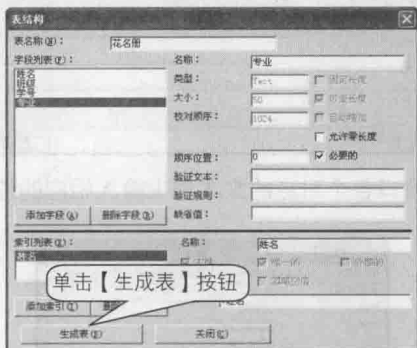


(8) 下面为表添加索引。单击【表结构】对话框中的【添加索引】按钮，在弹出的【添加索引到】对话框中的【名称】文本框中输入“姓名”，在【可用字段】列表框中选择【姓名】选项，然后单击【确定】按钮，最后单击【关闭】按钮。

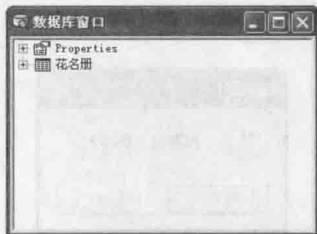






(9) 添加索引后的【表结构】对话框如图所示。接着在【表结构】对话框的【表名称】文本框中输入“花名册”，单击【生成表】按钮。

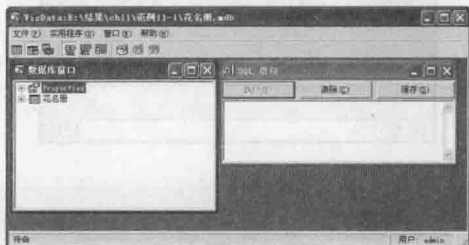


(10) 这样表就设计完成了，在【数据库窗口】中将会出现刚才创建的“花名册”表。

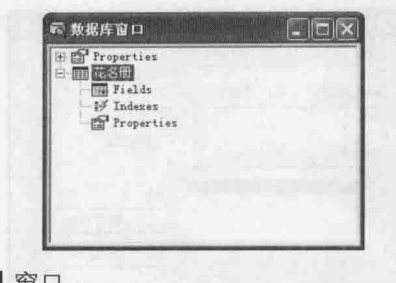


## 第 2 步：添加内容

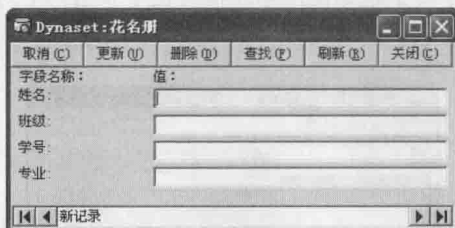
(1) 单击可视化数据管理器工具栏中的【动态集类型记录集】按钮  和【在新窗体上使用 Data 控件】按钮 ，以 Data 控件模式向表中添加记录。



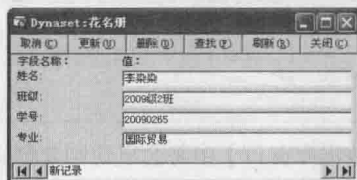
(2) 双击【数据库窗口】中的“花名册”。



(3) 弹出【Dynaset：花名册】窗口。



(4) 在【Dynaset：花名册】窗口各个对应的文本框中输入相应的值，然后单击【更新】按钮。

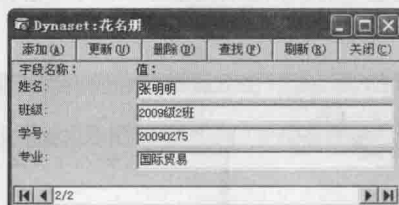


(5) 在弹出的确认对话框中单击【是】按钮。



(6) 使用同样的方法添加其余数据。

(7) 单击【Dynaset：花名册】窗口中的【关闭】按钮，完成数据的添加。



## 15.2.2 删除数据库中的表

我们可以使用 Drop Table 命令来删除数据库中的一个表。语法如下。

Drop Table [tablename];

tablename：要删除的表的名称。



**提示**

表是数据库的基础，表用来储存数据库的源数据，所有的数据操作都源自于表。一个数据库可以有多个表。删除表和删除表格中的全部记录不同。删除表格全部记录后该表格仍然存在，而使用 Drop Table 命令则会将整个数据库表格的所有信息全部删除。

### 15.2.3 修改数据表结构和数据

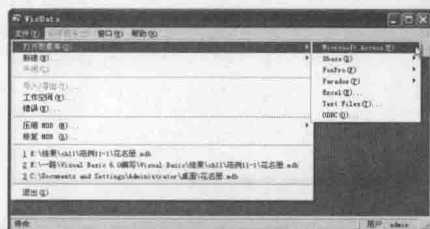
我们可以使用可视化数据管理器修改数据表结构和数据表内容。

#### 1. 使用可视化数据管理器修改数据表内容

使用可视化数据管理器修改数据表内容的具体步骤如下。

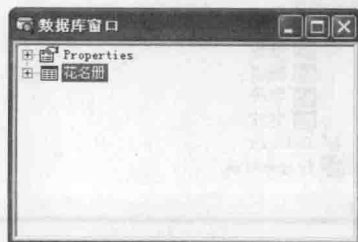
(1) 在可视化数据管理器中选择【文件】>【打开数据库】>【Microsoft Access】菜单命令。

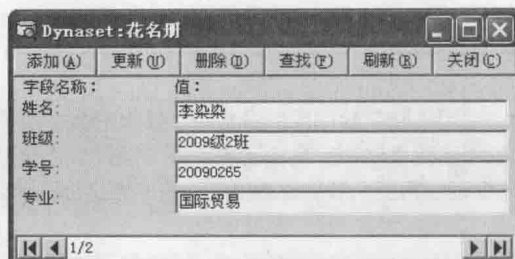
(2) 在弹出的【打开 Microsoft Access 数据库】对话框中找到前面创建的“花名册”数据库，单击【打开】按钮。



(3) 在可视化数据管理器中的【数据库窗口】中双击所要修改的表名。

(4) 在弹出的【Dynaset：花名册】窗口中单击▶按钮，可以逐条浏览表中的数据，并且可以在相应的字段值的文本框中修改数据。数据修改后单击【更新】按钮，即可完成对该条记录的修改。



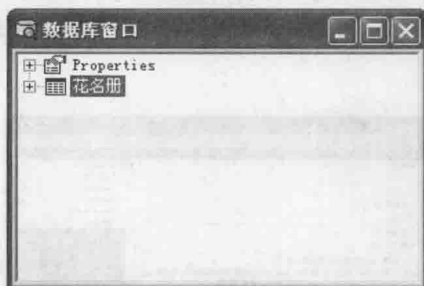


## 2. 使用可视化数据管理器修改数据表结构

表结构的修改是在【数据库窗口】中进行的。

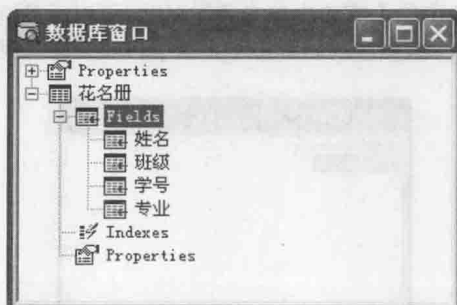
(1) 单击【数据库窗口】中要修改结构的表名前面的【+】符号。

(2) 将会展开 3 项，分别是【Fields】、【Indexes】和【Properties】，分别对应该表的字段、索引和表自身的属性。



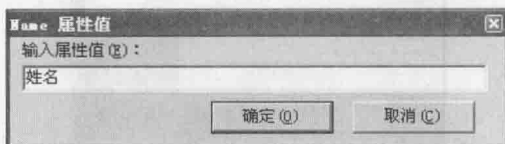
(3) 单击【Fields】项，将会展开该表的所有字段。

(4) 双击需要修改的字段名，将会展开该字段的属性列表。



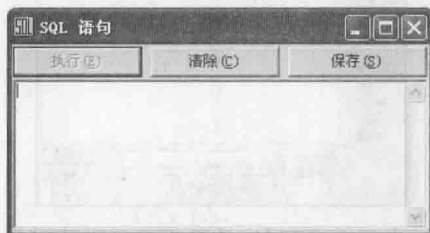


(5) 双击列表中需要修改的属性值，在弹出的【属性值】对话框中输入新的值即可。

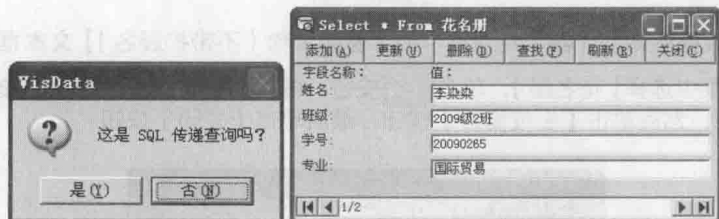


## 15.2.4 数据查询

可视化数据管理器中有一个【SQL 语句】窗体，可以在这个窗体内进行数据查询操作。



打开数据库后，直接在该窗体中输入需要查询的 SQL 语句即可。例如在【SQL 语句】窗体中输入“Select \* From 花名册”语句，然后单击【执行】按钮，弹出的对话框提示“这是 SQL 传递查询吗？”。单击【否】按钮，语句执行的结果如图所示。



SQL 传递查询用于将命令直接发送到 ODBC 数据库服务器。通过使用 SQL 传递查询，可以直接操作服务器表，而不是让 Microsoft Jet 数据库引擎处理数据。

注意

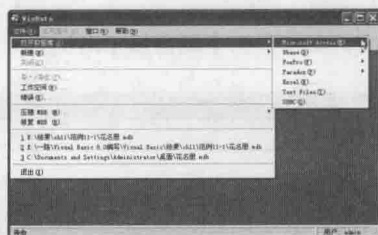
## 15.2.5 数据窗体设计器

在可视化数据管理器中带有数据窗体设计器工具,使用这个工具,我们可以非常快速、方便地设计出一个数据库应用程序,而不用写一行代码。

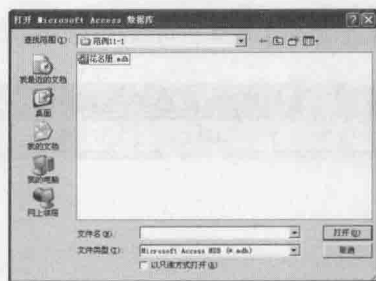
下面使用前面创建的【花名册】数据库和数据窗体设计器工具创建一个数据窗体。

### 【范例 15-2】使用数据窗体设计器创建数据库程序。

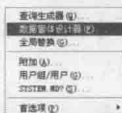
(1) 在可视化数据管理器中选择【文件】>【打开数据库】>【Microsoft Access】菜单命令。




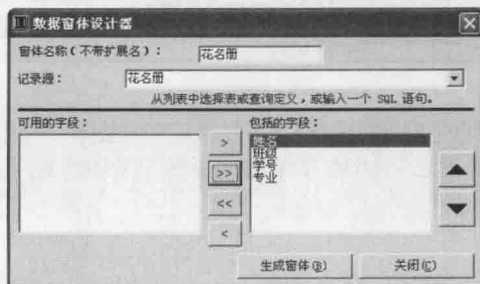
(2) 在弹出的【打开 Microsoft Access 数据库】对话框中,选择随书光盘中的“Sample \ch11\ 范例 15-2\ 花名册 .mdb”数据库文件,单击【打开】按钮。



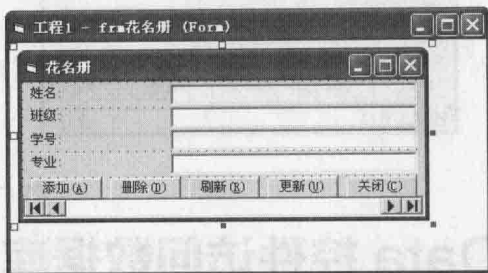
(3) 选择可视化数据管理器中的【实用程序】>【数据窗体设计器】菜单命令。



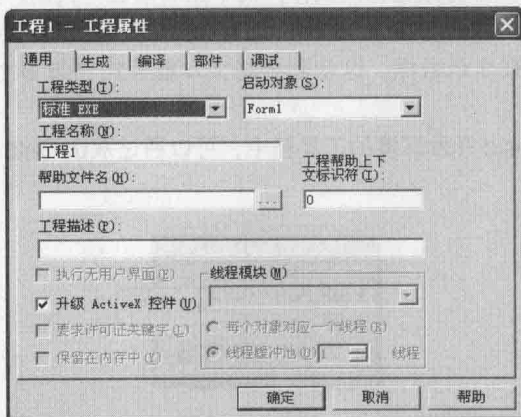
(4) 在弹出的【数据窗体设计器】对话框中的【窗体名称 (不带扩展名)】文本框中输入“花名册”,在【记录源】下拉列表中选择【花名册】,单击  按钮,将【可用的字段】列表框中的所有字段移至【包括的字段】列表框中,然后单击【生成窗体】按钮,最后单击【关闭】按钮。



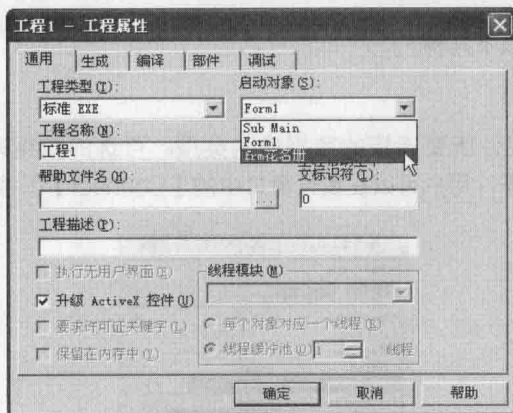
(5) 可以看到，工程中添加了一个名为“frm 花名册”的新窗体。



(6) 在 Visual Basic 6.0 窗口中选择【工程】>【工程 1 属性】菜单命令，弹出【工程 1- 工程属性】对话框。



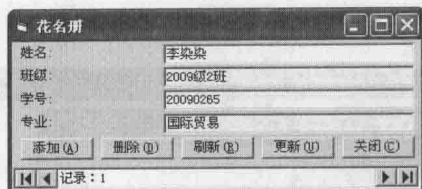
(7) 在【启动对象】下拉列表中选择【frm 花名册】选项，然后单击【确定】按钮。



## 【运行结果】

按快捷键【F5】运行程序，可以看到一个以数据库内容为基础的窗体已经建立起来，随即可以在窗体中进行查找、浏览、添加和删除等操作。





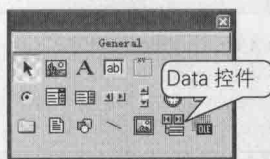
## 15.3 使用 Data 控件访问数据库



本节视频教学录像: 12 分钟

使用 Data 控件可以进行大部分数据访问操作, 而不用编写任何代码。Data 控件通过 Microsoft Jet 数据引擎实现数据访问。Data 控件能够操作数据库, 但它本身不能显示数据库中的数据, 而需要借助其他控件, 如文本框等控件来显示数据。这就需要将文本框等控件与 Data 控件关联起来, 使之成为 Data 控件的数据绑定控件。

Data 控件位于 Visual Basic 开发环境的工具箱中, 可以通过双击 Data 控件按钮向窗体中添加一个 Data 控件。



### 15.3.1 Data 控件的常用属性

Data 控件的常用属性有以下几种。

#### 1. 连接属性 (Connect)

该属性用于设置 Data 控件所要连接的数据库的类型。可选的数据库类型有 Microsoft Access、dBase、FoxPro 和 Paradox 等, 用户可以在属性窗口中的【Connect】下拉列表中选择数据库类型。



#### 2. 数据库名称属性 (DatabaseName)

该属性用于返回或设置 Data 控件的数据源名称及位置。单击属性窗口中该属性后面的 按钮, 将弹出数据库选择对话框供用户设置。



### 3. 记录集类型属性 (RecordsetType)

该属性返回或设置一个值，指出 Data 控件创建的 Recordset 对象的类型。用户可以在属性窗口中的【RecordsetType】下拉列表中设置该属性。



该属性可选的 3 个值的含义如表所示。

属性	值	描述
Table	0	只能对表中的记录进行更新、新增和删除等操作
Dynaset	1	(默认设置) 可以对记录进行更新、新增和删除等操作，可以包含多个表的字段
Snapshot	2	表中的记录为只读状态

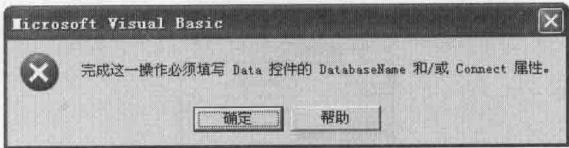
### 4. 记录源属性 (RecordSource)

该属性用于返回或设置基础表、SQL 语句或存储过程。用户可以在属性窗口中的【RecordSource】下拉列表中设置该属性。



在设置 RecordSource 属性前，必须确保已经设置了 DatabaseName 属性或者 Connect 属性，否则将弹出下图所示的对话框。

技巧



5. 独占属性 (Exclusive)

该属性返回或设置一个值，指出打开数据控件底层的数据库是供单用户访问还是多用户访问。当该属性值为 True 时，表示供单用户独占访问；当该属性值为 False 时，表示供多用户访问。该属性的默认值为 False，用户可以在属性窗口中的【Exclusive】下拉列表中设置该属性。



15.3.2 Data 控件的常用方法

Data 控件除了可以用于浏览数据库中的数据外，还提供了一些用于查看、编辑数据库文件的方法。

1. 移动方法 (Move)

Data 控件中提供了用于向前、向后浏览数据库中数据的箭头按钮。除此之外，使用 Move 方法也可以实现同样的功能。语法如下。

```
object.Move left, top, width, height
```

此外，还可以通过表中的方法快捷移动至某一条记录的位置。

方法	功能
MoveFirst	移动至第 1 条记录
MoveLast	移动至最后一条记录
MoveNext	移动至下一条记录
MovePrevious	移动至上一条记录
Move(n)	向前或向后移动 n 条记录

## 2. 查找方法 (Find)

我们可以使用 Find 方法查找满足条件的记录。表中列出了 4 种常用的 Find 方法。

方法	功能
FindFirst	查找满足条件的第 1 条记录
FindLast	查找满足条件的最后一条记录
FindNext	查找满足条件的下一条记录
FindPrevious	查找满足条件的上一条记录

例如, 查找数据库中第 1 条姓名字段值为“大猫”的记录语句如下。

```
Data1.Recordset.FindFirst "姓名 = '大猫' "
```

## 3. 添加新记录方法 (AddNew)

该方法用于将当前记录指向缓冲区, 从而可以添加新的记录。语法如下。

```
recordset.AddNew FieldList, Values
```



### 提示

该方法只是将当前记录指向缓冲区, 添加完新记录后, 需要使用 Update 方法才能将新记录写入数据库。

## 4. 更新记录方法 (Update)

该方法用于修改或添加记录后将数据从缓冲区写入数据库。在调用 AddNew 方法添加新记录后, 必须调用 Update 方法来保存新添加的记录, 否则所添加的记录不会保存在数据库中。

## 5. 删除记录方法 (Delete)

该方法用于删除当前的记录。删除一条记录后, 在数据绑定控件中该数据仍然是可见的, 因此需要调用 Refresh 方法来刷新记录集, 以反映最新的变化。

## 15.3.3 Data 控件的常用事件

Data 控件的常用事件有以下两个。

### 1. Validate 事件

该事件发生在一条不同的记录成为当前记录之前, Update 方法之前, 以及 Delete、Unload 或 Close 操作之前。

### 2. Error 事件

当没有 Visual Basic 代码在执行并且数据存取错误时, 则会激发 Error 事件。语法如下。

```
Private Sub object_Error ([index As Integer,] dataerr As Integer, response As Integer)
```


语句中, object 是一个对象表达式, index 用来标识该控件在一个控件数组中的位置, dataerr 表示错误号, response 是一个对应于所要采用响应的编号。

### 15.3.4 Data 控件访问数据库实例

本小节通过一个实例来学习使用 Data 控件访问数据库的方法。

#### 【范例 15-3】Data 控件访问数据库实例。

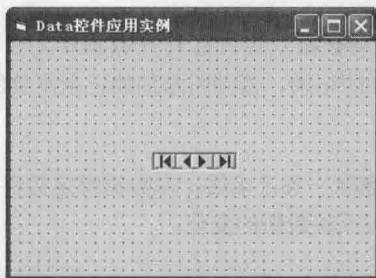
第 1 步: 设置 Data 控件


(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 将 Form1 的 Caption 属性设置为“Data 控件应用实例”。



(3) 在窗体中添加一个 Data 控件。



(4) 选择 Data 控件, 单击属性窗口中 Database Name 属性后面的  按钮, 弹出选择数据库文件的窗口, 从中选择随书光盘中“Sample\ch11\范例 15-2\花名册.mdb”文件, 然后单击【打开】按钮。



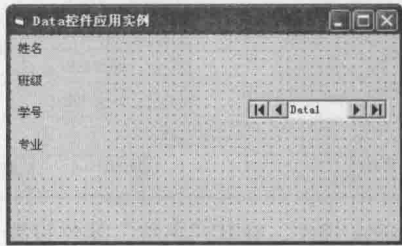
(5) 将 Data 控件所对应的属性窗口中的 RecordSource 属性设置为“花名册”。



第 2 步：设置其余控件

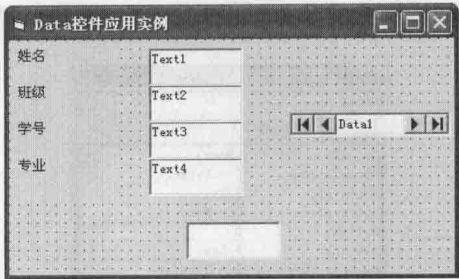
(1) 在窗体中添加 4 个标签（Label）控件，其属性值设置如表所示，并将控件按图所示排列。

控件名称	Caption
Label1	姓名
Label2	班级
Label3	学号
Label4	专业



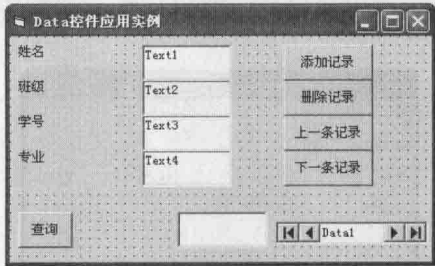
(2) 在窗体中添加 5 个文本框（TextBox）控件，按照下表所示的属性值进行设置，并将控件如图所示排列。

名称	Data Source	Data Field	控件作用
TxtNum	Data1	姓名	显示数据库中的“序号”字段
TxtStuNum	Data1	班级	显示数据库中的“学号”字段
TxtName	Data1	学号	显示数据库中的“姓名”字段
TxtAch	Data1	专业	显示数据库中的“成绩”字段
TxtFind	空	空	输入要查询的字段值



(3) 在窗体上放置 5 个命令按钮（Command Button）控件，按照下表所示的属性值进行设置，并将控件按照如图所示排列。

名称	Caption	控件作用
CmdAdd	添加记录	向数据库中添加记录
CmdDel	删除记录	删除当前记录
CmdPrev	上一条记录	浏览上一条记录
CmdNext	下一条记录	浏览下一条记录
CmdFind	查询	根据输入条件在数据库中查询记录

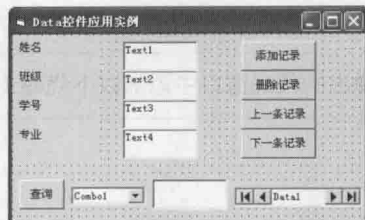


(4) 在窗体中添加一个 ComboBox 控件，将其“名称”属性设置为“CombFind”，Style 属性设置为“2-Dropdown List”。



(5) 最后完成的界面如图所示。





### 第 3 步：为控件添加代码

(1) 双击窗体空白处，在弹出的代码窗口中添加以下代码（代码 15-3-1.txt）。

```
01 Private Sub Form_Load()
02     Data1.Visible = False
    ' 设置 Data1 控件不可见
03     CombFind.AddItem " 姓名 "
    ' 添加查询下拉选项
04     CombFind.AddItem " 班级 "
05     CombFind.AddItem " 学号 "
06     CombFind.AddItem " 专业 "
07 End Sub
```

(2) 双击【添加记录】按钮，在弹出的代码窗口中添加以下代码（代码 15-3-2.txt）。

```
01 Private Sub CmdAdd_Click()
    ' 添加按钮鼠标单击事件代码
02 If CmdAdd.Caption = " 确定 " Then
03     Data1.UpdateRecord
    ' 更新记录集
04     Data1.Recordset.MoveLast
    ' 移动到最后一记录
05     CmdNext.Enabled = True
    ' 锁定其他按钮
06     CmdPrev.Enabled = True
07     CmdDel.Enabled = True
08     CmdFind.Enabled = True
09     CmdAdd.Caption = " 添加记录 "
10 Else
11     Data1.Recordset.AddNew
    ' 添加新的记录
12     CmdAdd.Caption = " 确定 "
13     CmdNext.Enabled = False
14     CmdPrev.Enabled = False
15     CmdDel.Enabled = False
16     CmdFind.Enabled = False
17 End If
```

18 End Sub

(3) 双击【删除记录】按钮，在弹出的代码窗口中添加以下代码（代码 15-3-3.txt）。

```
01 Private Sub CmdDel_Click()  
' 删除按钮鼠标单击事件代码  
02 Dim i As Integer  
03 i = MsgBox("真的要删除当前记录吗?", vbYesNo, "警告")  
' 提示用户是否删除记录  
04 If i = 6 Then  
05 Data1.Recordset.Delete  
' 删除记录  
06 Data1.Refresh  
07 End If  
08 End Sub
```

(4) 双击【上一条记录】按钮，在弹出的代码窗口中添加以下代码（代码 15-3-4.txt）。

```
01 Private Sub CmdPrev_Click()  
' 上一条记录按钮鼠标单击事件代码  
02 Data1.Recordset.MovePrevious  
03 CmdNext.Enabled = True  
04 If Data1.Recordset.BOF Then  
05 Data1.Recordset.MoveFirst  
06 CmdPrev.Enabled = False  
07 End If  
08 End Sub
```

(5) 双击【下一条记录】按钮，在弹出的代码窗口中添加以下代码（代码 15-3-5.txt）。

```
01 Private Sub CmdNext_Click()  
' 下一条记录按钮鼠标单击事件代码  
02 Data1.Recordset.MoveNext  
03 CmdPrev.Enabled = True  
04 If Data1.Recordset.EOF Then  
05 Data1.Recordset.MoveLast  
06 CmdNext.Enabled = False  
07 End If  
08 End Sub
```

(6) 双击【查询】按钮，在弹出的代码窗口中添加以下代码（代码 15-3-6.txt）。

```
01 Private Sub CmdFind_Click()  
' 查询按钮鼠标单击事件代码  
02
```

```

03 If TxtFind.Text = "" Then
' 若用户没有输入查询字符
04 MsgBox " 请选择查询项目并输入查询内容! ", 48, " 提示 "
05 Exit Sub
06 End If
07 Data1.Recordset.FindFirst CombFind.Text & "=" & "" & TxtFind.Text & ""
' 根据选择条件进行查询
08 If Data1.Recordset.NoMatch Then
' 如果没有找到记录, 提示用户记录不存在
09 MsgBox " 记录不存在 ", 64, " 提示 "
10 End If
11 End Sub

```

## 【代码详解】

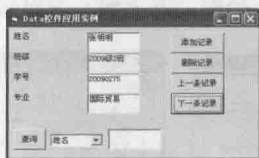
第 3 步步骤(2)中的代码利用 If 判断语句, 如果按下按钮前, 按钮控件 CmdAdd 的标题为“确定”, 则更新记录并移至最后一条记录, 然后设置其他按钮为可用, 并重新设置按钮控件 CmdAdd 的标题为“添加记录”; 反之, 将按钮标题设置为“确定”, 锁定其他按钮为不可用, 等待输入新的记录。

在步骤(3)中, 当按下【删除记录】按钮时, 则弹出对话框, 提示用户“真的要删除当前记录吗?”, 如果返回值为 6, 即用户点击了“YES”按钮, 则调用 Recordset.Delete 事件, 并刷新数据库。

步骤(4)和(5)分别调用移至上一条记录和移至下一条记录方法, 并判断如果到达记录开关或结尾, 则使【上一条记录】或【下一条记录】按钮不可用。

## 【运行结果】

按快捷键【F5】运行程序, 可以使用这个程序完成对数据库中的数据的查询、添加和删除等操作。



## 15.4 使用 ADO 控件访问数据库



本节视频教学录像: 23 分钟

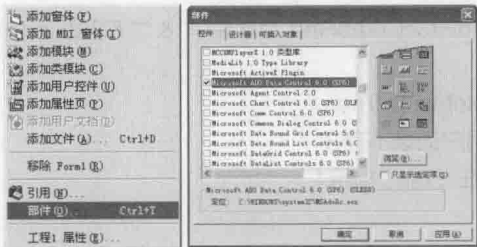
ADO(ActiveX Data Object) 是 Microsoft 数据库应用程序开发的新接口, 是建立在 OLE DB 之上的高层数据库访问技术, 请不必为此担心, 即使你对 OLE DB、COM 不了解也能轻松对付 ADO, 因为它非常简单易用, 甚至比你以往所接触的 ODBC API、DAO 都要容易使用, 并不失灵活性。

### 15.4.1 添加 ADO 控件

在讨论、研究 ADO 控件的属性、方法和事件之前, 首先要将 ADO 控件添加到工具栏中。具体的

操作步骤如下。

- (1) 在 Visual Basic 6.0 窗口中选择【工程】>【部件】菜单命令。
- (2) 在弹出的【部件】对话框中选择【控件】选项卡，选择【Microsoft ADO Data Control 6.0 (OLEDB)】复选框，然后单击【确定】按钮。



(3) 这样就在工具箱中增添了一个 ADO 控件。



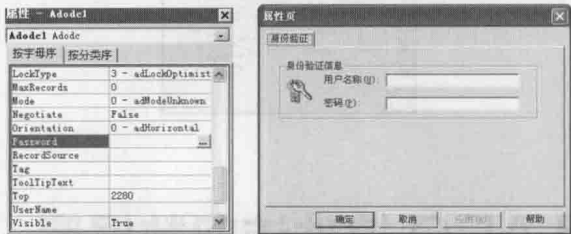
### 15.4.2 ADO 控件的常用属性

ADO 控件的常用属性有以下几种。

#### 1. 密码属性 (Password)

该属性用于设置 ADO Recordset 对象创建过程中所使用的口令。用户可以在属性窗口中的【Password】中设置该属性。

单击 Password 属性后面的...按钮，将弹出要求输入身份验证的对话框，如图所示。



#### 2. Recordset 属性

本属性返回或设置对下一级 ADO Recordset 对象的引用。  
使用该属性的语法如下。

object.Recordset [= recordset]

#### 3. RecordSource 属性

该属性返回或设置一个记录集的查询。用户可以在属性窗口中的【RecordSource】中设置该属性。



### 15.4.3 ADO 控件的常用方法

UpdateControls 方法用于从控件的 ADO Recordset 对象中获取当前行，并在绑定到此控件的控件中显示相应的数据。

语法如下。

```
object.UpdateControls
```

### 15.4.4 ADO 控件的常用事件

Error 事件只有在没有执行任何 Visual Basic 代码而发生了一个数据访问错误的情况下才会被触发。语法如下。

```
object_Error([Index As Integer,] ByVal Error Number As Long, Description As String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, fCancelDisplay As Boolean)
```

语句中主要关键字的含义如表所示。


部分	描述
Object	一个对象表达式，其值为“应用于”列表中的一个对象
Index	表示该控件在控件数组中的位置
ErrorNumber	本地错误号码
Description	错误描述
Scode	服务器返回的错误代码
Source	错误的来源
HelpFile	包含该错误详细信息的帮助文件的路径
HelpContext	帮助主题的上下文号码
fCancelDisplay	一个布尔值，可以设置这个值来取消对错误消息的显示

## 15.4.5 ADO 控件访问数据库实例

本小节通过一个实例来学习使用 ADO 控件访问数据库的方法。

### 【范例 15-4】在 Visual Basic 6.0 中使用 ADO 控件连接数据库。

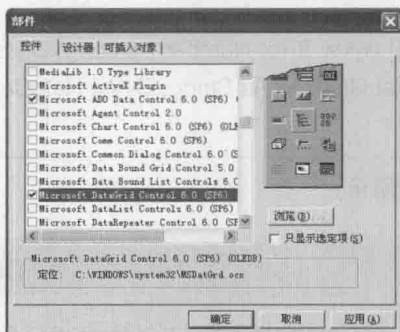
第 1 步：设置 ADO 控件

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

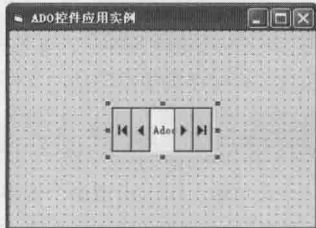
(2) 将 Form1 的 Caption 属性设置为“ADO 控件应用实例”。



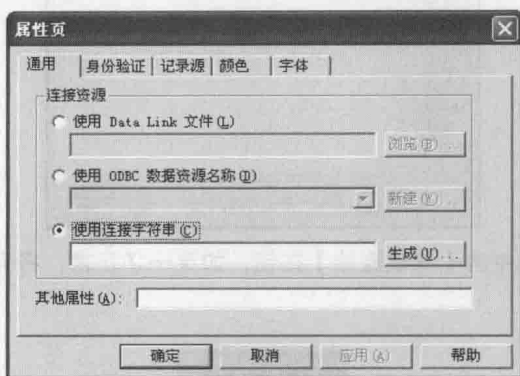
(3) 选择【工程】>【部件】菜单命令，打开【部件】窗口，在【控件】选项卡中选择【Microsoft ADO Data Control 6.0 (OLE-DB)】和【Microsoft DataGrid Control 6.0 (OLEDB)】复选框，然后单击【确定】按钮，将 DataGrid 控件和 Adodc 控件添加到工具箱中。



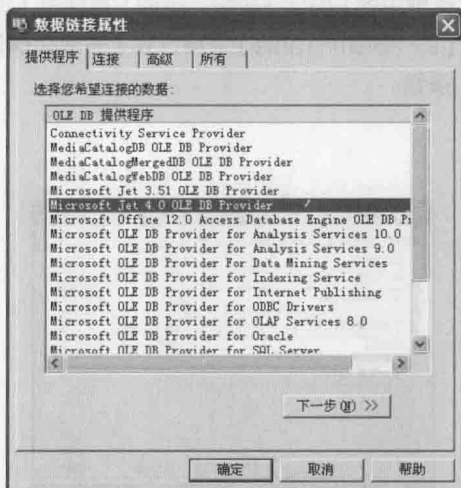
(4) 双击工具箱中的“Adodc 控件”按钮，在窗体中添加一个 Adodc 控件。



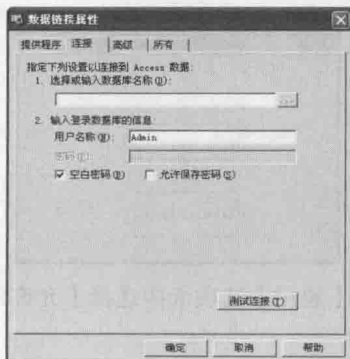
(5) 在 Adodc 控件上右击，在弹出的快捷菜单中选择【ADODC 属性】菜单项，在弹出的【属性页】对话框中，选中【通用】选项卡中的【使用连接字符串】单选按钮，然后单击【生成】按钮。



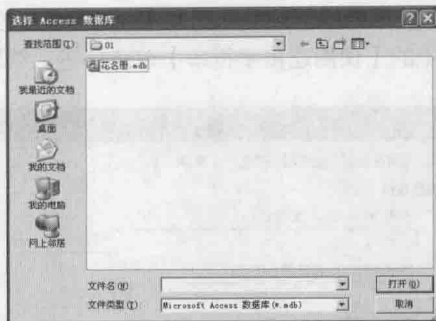
(6) 弹出【数据链接属性】对话框，选择【Microsoft Jet 4.0 OLE DB Provider】选项，然后单击【下一步】按钮，对话框将切换到【连接】选项卡。



(7) 在【连接】选项卡中，单击【1. 选择或输入数据库名称】文本框右侧的...按钮，弹出【选择 Access 数据库】对话框，从中选择随书光盘中的“Sample\ch11\范例 15-2\花名册.mdb”数据库文件，然后单击【打开】按钮。



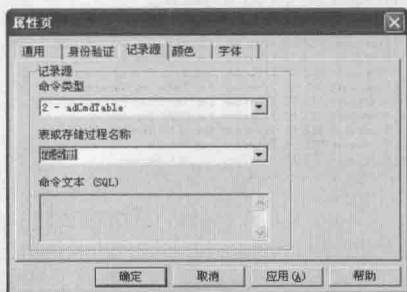




(8) 单击【连接】选项卡中的【测试连接】按钮，如果一切正常，将会弹出测试连接成功对话框，单击【确定】按钮。

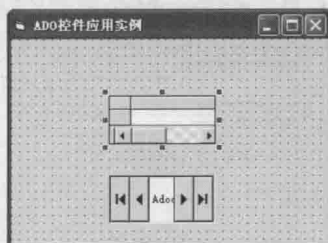


(9) 单击【数据链接属性】对话框中的【确定】按钮，返回【属性页】对话框，选择【记录源】选项卡，在【命令类型】下拉列表中选择【2 - adCmdTable】选项，在【表或存储过程名称】下拉列表中选择【花名册】选项，然后单击【确定】按钮。

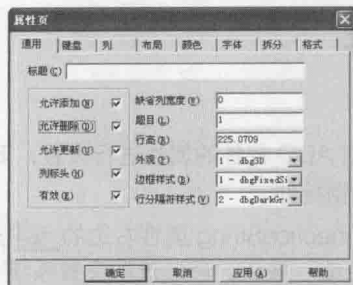


## 第2步：设置 DataGrid 控件

(1) 双击工具箱中的“DataGrid 控件”按钮，在窗体中添加一个 DataGrid 控件。右击添加的 DataGrid 控件，在弹出的快捷菜单中选择【属性】菜单项。



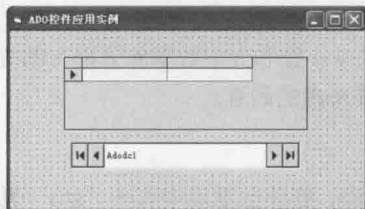
(2) 弹出【属性页】对话框，在【通用】选项卡中选择【允许添加】和【允许删除】两个复选框，然后单击【确定】按钮。



(3) 将 DataGrid 控件对应的属性窗口中的 DataSource 属性设置为“Adodc1”。

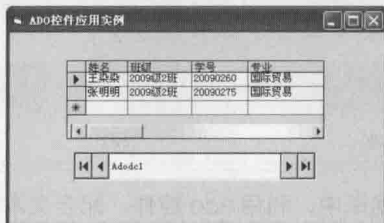


(4) 将添加的两个控件按照下图排列整齐。



## 【运行结果】

按快捷键【F5】运行程序，随之可以使用这个程序，利用 ADO 控件完成对数据库中的数据的查询、添加和删除等操作。利用 ADO 控件可以不添加任何代码而设计出能操作数据库的程序。



如果想改变 ADO 数据对象中的显示标题，选中 ADO 对象并且打开其属性窗口，修改其中的 Caption 属性即可。

## 15.5 高手点拨



本节视频教学录像：9 分钟

- 连接 ACCESS 数据库时必须要对 ADO 控件的属性进行设置，下面提供了详细的设置步骤。
- (1) 先在窗体上放置一个 ADO 数据控件。
  - (2) 在 ADO 属性窗口中单击 ConnectionString 属性右边的 按钮，从对话框中选择连接数据源的方式：使用连接字符串——单击“生成”按钮，通过选项设置系统自动产生连接字符串使用 Data Link 文件——通过一个连接文件来完成使用 ODBC 数据资源名称——在下拉列表中选择某个创建好的数据源名称作为数据来源对远程数据库进行控制。
  - (3) 在 ADO 属性窗口中单击 RecordSource 属性右边的 按钮，指出可以操作的数据库的来源（即结果字符串，可以是 SQL 查询产生的）。在设置好后，可以用文本框 (textbox) 等绑定数据表中的字段进行显示。在“命令类型”中选择 2——adCmdTable，在“表或存储过程名称”中选择所需要的表。
- 以上步骤(2)、(3)可以合并成一步：在 ADO 控件上单击右键，从快捷菜单中选择 ADODC 属性，直接在属性页对话框中进行所有设置。
- 设置好以后控件就能访问数据库了。

## 15.6 实战练习

- 一、思考题
- 1. 在 Visual Basic 中可以访问的中小型和大型数据库分别有哪些？
  - 2. 简述在选择数据库类型时需要考虑的因素。
- 二、上机题
- 现有一“学生信息 .mdb”数据库，在该数据库中有一“stu”表，表中的各字段如表所示。
- | 字段名  | 数据类型 | 字段大小 | 是否主键 |
|------|------|------|------|
| 学号   | 文本   | 12   | 否    |
| 姓名   | 文本   | 16   | 否    |
| 性别   | 文本   | 2    | 否    |
| 出生年月 | 日期   |      | 否    |
| 装爷   | 文本   | 50   | 否    |
- 要求在“学生信息 .mdb”数据库中，利用 ado 控件，配合文本框和命令按钮控件，实现对“stu”表中的数据记录进行添加、更新、删除及查询等操作。

# 第16章



本章视频教学录像：17 分钟

## Visual Basic 6.0 生成的报表——数据报表

通过使用数据报表功能，用户可以很方便地将所需要的数据以报表的形式显示或打印出来，以便阅读和判断数据信息。本章通过对数据环境设计器（Data Environment Designer）和数据报表设计器（Data Report Designer）的讲解，来实现数据报表功能。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解数据报表生成环境
- ☐ 掌握数据报表的生成
- ☐ 能够制作简单的报表

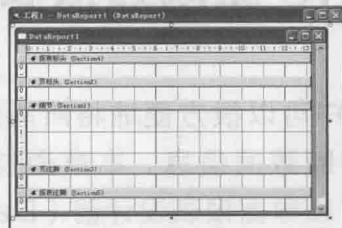
## 16.1 数据报表简介

本节视频教学录像: 2 分钟

报表就是用表格、图表等格式来动态显示数据, 也可以用一个公式来表示: 报表 = 多样的格式 + 动态的数据。

报表用于访问、格式化数据, 并把数据信息以可靠和安全的方式呈现给使用者。通过清晰明确的数据关系, 以帮助用户进行有效的决策。有效的报表须以逻辑方式提供正确的数据。如果提供错误的的数据, 或用随意的样式提供正确数据, 那么制作出来的报表将会减慢决策的进程, 甚至导致不正确的决定。

在 Visual Basic 6.0 中, 一般是通过数据环境设计器与数据报表设计器相结合来对数据进行报表设计。数据报表设计器如图所示。



## 16.2 数据报表的生成环境

本节视频教学录像: 11 分钟

在 Visual Basic 中制作报表, 一般是用数据环境设计器 (Data Environment Designer) 与数据报表设计器 (Data Report Designer) 相结合来实现的。也就是说, 首先利用数据环境设计器创建与数据库的连接, 在连接的过程中需要选择指定的数据库文件, 待完成与数据库的连接后, 再利用 Command 对象连接数据库内的表。




**提示**

用数据环境设计器与数据报表设计器相结合来制作的报表, 通常应用于客户 / 服务器 (C/S) 结构。

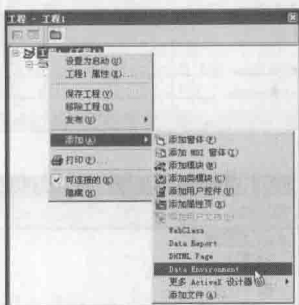
下面举例说明, 如何打开数据环境设计器窗口和数据报表设计器窗口。

**【范例 16-1】** 首先添加数据环境设计器和数据报表设计器, 然后对数据环境设计器进行配制, 使其能够正确连接至数据表和表中字段。最后, 配制数据报表设计器, 生成报表。

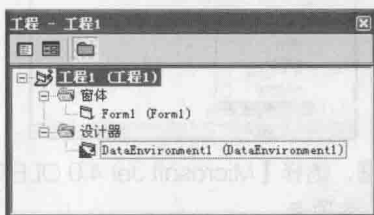
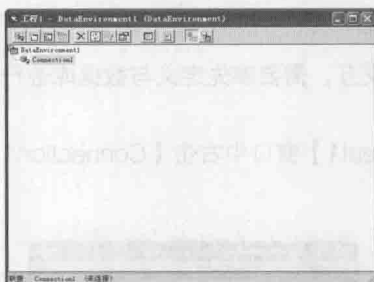
第 1 步: 添加数据环境设计器和数据报表设计器

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

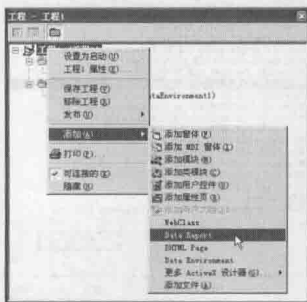
(2) 在【工程】窗口中，用鼠标右击【工程 1 (工程 1)】工程名，在弹出的快捷菜单中选择【添加】>【Data Environment】菜单命令。



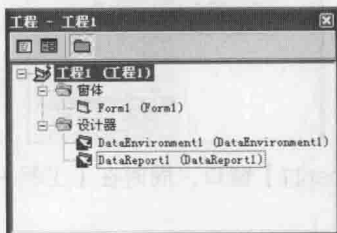
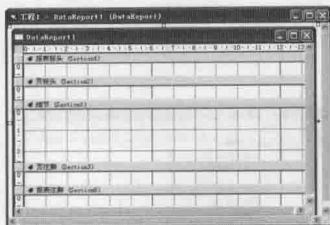
(3) 弹出【工程 1—DataEnvironment1】窗口，同时在【工程—工程 1】窗口中，会添加一个名为【DataEnvironment1】的设计器文件。



(4) 在【工程—工程 1】窗口中，用鼠标右击【工程 1 (工程 1)】工程名，在弹出的快捷菜单中选择【添加】>【Data Report】菜单命令。



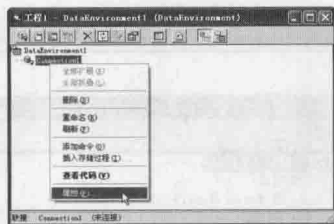
(5) 弹出【工程 1—DataReport1】窗口，同时在【工程—工程 1】窗口中，会添加一个名为【DataReport1】的设计器窗体。



## 第2步：配制数据环境设计器

为了能和设计好的数据库进行交互，需要事先定义与数据库进行连接的字符串，然后再定义与数据库中表的连接。

(1) 在【工程1—DataEnvironment1】窗口中右击【Connection1】对象，在弹出的快捷菜单中选择【属性】菜单项。

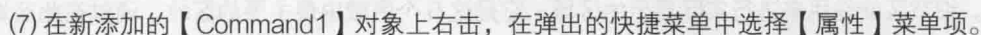
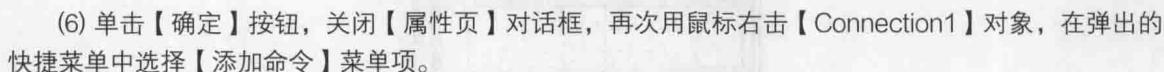
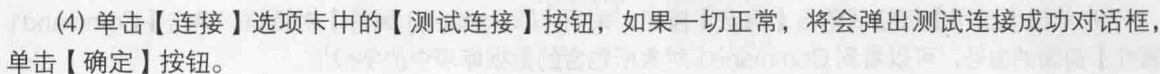


(2) 弹出【数据链接属性】对话框，选择【Microsoft Jet 4.0 OLE DB Provider】选项，然后单击【下一步】按钮，对话框切换到【连接】选项卡。

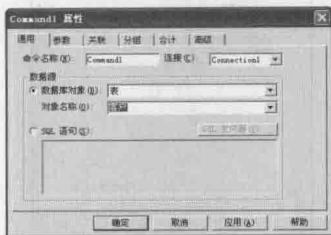


(3) 在【连接】选项卡中，单击【1. 选择或输入数据库名称】文本框右侧的...按钮，选择 Access 数据库，然后从中选择随书光盘中的“Sample\ch16\database.mdb”数据库文件。





(8) 弹出【Command1 属性】对话框, 选择【数据库对象】为“表”, 选择【对象名称】为“客户”。

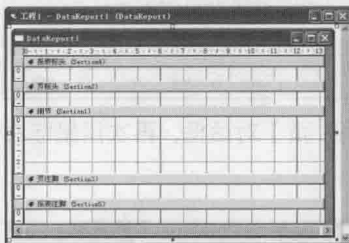


(9) 单击【应用】按钮后单击【确定】按钮, 关闭【Command1 属性】对话框。单击【Command1 属性】前面的田号, 可以看到 Command1 对象所包含的数据库表中的字段值。



### 第 3 步: 配制数据报表设计器

在配制数据报表设计器之前, 先来熟悉一下数据报表设计器的设计界面。



数据报表设计器窗口由报表标头、页标头、细节、页注脚和报表注脚等 5 部分组成。

1. 报表标头。出现于整个报表的每一页, 可以用标签建立一个报表名。在报表中该部分只会在报表的顶部出现一次。

2. 页标头。只能出现在当前页的最上部, 也就是说它只能出现在当前页中, 不出现在其他页面中, 可以放置一些字段名等。

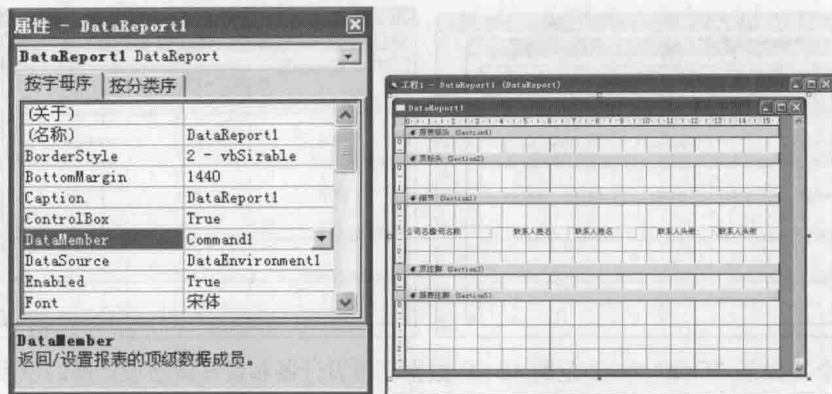
3. 细节。细节区就是用来进行实际显示的区域, 它是这 5 个区域中最主要的区域, 通过在此区域内放置显示控件可以控制报表的实际显示输出。

4. 页注脚。页注脚只能出现在当前页的最下部, 它同样只能出现在当前页中, 不出现在其他页中, 可以放置随页面变化的一些量, 例如页码等。

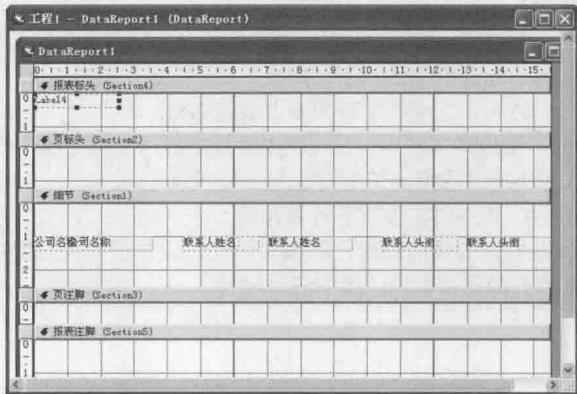
5. 报表注脚。每份报表中只能有一个报表注脚, 可以用标签建立对本报表的注释、说明等。

(1) 选择【datareport1】窗体, 在属性窗口中设置【DataSource】属性为“DataEnvironment1”, 【DataMember】属性为“Command1”。

(2) 将数据环境设计器中 Command1 对象内的【公司名称】、【联系人姓名】和【联系人头衔】等 3 个字段拖曳到数据报表设计器的细节区。



(3) 选中“报表标头”区，双击工具箱中的【RptLabel】标签，在报表标头区添加一个 RptLabel 标签。



(4) 当鼠标指针放至 RptLabel 标签上时，会变成“手形”图标，按下鼠标不放，拖动 RptLabel 标签至“报表标头”区的中央。

(5) 选中 RptLabel 标签，在属性窗口中更改【Caption】属性为“我的第一张报表”，设置 Font 属性的【字体】为隶书，大小为【二号】，并将 RptLabel 标签调整至适当的位置。



(6) 选中“页标头”区，双击工具箱中的【RptLabel】标签 3 次，在“页标头”区添加 3 个 RptLabel 标签，其【Caption】属性分别按照“细节”区的字段名进行设置，并将其调整至适当的位置。

(7) 将“细节”区的各标签删除，并将【RptTextBox】文本框调整至适当的位置。



(8) 保存整个工程至“Finalch16\范例 16-1\数据环境设计器和数据报表设计器的使用”文件夹下。



#### 提示

添加数据报表设计器后，在工具箱中会出现一组【数据报表】工具栏。【数据报表】工具栏中主要有以下几个控件。

- (1) RptLabel 控件。该控件主要用于在报表或页的标头显示说明信息或标识字段。
- (2) RptTextBox 控件。该控件主要用于在报表上放置规定的格式文本。
- (3) RptImage 控件。该控件主要用于在报表上放置图形。
- (4) RptShape 控件。该控件主要用于在报表上放置矩形、三角形或（椭）圆形等。
- (5) RptLine 控件。该控件主要用于在报表上绘制标尺。
- (6) RptFunction 控件。该控件主要用于计算数值。

使用数据报表设计器最大的优点就是无需编写代码，只需进行数据绑定就可以实现最终的数据报表。下面介绍如何在数据报表设计器中进行配制。

## 16.3 数据报表的生成

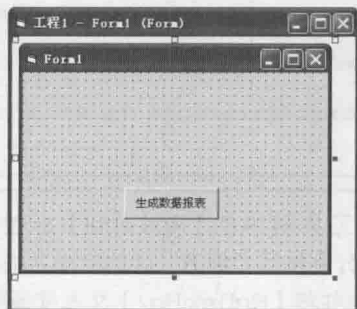


本节视频教学录像：3 分钟

对数据环境设计器和数据报表设计器进行添加和配制后，最为重要的就是如何能够运行数报表并进行打印。

(1) 双击随书光盘中的“Finalch16\范例 16-1\数据环境设计器和数据报表设计器的使用”文件夹下的工程名，打开工程文件中的所有窗体和设计器。

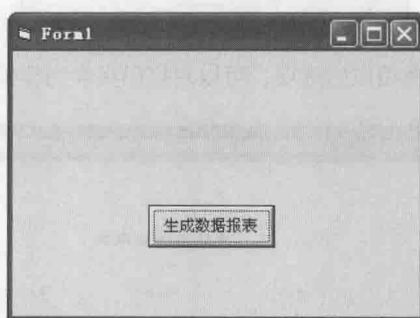
(2) 选中【Form1】窗体，从中添加一个【CommandButton】命令按钮，设置其【Caption】属性为“生成数据报表”。



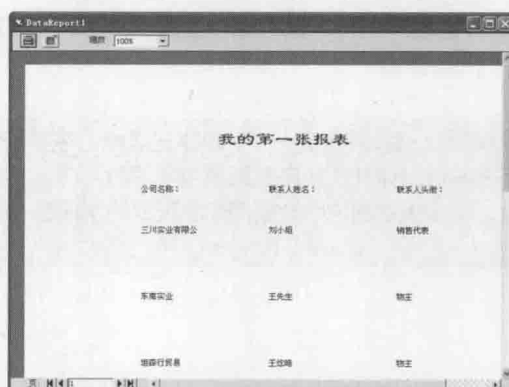
(3) 双击【Form1】窗体上的【生成数据报表】按钮，在打开的代码窗口中输入以下代码。


```
Private Sub Command1_Click()  
DataReport1.Show  
End Sub
```

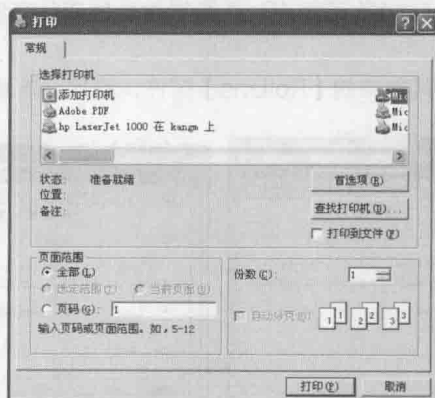
(4) 按【F5】快捷键运行程序。




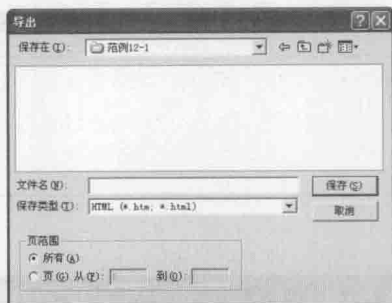
(5) 单击【生成数据报表】按钮，弹出数据报表预览窗口。



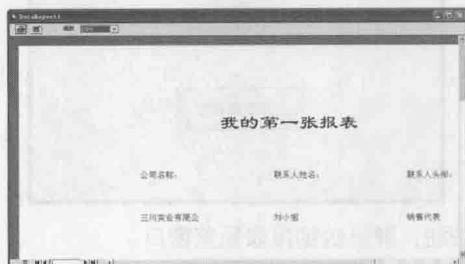
(6) 单击预览窗口上的【打印】按钮，在弹出的【打印】对话框中选择打印机对报表进行打印。




(7) 单击预览窗口上的【导出】按钮，在弹出的【导出】对话框中可以将报表内容输出成文本文件或 Html 文件。



(8) 在【缩放】下拉列表中选择相应的选项，可以对打印报表内容缩小或放大查看。



(9) 单击【结束】按钮 , 停止程序运行。保存整个工程至“Final\ch12\范例 16-2\数据报表生成”文件夹下。



#### 提示

使用数据报表时，TextBox 必须绑定到一个 ADO 记录集。在某些情况下，你需要在运行时创建一个数据报表，而在设计时你并不知道数据源中字段的名称。为完成此功能，你必须在工程中添加一个数据报表，该报表中的控件数须同你想获取的字段数一致。然后打开 ADO 记录集，循环更新数据报表。

## 【拓展训练 16-1】

对报表设计好之后，可以利用线条控件在报表内加入直线，或利用图形控件和形状控件加入图案或图形，以达到对表美化的作用。

(1) 双击随书光盘中的“Final\ch12\范例 16-2\数据报表生成”文件夹下的工程名，打开工程文件中的【DataReport1】设计器。

(2) 选中“细节”区，双击工具箱中的【RptLine】控件，添加一条直线。



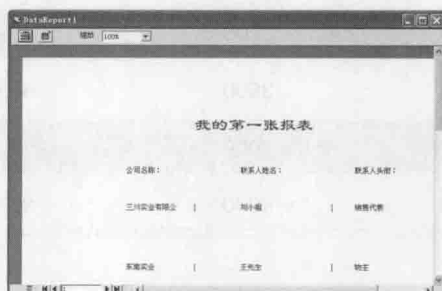
(3) 将直线移动至第 1 个字段和第 2 个字段之间，然后在【属性】窗口中设置【Width】值为 0,【Height】


值为 200。

(4) 使用同样的方法，在第 2 个字段和第 3 个字段之间添加一条直线。



(5) 按【F5】快捷键运行程序，然后单击【生成数据报表】按钮，弹出数据报表预览窗口，从中可以看到字段与字段之间以短竖线隔开了。



(6) 单击【结束】按钮 , 停止程序运行。保存整个工程至“Finalch12\范例 16-3\美化报表”文件夹下。

## 16.4 高手点拨



本节视频教学录像：1 分钟

Visual Basic 中制作报表通常有以下几种方法。

方法一：利用数据环境设计器（Data Environment Designer）与数据报表设计器（Data Report Designer）这两个工具来设计。这是 VB 中最常规的报表制作方法，VB 编程人员可利用这两个工具来制作一些简单的报表。

方法二：使用第三方产品来完成并通过 Activex 控件输出。我们可以从网络上下载诸如水晶报表、Activereport、Videosoft Vsgird 等工具设计数据报表。

方法三：利用 Excel 对象将数据库导出到 Excel 工作簿。

## 16.5 实战练习

### 一、思考题

数据报表设计器窗口一共由哪 5 部分组成？试简述它们的作用。



## 二、上机题

根据某公司 2014 年 10 月份工资表,通过数据库应用程序,实现工资表的报表和打印功能。

编号	姓名	基本工资	资金	总计
50101	刘赫	3500	500	4000
50201	陈真	3500	800	4300
50202	张三	3800	800	4600
50301	李四	3800	600	4400
50302	王五	3500	700	4200
50303	赵六	3800	800	4600
50304	侯七	3500	600	4100
50202	胡凤娇	4000	800	4800
50305	庄前	4000	800	4800

# 第 17 章



本章视频教学录像：33 分钟

## Visual Basic 编程的核心——API 编程

Windows 应用程序接口（简称 API）是微软公司为开发者提供的重要功能之一，它可以控制 Windows 系统的各个部件的外观和行为，在 Visual Basic 6.0 中可以方便地调用 API 函数。

本章讲解 API 的相关概念、API 浏览器的使用以及几种最为常见的 API 函数，并通过一个实例学习 API 函数的使用。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉 API 的相关概念
- ☐ 掌握 API 浏览器的使用
- ☐ 了解窗口管理类函数
- ☐ 熟悉图形设备接口类函数
- ☐ 了解系统服务类函数
- ☐ 了解网络服务函数
- ☐ 熟悉 API 函数的应用

# 17.1 API 概述



本节视频教学录像：10 分钟

软件的运行需要使用计算机的各种资源，例如内存空间、硬盘空间或者调用显示器的功能等。自己编写程序来直接管理这些资源虽然可行，但这无疑会让程序员多做很多和软件功能无关的编程。计算机操作系统（如 Windws、Linux、UNIX 等）就是为了解决这一问题而产生的，它们使计算机变得更加容易使用。例如在 Windows 操作系统下，双击某个文件的图标就可以打开该文件，而不必了解该文件在硬盘中的实际位置以及如何访问它。

为了方便程序通过 Windows 调用计算机资源，将一些常用的功能（如创建窗体、绘制图形等）做成一个个函数，通过调用这些函数提供的接口就可以方便地调用计算机资源，这些接口称作应用程序接口（Application Programming Interface, API）。



提示

Visual Basic 函数是在 Visual Basic 开发环境中自动定义的，而 Windows API 函数要求你首先要在工程的标准模块或事件过程中声明它们。

## 17.1.1 API 基本数据类型

API 编程所用到的数据类型和 Visual Basic 中常见的数据类型有所不同，下表列出了 Windows API 中常见的基本数据类型及其描述。

API 数据类型	描述
WORD	16 位无符号整数
LONG	32 位有符号整数
DWORD	无符号长整数
HANDLE	用做句柄的 32 位整数
HWND	用做窗口句柄的 32 位整数
UINT	32 位无符号整数
BOOL	用做真值和假值
BYTE	8 位无符号字符
PSTR	指向字符串的 32 位指针
PCSTR	指向字符串的常量指针



#### 提示

句柄实际上就是一个标识资源的值，其最重要的特性是同一时间不会有二个资源的句柄值是相同的。Windows 系统包含有许多句柄类型，如应用程序实例、窗口、菜单、控件、输出设备及画笔等，所有的句柄类型都以 H 打头。通过句柄，应用程序才能访问信息，才能借助系统完成实际工作，这是 Windows 系统在多任务环境下保护信息的一种手段。

## 17.1.2 API 常见数据结构

Win32 API 中除了基本的数据类型外，还定义了几种常见的数据结构，如表所示。

数据结构	描述
MSG	应用程序消息的结构
WNDCLASS	定义窗口类
PAINTSTRUCT	定义窗口用户域的绘制消息
RECT	定义矩形

## 17.1.3 API 浏览器

由于 API 函数涵盖计算机操作的各个方面，数量庞大，记住全部 API 函数既不可能，也没有必要，所以在需要的时候查看 API 帮助就足够了。

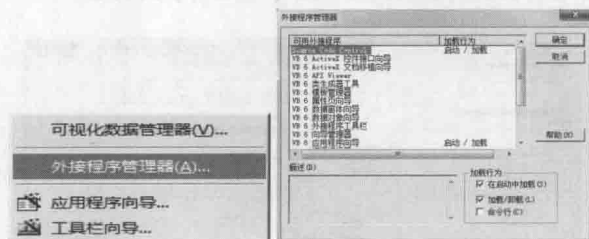
API 浏览器用于浏览关于 API 函数的常数、声明和类型等，可以把它的内容复制到剪贴板中，然后在 Visual Basic 6.0 的代码窗口中粘贴即可使用。下面介绍如何使用 API 浏览器。

### 1. 启动 API 浏览器

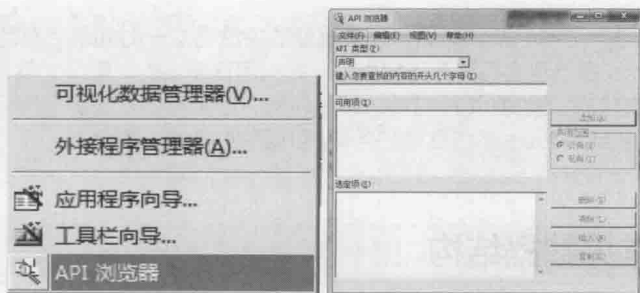
在 Visual Basic 6.0 中，API 浏览器默认是没有加载的，启动方法如下。

(1) 在 Visual Basic 6.0 窗口中，选择【外接程序】>【外接程序管理器】菜单命令。

(2) 弹出【外接程序管理器】窗口，选中【可用外接程序】列表框中的“VB 6 API Viewer”选项，选中【在启动中加载】和【加载 / 卸载】两个复选框，然后单击【确定】按钮。



(3) 这样就把 API 浏览器加载到【外接程序】菜单中了。选择【外接程序】>【API 浏览器】菜单命令，即可打开 API 浏览器。



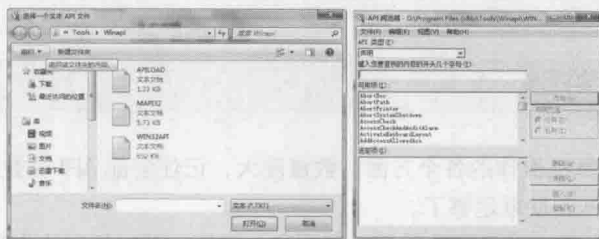
## 2. 在 API 浏览器中加载文本文件

API 函数的相关信息，包括常数、声明和类型等，存放在两个文本文件，即“WIN32API.txt”和“MAPI32.txt”中。“WIN32API.txt”中含有 Windows API 函数的 32 位版本的常量、声明和类型，而“MAPI32.txt”中则含有 Windows 多媒体 API 函数的常量、声明和类型。

下面介绍在 API 浏览器中加载“WIN32API.txt”的方法。

- (1) 选择【文件】>【加载文本文件】菜单命令。
- (2) 弹出【选择一个文本 API 文件】对话框，选择“WIN32API.txt”文件，单击【打开】按钮。

加载“WIN32API.txt”文件后的 API 浏览器如图所示。

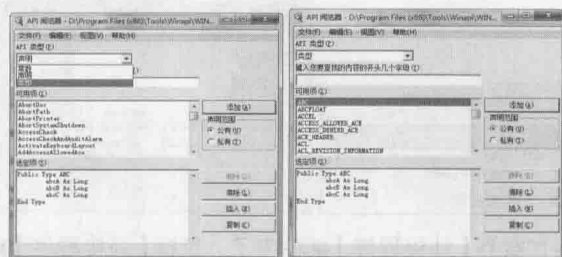


## 3. 使用 API 浏览器

把文本文件加载到 API 浏览器中后，就可以使用 API 浏览器查看 API 函数的常数、声明或类型等信息。使用 API 浏览器的一般步骤如下。

- (1) 在 API 浏览器中的【API 类型】下拉列表中选择“常数”、“声明”或“类型”选项，即可在【可用项】列表框中列出相应的项目。

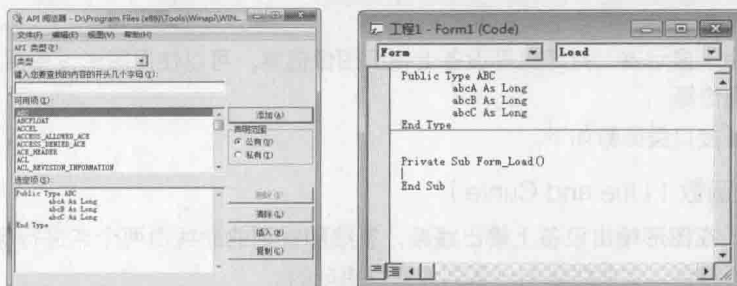
- (2) 在【可用项】列表框中选择要复制的项目后，在【声明范围】中选中【公有】或【私有】单选按钮，然后单击【添加】按钮，即可将要复制的项目添加到【选定项】列表框中。



- (3) 单击【复制】按钮，即可把选定项的代码复制到剪贴板中。

- (4) 在 Visual Basic 的代码窗口中，选择【编辑】>【粘贴】菜单命令，即可把选定项的代码粘贴到

代码窗口中。



单击【添加】按钮后，再单击【插入】按钮，即可直接将选定项的代码插入到 VB 代码窗口中的声明部分。

技巧

## 17.2 API 的函数分类



本节视频教学录像：14 分钟

Win32 API 函数按照功能可以分为窗口管理类函数、窗口通用控制类函数、Shell 特性类函数、图形设备接口类函数、系统服务类函数、国际特性类函数和网络服务类函数等。下面简单介绍几个 API 函数。

### 17.2.1 窗口管理类函数

一般来说，一个应用软件至少需要一个窗体。窗体用于软件 and 用户间的交互，例如向用户显示程序的执行信息和结果，让用户提供程序运行所需参数等。我们可以使用窗口管理类函数向应用程序提供一些用于创建和管理用户界面的方法。下面简要列出窗口管理类的函数。

#### 1. 按钮函数 (Button)

很多程序需要由用户和软件的交互来实现各种功能。按钮就是一种控制，用户可通过单击按钮向应用程序提供输入信息。

#### 2. 组合框函数 (ComboBox)

组合框是由 ComboBox 类定义的一种控制，综合了列表框和编辑控制的很多功能。使用组合框函数可以在组合框中显示或获取不同类型的数据。

#### 3. 通用对话框函数 (Common Dialog Box)

通用对话框是在通用对话框库中定义的，其功能是用来完成一些通用的任务，比如打开文件、打印文档等。通用对话框为用户提供了一个统一的用户界面，使得用户对软件的操作能够驾轻就熟。

#### 4. 对话框函数 (Dialog Box)

对话框是应用程序创建的一个临时窗口，用于获取用户的输入。应用程序通常使用对话框向用户显示一些命令提示信息。一个对话框一般由一个或多个子窗口组成，可用来输入文本、选择选项或执行命令动作。

#### 5. 菜单函数 (Menu)

菜单函数向 Win32 应用程序提供了一系列创建、管理和使用菜单的方法，包括对菜单条、菜单项等的处理。

## 17.2.2 图形设备接口类函数

图形设备接口用于显示器、打印机等设备上输出图像信息。可以使用图形设备接口类函数来绘制图像、文本以及位图图像等。

常见的图形设备接口类函数如下。

### 1. 直线和曲线函数 (Line and Curve)

直线和曲线用于在图形输出设备上输出线条, 直线和规则的曲线由两个点进行标识: 起点和终点。不规则曲线则是由不符合二次曲线段的一系列像素点组成的。

### 2. 图形填充函数 (Filled Shape)

图形填充函数用于对填充图形进行操作。图形填充函数所绘制的是一些由当前的画笔绘制其轮廓、内部由当前的笔刷进行填充的几何图形。共有椭圆、弦图、饼图、多边形和矩形等 5 种填充图形。

### 3. 绘图和画图函数 (Painting and Drawing)

绘图和画图函数为应用程序提供了一些在窗口中绘图, 以及创建和使用显示设备环境的方法。

### 4. 路径函数 (Path)

该函数用于创建、改变和绘制路径。一个路径是指一个或多个被填充、被绘制轮廓, 或者既被填充又被绘制轮廓的图形 (或形状)。

### 5. 矩形函数 (Rectangle)

该函数用于对矩形进行操作。Win32 应用程序使用矩形来指定显示屏幕上或窗口中的一个矩形区域。

### 6. 区域函数 (Region)

区域函数用于对区域进行操作。区域是指一个可被填充、着色、转换和加外框的任意几何图形, 包括矩形、多边形或椭圆 (或这几种形状的组合), 用于完成击键测试 (测试光标位置)。

### 7. 坐标空间及映射函数 (Coordinate Space and Transformation)

Win32 应用程序使用坐标空间和映射函数对输出的图形进行比例缩放、旋转、转换、剪裁和反射等操作。



**提示**

坐标空间是基于笛卡尔坐标系统的一个平面空间。该坐标系统要求有两个垂直相交、长度相等的坐标轴。共有 4 种坐标空间: 现实坐标、页面坐标、设备坐标和物理设备坐标 (显示区, 或桌面打印纸的页面)。映射方式就是改变 (“映射”) 对象的大小、方向和形状的一种算法。

## 17.2.3 系统服务类函数

系统服务类函数提供了访问文件、目录以及输入输出 (I/O) 设备的手段。应用程序通过系统服务类函数访问计算机资源以及底层操作系统特性, 如访问内存、文件系统、设备、进程和线程等。应用程序可以使用系统服务类函数来管理和监视它所需要的资源。例如, 应用程序可以使用内存管理函数来分配和释放内存, 使用进程管理和同步函数来启动和调整多个应用程序, 或在一个应用程序中运行的多个线程的操作。

### 1. 访问控制函数 (Access Control)

Microsoft Windows NT 所提供的安全功能对 Win32 应用程序是自动使用的。在系统中运行的每个



应用程序，都会受到 Windows NT 特殊配置所提供的安全功能的影响。Windows NT 是支持 Win32 安全功能的唯一平台。Windows NT 的安全功能对大多数 Win32 函数的影响都是最小的，不需要安全功能的 Win32 应用程序不必合并任何特殊代码。不过，可以使用 Windows NT 的安全属性向 Win32 应用程序提供一些服务。访问控制函数提供了一系列控制访问 Win32 对象（比如文件）、管理函数（比如设置系统时间或审核运行动作的函数）的 Windows NT 安全模型。

## 2. 原子函数（Atom）

利用该函数可以进行一系列对原子的添加、删除、初始化等操作。原子表格是一个系统定义的表格，用于保存字符串和相应的标识符。应用程序将一个字符串放到原子表格中，并接受一个 16 位的整数（称为一个原子），用于访问该字符串。放到原子表格中的字符串被称为原子名字。

## 3. 客户服务器访问控制函数（Client/Server Access Control）

客户/服务器访问控制函数包括 3 类：用于模拟客户机、用于检查和设置私有对象上的安全描述符、用于生成安全时间日志中的审核消息。

## 4. 剪贴板函数（Clipboard）

剪贴板是由一系列函数和消息组成的，Win32 应用程序可使用它来传输数据。由于所有的应用程序都可以访问剪贴板，所以数据可以很容易地在应用程序之间或一个应用程序内部进行传输。

## 5. 通信函数（Communication）

通信资源是一个物理或逻辑设备，用于提供双向的异步数据流，例如串行端口、并行端口、传真机以及调制解调器等都是通信资源。对应每个通信资源都有一个服务供应程序（包含一个库或驱动程序），使得应用程序可以访问该资源。通信函数是通信设备所使用的函数。

# 17.2.4 国际特性类函数

如果想让自己编写的软件被更多的人使用，就必须考虑到国际化的问题。Unicode 字符集使用 16 位的字符值来表示计算过程中所用的字符，比如各种符号以及很多编程语言。国家语言支持（NLS）函数可以帮助用户将应用程序本地化，输入方法编辑器（IME）函数（在 Windows 亚洲版中可用）用于帮助用户输入包含 Unicode 和 DCBS 字符的文本。

## 1. 输入方法编辑器函数（Input Method Editor）

输入方法编辑器函数用于创建和管理 IME 窗口。使用输入方法编辑器（IME）有助于简化用户的文本输入过程（文本中包含 Unicode 字符和双字节字符 DBCS）。

## 2. 国家语言支持函数（National Language Support）

使用国家语言支持函数可以使 Win32 应用程序支持世界各地的不同语言，以满足不同地区用户的特殊需要。

## 3. Unicode 和字符集函数（Unicode and Character Set）

Win32 API 通过 Unicode 和传统字符集可以支持世界各国的不同语言。

**提示**

Unicode 是一种在计算机上使用的字符编码。它为每种语言中的每个字符设定了统一并且唯一的二进制编码,以满足跨语言、跨平台进行文本转换、处理的要求。1990 年开始研发,1994 年正式公布。Unicode 也在面世以来的 10 多年里得到普及。Unicode 作为世界通用的字符编码标准,使用 16 位的字符值来表示各种字符,包括技术符号和出版所用的特殊字符。传统字符集是指以前所用的字符编码标准,比如 Windows ANSI 字符集,它是使用 8 位的字符值或 8 位值的组合来表示在指定的语言或地理区域中所用的字符。

## 17.2.5 网络服务函数

网络服务是现代计算机中非常重要的服务之一,计算机通过网络服务函数与其他的计算机或设备进行通信。网络服务函数用于在网络中的各计算机上创建和管理共享资源的连接。网络接口包括 Windows 网络函数、Windows 套接字 (Socket)、NetBIOS、RAS、SNMP、Net 函数以及网络 DDE。

### 1. DLC 函数 (DLC)

数据连接控制 (DLC) 接口函数是为运行 Windows 或 Windows NT 的计算机与 IBM 主机或直接连接到网络上的打印机之间提供连通性的函数。

### 2. 网络函数 (Net)

Windows XP、Windows 2003 以及早期的 Windows NT、Windows 95 和 Windows 98 等支持多种网络函数。Net 函数集提供了一些其他网络函数的函数覆盖功能。另外,还可以使用这些函数来监测和管理基于 OS/2 的 LAN Manager 服务器。

### 3. NetBIOS 函数

该函数用于应用程序与网络中的其他计算机上的应用程序进行通信。NetBIOS 接口包括一系列显式命令,由一个被称为网络控制块 (NCB) 的结构提供。应用程序可以对任何支持 NetBIOS 接口的协议发出 NetBIOS 命令。

### 4. 网络 DDE 函数 (Networking DDE)

一个进程可以使用 Win32 API 提供的网络动态数据交换 (DDE) 函数与在网络中的不同计算机上运行的进程建立会话。

### 5. RAS 服务器管理函数 (RAS Server Administration)

在 Windows NT 4.0 上,可使用 RAS 服务器管理函数来实现 RAS 服务器管理功能。Windows 95 不提供 RAS 服务器支持。

## 17.3 API 函数的应用



本节视频教学录像: 3 分钟

在使用 API 函数前首先要声明它。可以使用 Declare 语句手动声明 API 函数,或者使用 API 浏览器声明 API 函数。

## 17.3.1 使用 Declare 语句手动声明 API 函数

如果使用 Declare 语句手动声明 API 函数，则需要在代码窗口的【通用】部分添加 Declare 语句。语法如下。

```
[Public | Private] Declare Sub name Lib "libname" [Alias "aliasname"] [[[arglist]]]
```

或：

```
[Public | Private] Declare Function name Lib "libname" [Alias "aliasname"] [[[arglist]]] [As type]
```



**提示**

如果声明的过程不需要返回一个值，则应选择第 1 种语法格式；如果声明的过程需要返回一个值，则应选择第 2 种语法格式。

上述语法中的 Declare 语句的含义如表所示。

部分	含义
Public	可选参数，用于声明对所有模块中的所有其他过程都可以使用的过程
Private	可选参数，用于声明只能在包含该声明的模块中使用的过程
Sub	可选参数，表示该过程没有返回值
Function	可选参数，表示该过程会返回一个可用于表达式的值
name	必需参数，为任何合法的过程名。注意动态链接库的入口处（Entry Points）区分大小写
Lib	必需参数，用于指明包含所声明过程的动态链接库或代码资源。所有声明都需要 Lib 子句
libname	必需参数，包含所声明的过程动态链接库名或代码资源名
Alias	可选参数，表示将被调用的过程在动态链接库（DLL）中还有另外的名称。当外部过程名与某个关键字重名时，就可以使用这个参数。当动态链接库的过程与同一范围内的公用变量、常数或任何其他过程的名称相同时，也可以使用 Alias。如果该动态链接库过程中的某个字符不符合动态链接库的命名约定时，也可以使用 Alias
aliasname	可选参数，表示动态链接库或代码资源中的过程名。如果首字符不是数字符号（#），则 aliasname 是动态链接库中该过程的入口处的名称。如果首字符是（#），则随后的字符必须指定该过程的入口处的顺序号
arglist	可选参数，代表调用该过程时需要传递的参数的变量表
type	可选参数，表示 Function 过程返回值的数据类型；可以是 Byte、布尔、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）或 Variant，以及用户定义类型或对象类型

arglist 参数的语法格式如下。

```
[Optional] [ByVal | ByRef] [ParamArray] varname[( )] [As type]
```

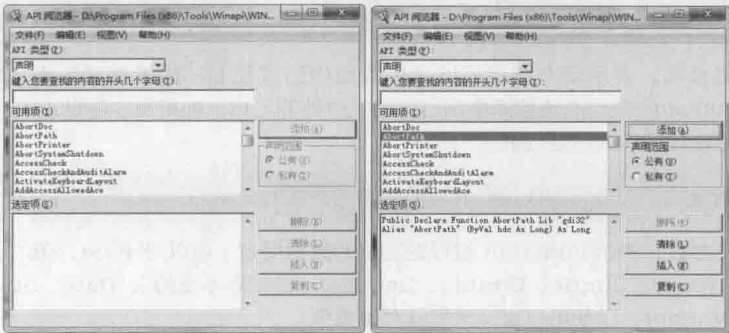
arglist 参数语法包含部分及描述如表所示。

部分	含义
Optional	可选参数，用于表示参数不是必需的。如果使用该项，则 arglist 中的后续参数都必须是可选的，而且必须都使用 Optional 关键字声明。如果使用了 ParamArray，则任何参数都不能使用 Optional
ByVal	可选参数，表示该参数按值传递
ByRef	可选参数，表示该参数按地址传递。ByRef 是 Visual Basic 的默认选项
ParamArray	可选参数，只用于 arglist 的最后一个参数，表示最后的参数是一个 Variant 元素的 Optional 的数组。使用 ParamArray 关键字可以提供任意数目的参数。ParamArray 关键字不能与 ByVal、ByRef 或 Optional 一起使用
varname	必需参数，代表传给该过程的参数的变量名；遵循标准的变量命名约定
( )	对数组变量是必需的。指明 varname 是一个数组
type	可选参数，表示传递给该过程的参数的数据类型；可以是 Byte、Boolean、Integer、Long、Currency、Single、Double、Decimal（目前尚不支持）、Date、String（只支持变长）、Object、Variant、用户自定义的类型或对象类型

### 17.3.2 使用 API 浏览器声明 API 函数

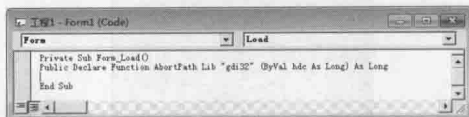
如果觉得用语句声明 API 函数不够直观，用户也可以使用 Visual Basic 中的 API 浏览器来声明 API 函数。具体操作步骤如下。

- (1) 使用 API 浏览器打开“WIN32API.txt”文件。
- (2) 双击【可用项】列表框中的项目，该项的代码即会出现在【选定项】列表框中，然后单击【复制】按钮，将选取的项目代码复制到剪贴板中。



- (3) 在 Visual Basic 的代码窗口中选择【编辑】>【粘贴】菜单命令，即可将函数的声明粘贴到代码

窗口中。



### 17.3.3 API 函数的调用

在 Visual Basic 中，API 函数的调用方式有两种：直接调用和 CALL 调用。

下面以 Windows 函数 FlashWindow（使给定的窗口闪烁一次）为例，介绍 Visual Basic 中 Windows API 的调用方式。

FlashWindow 函数原型：

```
BOOL FlashWindow(hwnd,flInvert)
```

Visual Basic 声明：

```
Private Declare Function AAA FlashWindow Lib "user32" Alias "FlashWindow" (ByVal hwnd As Long,
ByVal blInvert As Long) As Long
```

调用过程：

```
abc=AAA(hwnd,CLng(True)) ' 开始闪烁（直接调用方式）
```

或：

```
Call AAA(hwnd,CLng(True)) 'CALL 调用方式
```



**技巧**

Visual Basic (VB) 作为一种高效编程环境，它封装了部分 Windows API 函数，但也牺牲了一些 API 的功能。调用 API 时稍有不慎就可能导致 API 编程错误，出现难于捕获或间歇性错误，甚至出现程序崩溃。所以要减少 API 编程错误，提高 VB 调用 API 时的安全性。


## 17.4 插上翅膀去飞翔——API 编程实例



本节视频教学录像：3 分钟

本节通过一个实例来学习如何使用 API 编程。

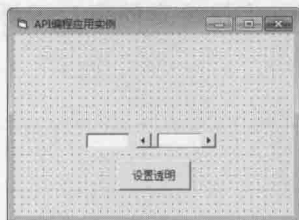
### 【范例 17-1】使用 API 函数控制窗体的透明度。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 把 Form1 的 Caption 属性设置为“API 编程应用实例”。



(3) 在窗体中添加 1 个文本框控件, 名称为 “Text1”; 添加 1 个水平滚动条控件, 名称为 “ScrolBar1”, Min 属性值为 50, Max 属性值为 255; 添加 1 个按钮控件, 名称为 “Command1”, Caption 属性值为 “设置透明”。将各控件按照下图排列整齐。



(4) 右击窗体空白处, 在弹出的快捷菜单中选择【查看代码】选项, 进入代码窗口, 添加以下代码 (代码 17-1-1.txt)。

```
01 Private Declare Function SetLayeredWindowAttributes Lib "user32" (ByVal hwnd As Long, ByVal
  crKey As Long, ByVal bAlpha As Byte, ByVal dwFlags As Long) As Long
    ' 定义设置窗体透明度的 API 函数
02 Const WS_EX_LAYERED = &H80000
03 Const GWL_EXSTYLE = (-20)
04 Const LWA_ALPHA = &H2
05 Const LWA_COLORKEY = &H1
06 Private Declare Function GetWindowLong Lib "user32" Alias "GetWindowLongA" (ByVal hwnd As
  Long, ByVal nIndex As Long) As Long      ' 定义获得指定窗口信息的 API 函数
07 Private Declare Function SetWindowLong Lib "user32" Alias "SetWindowLongA" (ByVal hwnd As
  Long, ByVal nIndex As Long, ByVal dwNewLong As Long) As Long      ' 定义修改指定窗口属性的 API 函数
```

(5) 双击 ScrolBar1 滚动条控件, 在弹出的代码窗口中添加以下代码。

```
01 Private Sub ScrolBar1_Change()      ' 滚动条改变事件
02 Text1.Text = ScrolBar1.Value      ' 将滚动条的值传给 Text1
03 End Sub
```

(6) 双击 Command1 命令按钮控件, 在弹出的代码窗口中添加以下代码 (代码 17-1-2.txt)。

```
01 Private Sub Command1_Click()
02 x = Text1.Text      ' 将 Text1 的内容传给 x
03 Dim sty As Long
04 sty = GetWindowLong(Me.hwnd, GWL_EXSTYLE)
05 sty = sty Or WS_EX_LAYERED
06 SetWindowLong Me.hwnd, GWL_EXSTYLE, sty
07 SetLayeredWindowAttributes Me.hwnd, 0, x, LWA_ALPHA      ' 设置窗体透明度
```



08 End Sub

## 【运行结果】

按快捷键【F5】运行程序。拖动滚动条，滚动条的值会在文本框中显示。单击【设置透明】按钮，文本框中的数值传递给参数 x，并且窗体以不透明度值为 x 透明显示。



## 【代码详解】

SetLayeredWindowAttributes 函数用于设置透明窗体，要使用它必须声明 GetWindowLong 和 SetWindowLong 函数。步骤(4)首先声明这 3 个函数。

步骤(5)将滚动条的值在文本框中显示出来。

步骤(6)首先将文本框的文本值传递给参数 x，然后调用 SetLayeredWindowAttributes 函数，以不透明度为 x 进行透明显示。

## 【拓展训练 17-1】API 编程应用实例 2。

在【范例 17-1】中拖动滚动条后，需要单击命令按钮，窗体才能显示透明效果。如果要拖动改变滚动条的数值后，窗体就透明显示，应该怎么办呢？

将步骤(5)的代码修改如下。

```
01 Private Sub ScrolBar1_Change() ' 滚动条改变事件
02 Text1.Text = ScrolBar1.Value ' 将滚动条的值传给 Text1
03 x = Text1.Text ' 将 Text1 的内容传给 x
04 Dim sty As Long
05 sty = GetWindowLong(Me.hwnd, GWL_EXSTYLE)
06 sty = sty Or WS_EX_LAYERED
07 SetWindowLong Me.hwnd, GWL_EXSTYLE, sty
08 SetLayeredWindowAttributes Me.hwnd, 0, x, LWA_ALPHA ' 设置窗体透明度
09 End Sub
```

这样，只要滚动条的数值改变，不需要单击命令按钮，窗体也会自动透明显示。

## 17.5 高手点拨



本节视频教学录像：3 分钟

通过 API 文本查看器，我们可以方便地查找程序所需要的函数声明、结构类型和常数，然后将它复制到剪贴板，最后再粘贴到 VB 程序的代码段中。在大多数情况下，只要我们确定了程序所需要的函数、结构和常数这三个



方面后,就可以通过对 API 文本浏览器的以上操作将他们加入到程序段中,从而程序中可以使用这些函数了。

不是所有的 API 函数都有别名,选用 Alias 的时候,应注意别名的大小写;如果不选用 Alias 时,函数名必须注意大小写,而且不能改动。当然,在很多情况下,由于函数声明是直接来自 API 文本浏览器中拷贝过来的,所以这种错误的发生机会是很少的,但您有必要知道这一点。最后提醒您一句,API 声明(包括结构、常数)必须放在窗体或模块的“通用 (General Declarations)”段。

API 函数中使用的数据类型基本上和 VB 中的一样。但作为 Win32 的 API 函数中,不存在 Integer 数据类型。另外一点是在 API 函数中看不到 Boolean 数据类型。Variant 数据类型在 API 函数中是以 Any 的形式出现,如 Data As Any。尽管其含义是允许任意参数类型作为一个该 API 函数的参数传递,但这样做存在一定的缺点。其原因是,这将会使得对目标参数的所有类型检查都会被关闭。这自然会给各种类型的参数调用带来了产生错误的机会。

## 17.6 实战练习

### 一、思考题

1. 简述 Win32 API 函数与 Visual Basic 函数的不同。
2. 写出使用 Declare 语句手动声明 API 函数的语法。

### 二、操作题

打开随书光盘中的“Sample\ch17\跟我上机 VAPI 编程控制光驱的开关.vbp”工程文件,补充完整代码,完成以下功能。

- (1) 单击“打开光驱”按钮,光盘驱动器弹出,按钮标题变为“关闭光驱”;
- (2) 单击“关闭光驱”按钮,光盘驱动器关闭,按钮标题变为“打开光驱”。

提示: CDdoor 函数用于控制光驱的打开和关闭,声明语句如下。

---

```
Private Declare Function CDdoor Lib "winmm.dll" Alias "mciSendStringA" _
    ( _
    ByVal lpstrCommand As String, _
    ByVal lpstrReturnString As String, _
    ByVal uReturnLength As Long, _
    ByVal hwndCallback As Long _
    ) As Long
```

---

引用的语法如下,关闭光盘驱动器。

---

```
Call CDdoor("set CDAudio door closed", 0, 0, 0)
```

---

打开光盘驱动器。

---

```
Call CDdoor("set CDAudio door open", 0, 0, 0)
```

---

# 第18章



本章视频教学录像：31 分钟

## Visual Basic 中的网络世界——网络编程

随着计算机技术的不断成熟和发展，新技术在网络中的应用也呈现出欣欣向荣、日新月异的局面。本章通过邮件应用编程、互联网传输应用及自定义网页浏览器的应用实例，讲述 Visual Basic 在网络中的编程技术。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解邮件程序的发送与接收
- ☐ 熟悉互联网传输的编程思想
- ☐ 了解网页浏览器的编程控件

## 18.1 邮件应用编程



本节视频教学录像: 11 分钟

电子邮件是指用电子手段传送信件、单据和资料等信息的通信形式。电子邮件能够像电话一样瞬间将信息传递到目的地, 同时又能像信件一样发送图文并茂的各种信息。

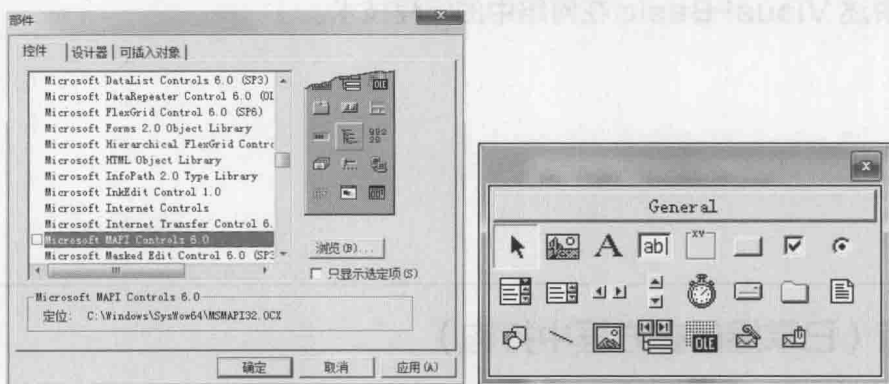
### 18.1.1 邮件程序接口控件的属性和方法

若要创建出具有电子邮件功能的 Visual Basic 应用程序, 程序员可以使用 Visual Basic 中的邮件应用程序接口控件 (MAPI)。在 Visual Basic 中提供有两个 MAPI 控件: MAPISession 控件和 MAPIMessages 控件。其中 MAPISession 控件用于管理一个 MAPI 会话, MAPIMessages 控件用于执行消息功能。

在了解 MAPI 控件的各种属性和方法之前, 首先需要把 MAPI 控件添加到工具箱中。具体的操作步骤如下。

(1) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令, 在弹出的【部件】对话框中选择【控件】选项卡, 选择【Microsoft MAPI Controls 6.0】复选框, 然后单击【确定】按钮。

(2) 可以发现工具箱中多出了两个 MAPI 控件, 即 MAPISession 控件和 MAPIMessages 控件。



#### 1. MAPISession 控件的属性

(1) SessionID 属性。该属性用于返回当前的会话 (Session) 句柄。当我们成功地建立了会话后, SessionID 属性将返回句柄, 这个句柄是唯一的, 不会互相重复。SessionID 的默认值为 0。该属性的语法如下。

```
object.SessionID
```

(2) Action 属性。该属性用于选择启动和结束会话。该属性的语法如下。

```
object.Action [ = value ]
```

(3) DownloadMail 属性。该属性用于指定在当前会话开始时是否自动下载电子邮件。

(4) LogonUI 属性。该属性用于设置是否在启动会话时显示登录对话框。



(5) NewSession 属性。该属性用于设置是否建立一个新的邮件会话，即使现在已存在一个有效的会话，也可以再建一个会话。

(6) UserName 属性。该属性指定账户用户名，或者建立会话所使用的配置文件。

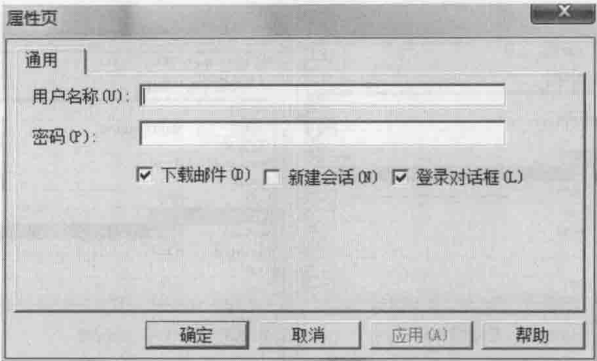


(7) Password 属性。该属性用于设置和 UserName 属性所对应的账户密码。



提示

用户也可以右击 MAPISession 控件，在弹出的快捷菜单中选择【属性】菜单项，然后在弹出的【属性页】对话框中设置 DownloadMail、NewSession、UserName 和 Password 等属性。



2. MAPISession 控件的方法

(1) SignOff 方法

该方法用于终止消息会话，从用户当前所登录的账户中退出。语法如下。

object.SignOff

(2) SignOn 方法

该方法用于登录到由用户所输入的 UserName 和 Password 属性所指定的账户中，并返回一个会话句柄给当前的消息系统。语法如下。

object.SignOn

3. MAPIMessages 控件的属性

(1) SessionID 属性。该属性用于存储当前会话的句柄。该属性的值来自 MAPISession 控件的 SessionID 属性返回的消息会话句柄。语法如下。

object.SessionID [ = value ]

(2) Action 属性。该属性用于设置在 MAPIMessages 控件被激活时将进行什么操作。语法如下。

object.Action [ = value ]

该语法中的 value 参数的取值及其对应的操作如表所示。

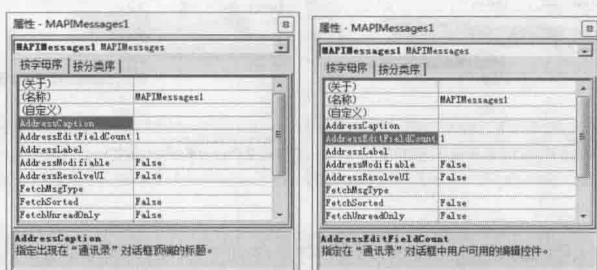
value 值	对应的操作
MESSAGE_FETCH	获取
MESSAGE_SENDDLG	发送
MESSAGE_SEND	发送
MESSAGE_SAVEMSG	保存
MESSAGE_COPY	复制
MESSAGE_COMPOSE	构成

续表

value 值	对应的操作
MESSAGE_REPLY	应答
MESSAGE_REPLYALL	全应答
MESSAGE_FORWARD	转发
MESSAGE_DELETE	删除
MESSAGE_SHOWADBOOK	显示
MESSAGE_SHOWDETAILS	显示
MESSAGE_RESOLVENAME	分析名字
RECIPIENT_DELETE	删除
ATTACHMENT_DELETE	删除

(3) AddressCaption 属性。该属性用于指定【通讯录】对话框的标题。

(4) AddressEditFieldCount 属性。该属性用于指定在【通讯录】对话框中为最终用户显示哪些编辑控件。



(5) AddressLabel 属性。该属性用于指定在【通讯录】对话框中【To】编辑控件的外观。

(6) AddressModifiable 属性。该属性用于设置用户是否可以编辑【地址簿】。



(7) AddressResolveUI 属性。该属性用于设置在指定 ResolveName 方法后，在处理收件人姓名时是否显示分辨收件人名称对话框。

(8) FetchMsgType 属性。该属性用于指定消息类型。



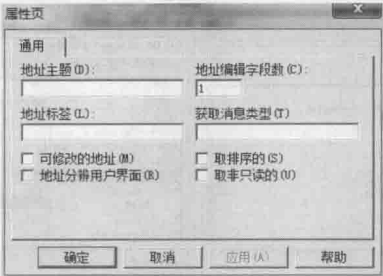
(9) FetchSorted 属性。该属性用于指定消息的顺序。当 FetchSorted 属性值为 True 时，消息按照被接收的顺序加到消息集；当 FetchSorted 属性值为 False 时，消息按照用户收件箱中指定的排序加到消息集。

(10) FetchUnreadOnly 属性。该属性用于设置消息是否可读。当该属性值设置为 True 时，只有 FetchMsgType 属性中指定的不可读消息会被添加到消息集中；当该属性值设置为 False 时，收件箱中所有类型的消息都会被添加到消息集中。



提示

用户也可以在 MAPIMessages 控件上右击，在弹出的快捷菜单中选择【属性】菜单项，然后在弹出的【属性页】对话框中设置 AddressCaption、AddressEditFieldCount、AddressLabel、AddressModifiable、AddressResolveUI、FetchSorted 和 FetchUnreadOnly 等属性。



(11) 文件附件属性、消息相关属性和收件人信息属性。

Visual Basic 中邮件附件管理所需的属性如表所示。



属性名称	功能	语法
AttachmentCount	用于返回与当前编号消息相关联的附件的总数	object.AttachmentCount
AttachmentIndex	用于设置当前编号的附件	object.AttachmentIndex [ = value]
AttachmentName	用于设置当前编号文件附件的名称	object.AttachmentName [ = value]
AttachmentPathName	用于设置当前编号文件附件的完整路径	object.AttachmentPathName [ = value]
AttachmentPosition	用于设置在消息体中当前编号的附件位置	object.AttachmentPosition [ = value]
AttachmentType	用于设置当前编号文件附件的类型	object.AttachmentType [ = value]

#### 4. MAPIMessages 控件的方法

(1) Compose 方法。MAPI 使用该方法构成一条新消息。语法如下。

```
object.Compose
```

(2) Copy 方法。该方法用于将当前选中的邮件复制到缓冲区，并将 MsgIndex 属性置为 -1。语法如下。

```
object.Copy
```

(3) Delete 方法。该方法用于删除附件、消息或者收件人。语法如下。

```
object.Delete [ value ]
```

(4) Save 方法。该方法用于保存当前缓冲区中的数据。语法如下。

```
object.Save
```

(5) Forward 方法。该方法用于转发邮件，将当前选中的邮件作为一个转发邮件复制到缓冲区，并在邮件主题前面加上“FW:”。语法如下。

```
object.Forward
```

(6) Send 方法。该方法用于发送邮件。语法如下。

```
object.Send [ value ]
```

(7) Reply 方法和 ReplyAll 方法。Reply 方法用于回复邮件，该方法将当前选中的邮件复制到缓冲区中，并在邮件主题前面加上“RE”。语法如下。

```
object.Reply
```

ReplyAll 方法用于回复所有收件人。语法如下。

object.ReplyAll

(8) ResolveName 方法。该方法用于查找通讯簿中是否有匹配收件人名称的信息。语法如下。

object.ResolveName

(9) Show 方法。该方法用于显示通讯簿中相关记录的详细信息。语法如下。

object.Show [ value ]

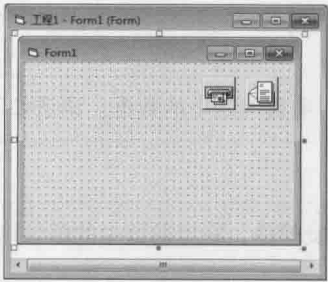
### 18.1.2 实现邮件发送

对邮件应用程序接口有了一定的了解之后，下面通过一个实例来学习如何发送一封邮件。

**【范例 18-1】发送一封文本邮件。用 VB 编写程序，不用登录邮箱，只需几行代码就可以发送一封文本邮件。**

第 1 步：设置界面

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在打开的 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令，弹出【部件】对话框。
- (3) 在【控件】选项卡中选中【Microsoft MAPI Controls 6.0】复选框，单击【确定】按钮。
- (4) 在窗体中添加 1 个 MAPISession 控件和 1 个 MAPIMessages 控件。



- (5) 在窗体中添加 4 个标签（Label）控件，并按照下表所示设置其属性。

控件名称	Caption	控件作用
Label1	收件人姓名	提示用户输入收件人姓名
Label 2	收件人地址	提示用户输入收件人地址
Label3	邮件主题	提示用户输入邮件主题
Label4	邮件正文	提示用户输入邮件正文



(6) 在窗体中添加 4 个文本框 ( TextBox ) 控件，并按照下表所示设置其属性。

名称	MultiLine	Scroll Bars	Text	控件作用
TxtName	False	0- VbSBNone	“ ”	用于用户输入收件人姓名
TxtAddress	False	0- VbSBNone	“ ”	用于用户输入收件人地址
TxtSubject	False	0- VbSBNone	“ ”	用于用户输入邮件主题
TxtNote	True	3- Both	“ ”	用于用户输入邮件正文



(7) 在窗体中添加 4 个命令按钮 ( CommandButton ) 控件，并按照下表设置其属性。

名称	Caption	控件作用
CmdLogon	登录	用于用户登录
CmdLogOff	注销	用于用户账号注销
CmdSend	发送	用于发送邮件
CmdExit	退出	退出程序



## 第2步：编写代码

(1) 在 Form1 窗体的空白处双击，在打开的代码窗口中输入以下代码。

```
01 Private Sub Form_Load() ' 登录初始化
02     TxtName.Text = ""    ' 将收件人姓名文本框内容清空
03     TxtAddress.Text = "" ' 将收件人地址文本框内容清空
04     TxtNote.Text = ""    ' 将邮件主题文本框内容清空
05     TxtSubject.Text = "" ' 将邮件正文文本框内容清空
06 End Sub
```

(2) 在 Form1 窗体上双击【登录】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub CmdLogon_Click() ' 登录邮箱
02     MAPISession1.SignOn ' 用于登录到由用户所输入的 UserName 和 Password 属性所指定的账
                          ' 户中，并返回一个会话句柄给当前的消息系统
03     MAPIMessages1.SessionID = MAPISession1.SessionID ' 存储当前返回的会话句柄
04 End Sub
```

(3) 在 Form1 窗体上双击【注销】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub CmdLogOff_Click() ' 退出邮箱
02     MAPISession1.SignOff ' 用于取消由用户所输入的 UserName 和 Password 属性所指
                          ' 定的账户及密码，并返回一个会话句柄给当前的消息系统
03 End Sub
```


(4) 在 Form1 窗体上双击【发送】按钮，在打开的代码窗口中输入以下代码（代码 18-1-1.txt）。

```
01 Private Sub CmdSend_Click() ' 发送邮件
02     MAPIMessages1.Compose ' 建立新的 E-Mail Message
03     MAPIMessages1.RecipDisplayName = TxtName.Text ' 收件人 (Recipient's Name)
04     MAPIMessages1.RecipAddress = TxtAddress.Text ' 收件人的 E-Mail 地址
05     MAPIMessages1.MsgSubject = TxtSubject.Text ' E-Mail 的主题
06     MAPIMessages1.MsgNoteText = TxtNote.Text ' E-Mail 的内文
07     MAPIMessages1.Send False ' 发送 E-Mail
08     MsgBox "您已经发送成功!"
09 End Sub
```

(5) 在 Form1 窗体上双击【退出】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub CmdExit_Click()  
02 End '退出系统  
03 End Sub
```

## 【运行结果】

- (1) 保存程序，单击【启动】按钮 ，运行程序。
- (2) 单击【登录】按钮，然后填写收件人姓名、地址、主题和正文，最后单击【发送】按钮发送邮件。



如果在本地计算机中已进行过 Outlook 邮件设置，即可直接使用此程序发送邮件。



**技巧**

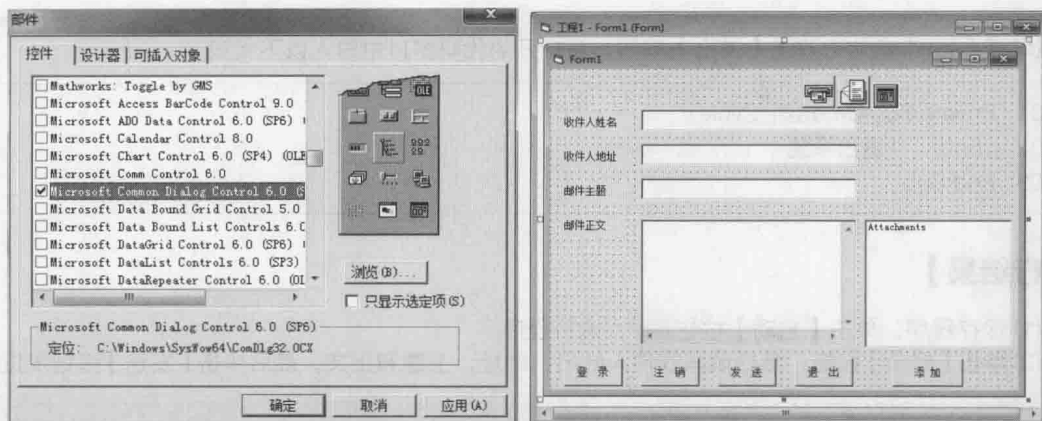
如果没有单击【登录】按钮而直接单击【注销】按钮，或者单击【登录】按钮一次后连续单击【注销】按钮，会出现会话标识符不存在的错误。因此在单击【注销】按钮之前，应确保用户已经登录。

## 【拓展训练 18-1】

如果在发送文字邮件的同时，需要发送附件，比如一张图片或者一个文件，则需要在【范例 18-1】的基础上，添加一个 CommonDialog 控件、一个列表框控件和一个命令按钮控件。

(1) 选择【工程】>【部件】菜单命令，在弹出的【部件】对话框中选择【控件】选项卡，选中【Microsoft Common Dialog Control 6.0】复选框，然后单击【确定】按钮。

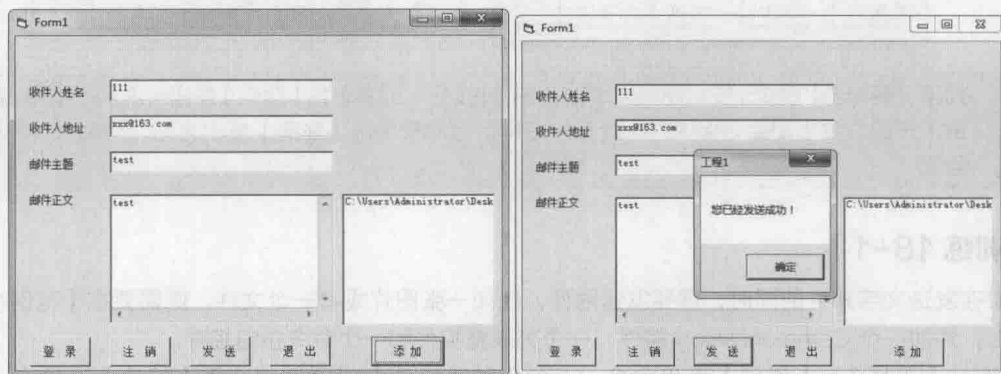
(2) 在窗体中添加 1 个对话框（CommonDialog）控件、1 个列表框（ListBook）控件和一个命令按钮（CommandButton）控件，设置列表框控件的【名称】属性为“Attachments”，设置命令按钮控件的【Caption】属性为“添加”。



(3) 在 Form1 窗体上双击【添加】按钮，在打开的代码窗口中输入以下代码（拓展代码 18-1.txt）。

```
01 Private Sub Command1_Click()
02     With CommonDialog1
03         .ShowOpen '打开通用对话框
04         If Len(.FileName) > 0 Then '判断是否选择了文件
05             Attachments.AddItem .FileName '如果选择了文件就添加文件
06         End If
07     End With
08 End Sub
```


(4) 按【F5】快捷键运行程序，单击【登录】按钮后填写收件人姓名、地址、主题和正文，同时添加所需要的附件，最后单击【发送】按钮发送邮件。



**提示**

使用 MAPIMessages 控件发送消息并不是真的发送了它，而是将它放在邮件系统的送件箱里。消息真正要在什么时候发送取决于邮件系统的设置。当你的程序完成与邮件相关的活动时，就要调用 MAPISession 控件的 SignOff 方法来结束会话。

## 18.2 互联网传输应用编程

 本节视频教学录像: 12 分钟

Visual Basic 中的互联网传输控件支持互联网上最为流行的两种网络传输协议: 超文本传输协议 (HTTP) 和文件传输协议 (FTP)。其中, 使用 HTTP 能够从 Web 服务器上获取 HTML 文档, 而使用 FTP 协议则可将文件上传或下载至 FTP 服务器上。



**提示**

超文本传输协议 (Hypertext Transfer Protocol, HTTP) 是用于从 WWW 服务器传输超文本到本地客户端浏览器的传送协议。它可以使浏览器更加高效, 使网络传输减少。它不仅能保证计算机正确快速地传输超文本文档, 还能确定传输文档中的哪一部分, 以及哪部分内容首先显示 (如文本先于图形) 等。我们平时上网的时候打开的网页地址都是以 “http://” 开头的原因就在于此。

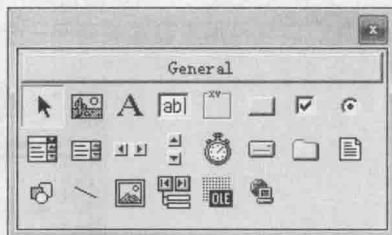
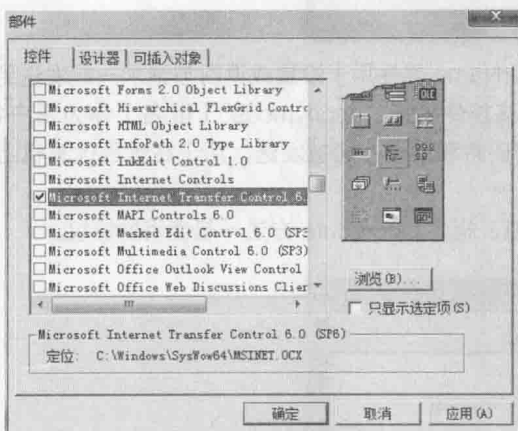
文件传输协议 (File Transfer Protocol, FTP) 是一个用于在两台装有不同操作系统的机器中传输计算机文件的软件标准。FTP 是一个 8 位的客户端—服务器协议, 能传输任何类型的文件而不需要进一步处理。

### 18.2.1 互联网传输控件的属性、事件和方法

在了解互联网传输控件 (Internet Transfer 控件) 的各种属性、事件和方法之前, 首先需要把互联网传输控件添加到工具箱中。具体的操作步骤如下。

(1) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令, 在弹出的【部件】对话框中选择【控件】选项卡, 选择【Microsoft Internet Transfer Control 6.0】复选框, 然后单击【确定】按钮。

(2) 这样就将互联网传输控件添加到了工具栏中。



#### 1. 互联网传输控件的主要属性

(1) 访问类型属性。该属性设置或返回一个值, 决定该控件用来与 Internet 进行通信的访问类型 (通过代理访问或直接访问)。正在处理异步请求时, 该值可以改变, 但直到创建了下一个连接时, 改变才会生效。





参数	值	含义
icUseDefault	0	使用注册表中的默认值访问 Internet
icDirect	1	直接连接到 Internet
icNameProxy	2	使用 Proxy 属性中指定的代理服务器

(2) 文档属性。该属性用于返回或设置与 Execute 方法一起使用的文件或文档。如果未指定该属性，将返回服务器中的默认文档。



(3) UserName 属性和 Password 属性。UserName 属性用于设置或返回与请求一起发送到远程计算机的名称。如果该属性为空，当提出请求时，该控件将把“anonymous”（匿名）作为用户名来发送。Password 属性用于设置或返回一个密码，该密码将与请求一起被发送，用于在远程计算机上登录。如果该属性为空，控件将发送一个默认密码。

(4) 协议属性。该属性用于设置或返回一个值，指定和 Execute 方法一起使用的协议。



其中，integer 参数的值及其所对应的含义如表所示。

参数	值	含义
icUnknown	0	未知协议
icDefault	1	默认协议
icFTP	2	FTP, 文件传输协议
icReserved	3	为将来预留
icHTTP	4	HTTP, 超文本传输协议
icHTTPS	5	安全 HTTP

(5) 代理服务器属性。该属性用于设置或返回和互联网进行通信的代理服务器名称。




(6) 远程主机属性。该属性用于返回或设置远程计算机，控件向它发送数据或从它那里接收数据。远程主机属性既可以是主机名，也可以是 IP 地址。

(7) 远程端口属性。该属性用于返回或设置要连接的远程端口号。在设置 Protocol 属性时，将对每个协议自动地把 RemotePort 属性设置成适当的默认端口，例如 HTTP 的默认端口为 80，FTP 的默认端口为 21。

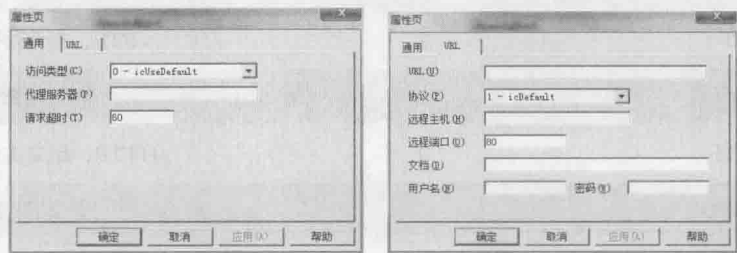


(8) URL 属性。该属性用于设置或返回 Execute 或 OpenURL 方法使用的 URL。



**提示**

用户也可以右击互联网传输控件，在弹出的快捷菜单中选择【属性】菜单项，在弹出的【属性页】对话框中，可以通过【通用】选项卡和【URL】选项卡设置访问类型、文档、UserName、Password、协议、代理服务器、远程主机、远程端口和 URL 等属性。



2. 互联网传输控件的主要方法

(1) 取消请求方法。该方法用于取消当前请求，并关闭当前创建的所有连接。语法如下。

```
object.Cancel
```

(2) 执行请求方法。该方法用于执行对远程服务器的请求，只能发送对特定协议有效的请求。语法如下。

```
object.Execute url, operation, data, requestHead- ers
```

其中，各个关键字所对应的含义如表所示。

关键字	功能
object	对象表达式，指定对象
url	用于指定控件将要连接的 URL。如果这里未指定 URL，将使用 URL 属性中指定的 URL
operation	用于指定将要进行的操作类型
data	用于指定要操作的数据
requestHeaders	用于指定由远程服务器传来的附加的标头

上述语法中的 operation 用于指定要进行的操作类型，它所支持的 HTTP 协议的有效设置值如表所示。

operation 值	含义
GET	检索由 URL 属性指定的 URL 中的数据
HEAD	发送请求的标头
POST	传递数据给服务器，该数据在 data 参数中
PUT	PUT 操作，被替代的页面名在 data 参数中

operation 参数所支持的 FTP 的有效设置值如表所示。

operation 值	含义
CD file1	改变目录。改变到 file1 中指定的目录
CDUP	改变到父目录。等效于“CD..”
CLOSE	关闭当前的 FTP 连接
DELETE file1	删除 file1 中指定的文件
DIR file1	目录。搜索 file1 中指定的目录（允许使用通配符，但要使用远程主机的语法）。如果没有指定 file1，将返回当前的整个工作目录。使用 GetChunk 方法返回目录数据
GET file1 file2	检索 file1 中指定的远程文件，并创建 file2 中指定的新本地文件
LS file1	列表。搜索 file1 中指定的目录（允许使用通配符，但要使用远程主机的语法）。使用 GetChunk 方法返回文件目录数据
MKDIR file1	创建目录。创建 file1 中指定的目录，创建是否成功取决于用户在远程主机上的权限
PUT file1 file2	复制 file1 指定的本地文件到 file2 指定的远程主机上
PWD	打印工作目录。返回当前目录名，使用 GetChunk 方法返回数据
QUIT	终止当前用户
RECV file1 file2	检索 file1 中指定的远程文件，并创建 file2 中指定的本地新文件，等效于 GET
RENAME file1 file2	将 file1 中命名的远程文件重命名为 file2 中指定的新名称，成功与否取决于用户在远程主机上的权限
RMDIR file1	删除目录。删除 file1 中指定的远程目录，成功与否取决于用户在远程主机上的权限
SEND file1 file2	复制 file1 指定的本地文件到 file2 指定的远程主机上，等效于 PUT
SIZE file1	返回 file1 指定的目录的大小

(3) GetHeader 方法。该方法用于获取 HTTP 文件头。语法如下。

```
object.GetHeader (hdrName)
```

(4) OpenURL 方法。该方法用于打开并返回指定 URL 的文档。语法如下。

```
object.OpenUrl url [,datatype]
```

### 3. 互联网传输控件的主要事件

状态改变事件。该事件在连接中状态发生改变时被触发。语法如下。

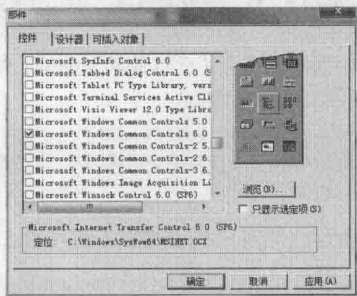
object\_StateChanged(ByVal State As Integer)

18.2.2 实现互联网文件上传

【范例 18-2】不需要登录 FTP 服务器，用 VB 编写程序，实现将文件上传至互联网。

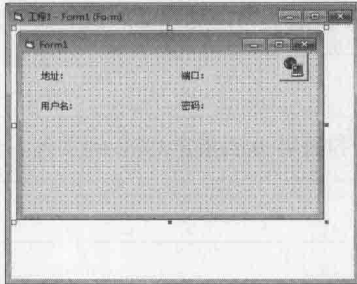
第 1 步：界面设计

(1) 新建一个工程，在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令，在弹出的【部件】对话框中选择【控件】选项卡，选择【Microsoft Internet Transfer Control 6.0】和【Microsoft Windows Common Controls 6.0】两个复选框，然后单击【确定】按钮。



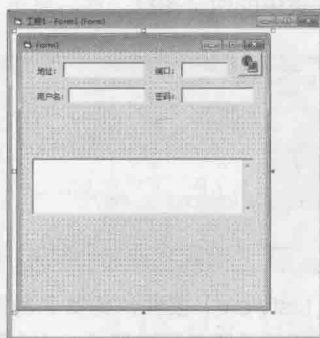
(2) 在 Form1 窗体中添加 1 个互联网传输 (Inet) 控件和 4 个标签 (Label) 控件，标签控件按照下表所示设置它们的属性。

控件名称	Caption	控件功能
Label1	地址：	提示用户输入 FTP 地址
Label2	端口：	提示用户输入 FTP 端口
Label3	用户名：	提示用户输入用户名
Label4	密码：	提示用户输入密码



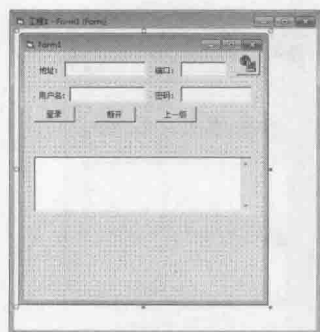
(3) 在 Form1 窗体中添加 5 个文本框 (TextBox) 控件，按照下表所示设置它们的属性。

名称	Text	Multi Line	Scroll Bars	控件功能
TxtUrl	“”	False	0-vbSBNone	输入地址
TxtPort	“”	False	0-vbSBNone	输入端口号
TxtUser	“”	False	0-vbSBNone	输入用户名
TxtPassword	“”	False	0-vbSBNone	输入密码
TxtStatus	“”	True	2-Vertical	显示登录信息

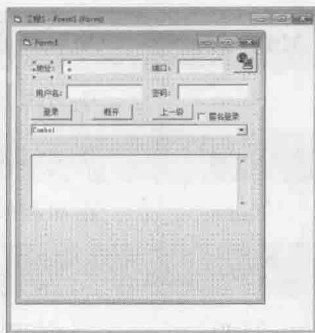


(4) 在 Form1 窗体中添加 1 个包含 3 个命令按钮 (CommandButton) 控件的控件组, 按照下表所示设置它们的属性。

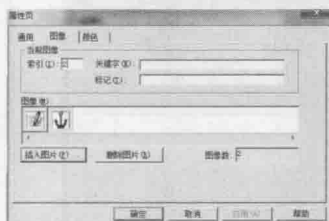
控件名称	Caption	控件功能
Cmd(0)	登录	用于用户登录
Cmd(1)	断开	用于用户账号注销
Cmd(2)	上一级	用于查看上一级目录



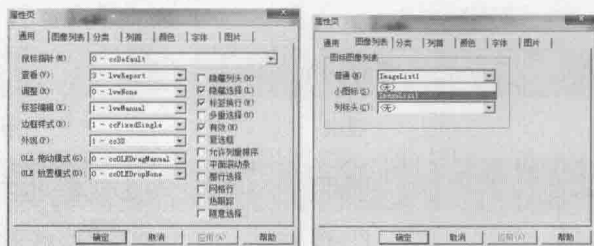
(5) 在 Form1 窗体上添加 1 个下拉列表框 (ComboBox) 控件和 1 个复选框 (CheckBox) 控件, 设置 CheckBox 控件的 Caption 属性为 “匿名登录”。



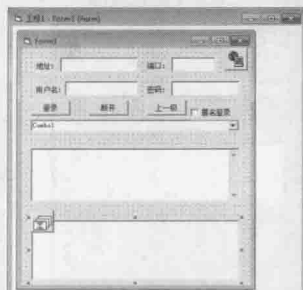
(6) 在窗体上添加 1 个图像列表 (ImageList) 控件, 在其【属性页】对话框中的【图像】选项卡中添加强书光盘中“Samplech14”文件夹中的两个图标, 然后单击【确定】按钮。



(7) 在窗体上添加 1 个列表视图 (ListView) 控件, 将其名称属性设置为“List1”, 在其【属性页】对话框中的【通用】选项卡中按照下图所示设置其所有属性; 在【图像列表】选项卡中的【普通】和【小图标】下拉列表中选择“ImageList1”, 然后单击【确定】按钮。



(8) 将上述所有控件的排列成下图所示的效果。



提示

名称为 TxtPassword 的文本框, 其 Password 属性需设置为 “\*”。



## 第 2 步：编写代码

(1) 在 Form1 窗体中右击鼠标，在弹出的快捷菜单中选择【查看代码】菜单项，在打开的代码窗口中声明公有变量。

```
01 Option Explicit
02 Dim InetData As Variant      '定义变型变量，用于获取缓冲区中的文件内容
03 Dim CurrentServerDir As String '定义字符变型，用于获取当前服务驱动器路径
04 Dim xpos As Long, ypos As Long '定义两个长整型变量，用于定位鼠标位置
05 Dim OperationStyle As Integer '定义整型变量，用于设置操作类型
```

(2) 在 Form1 窗体中双击鼠标，在打开的代码窗口中输入以下代码（代码 18-2-1.txt）。

```
01 Private Sub Form_Load() '初始化控件属性
02     Check1.Value = 1      '初始化为匿名访问
03     TxtUrl.Text = "192.168.0.1" '设置初始化地址
04     TxtPort.Text = "21"    '设置初始化端口
05     TxtUser.Text = ""      '设置初始化用户名为空
06     TxtPassWord.Text = "" '设置初始化密码为空
07     TxtStatus.Text = ""   '设置连接状态列表为空
08     TxtUser.Enabled = False '记名文本框不可用
09     TxtPassWord.Enabled = False '密码文本框不可用
10     TxtPassWord.PasswordChar = "*" '设置密码显示形式
11     Combo1.Text = ""      '清空驱动器路径
12     Cmd(0).Enabled = True '【登录】按钮可用
13     Cmd(1).Enabled = False '【断开】按钮不可用
14     Cmd(2).Enabled = False '【上一级】按钮不可用
15     List1.ColumnHeaders.Add 1, "名称", "名称", List1.Width '为列表框设置标题
16 End Sub
```

(3) 在 Form1 窗体中双击名称为【Cmd】的命令按钮，在打开的代码窗口中输入以下代码（代码 18-2-2.txt）。

```
01 Private Sub Cmd_Click(Index As Integer) '登录按钮用于连接服务器
02     If Index = 0 Then '如果单击了登录按钮
03         With Inet1 '使用互联网传输控件
04             If Left(Trim(TxtUrl.Text), 6) <> "ftp://" Then '判断“地址”文本框中是否有“ftp:”字样
05                 .URL = "ftp://" & Trim(TxtUrl.Text) '没有出现“ftp:”字样，自动添加
06             End If
07             If TxtPort.Text <> "" Then
08                 .RemotePort = CInt(Trim(TxtPort.Text)) '端口号不为空，添加远程端口号
09             Else
10                 .RemotePort = 21 '端口号为空，为远程端口号添加一个默认端口号
11             End If
12             If Check1.Value = 1 Then
13                 .UserName = "" '如果是匿名访问，用户名为空
```

```

14     Else
15         .UserName = Trim(TxtUser.Text) '非匿名访问, 添加用户名
16         .Password = Trim(TxtPassWord.Text) '非匿名访问, 添加密码
17     End If
18 End With
19 CurrentServerDir = "/" '设置当前服务驱动器
20 If Inet1.StillExecuting Then '判断互联网传输控件是否正忙
21     MsgBox "无法断开保持连接" '如果正忙, 弹出提示信息
22     Exit Sub
23 End If
24 ListServer '列出服务器根目录
25 Cmd(1).Enabled = True '【断开】按钮可用
26 Cmd(2).Enabled = True '【上一级】按钮可用
27 End If
28 If Index = 1 Then '单击【断开】按钮
29     Inet1.Cancel '断开连接
30     List1.ListItems.Clear '清空列表
31     Cmd(2).Enabled = False '【上一级】按钮不可用
32 End If
33 If Index = 2 Then '单击【上一级】按钮
34     If CurrentServerDir <> "" Then '判断当前服务驱动器是否为空
35         If Inet1.StillExecuting Then '判断互联网传输控件是否正忙
36             MsgBox "还没有执行完毕!" '如果忙, 弹出提示信息
37         Else
38             OperationStyle = 2 '当前驱动器控件不忙, 设置操作类型
39             Inet1.Execute, "CD ../" '改变驱动器路径
40             UpServerDir '返回上一层驱动器路径
41         End If
42     Else
43         MsgBox "已经到了最上一层目录!"
44     End If
45 End If
46 End Sub

```

(4) 在 Form1 窗体中双击【匿名登录】复选框, 在打开的代码窗口中输入以下代码。

```

01 Private Sub Check1_Click() '复选按钮选中时代码
02     If Check1.Value = 0 Then '判断是否允许匿名访问
03         TxtUser.Enabled = True '允许匿名访问, 无须输入用户名
04         TxtPassWord.Enabled = True '允许匿名访问, 无需输入密码
05     End If
06 End Sub

```

(5) 在 Form1 窗体中双击【互联网传输】控件, 在打开的代码窗口中输入以下代码 (代码 18-2-3.txt)。

```

01 Private Sub Inet1_StateChanged(ByVal State As Integer) '连接状态发生改变时所引发的事
件

```

```

02 Dim tempArray As Variant
03 Dim i As Integer
04 Dim FileSize As Variant
05 Dim itmX As ListItem
06 On Error Resume Next ' 程序继续运行
07 Select Case State ' 判断当前连接状态
08     Case 0
09         ' 不进行任何操作
10     Case 1
11         TxtStatus.Text = TxtStatus.Text & vbCrLf & "正在查询所指定的主机的 IP 地址"
12     Case 2
13         TxtStatus.Text = TxtStatus.Text & vbCrLf & "已成功地找到所指定的主机的 IP 地址"
14     Case 3
15         TxtStatus.Text = TxtStatus.Text & vbCrLf & "正在与主机连接"
16     Case 4
17         TxtStatus.Text = TxtStatus.Text & vbCrLf & "已与主机连接成功"
18     Case 5
19         TxtStatus.Text = TxtStatus.Text & vbCrLf & "正在向主机发送请求"
20     Case 6
21         TxtStatus.Text = TxtStatus.Text & vbCrLf & "发送请求已成功"
22     Case 7
23         TxtStatus.Text = TxtStatus.Text & vbCrLf & "正在接收主机的响应"
24     Case 9
25         TxtStatus.Text = TxtStatus.Text & vbCrLf & "正在解除与主机的连接"
26     Case 10
27         TxtStatus.Text = TxtStatus.Text & vbCrLf & "已成功地与主机解除了连接"
28     Case 11
29         TxtStatus.Text = TxtStatus.Text & vbCrLf & "与主机通信时出现了错误"
30         TxtStatus.Text = TxtStatus.Text & vbCrLf & "错误" & Inet1.ResponseCode & "." & Inet1.
            ResponseInfo
31     Case 8, 12
32         Select Case OperationStyle ' 判断并选择操作类型
33             Case 1
34                 TxtStatus.Text = TxtStatus.Text & vbCrLf & "成功列出目录内容"
35                 List1.ListItems.Clear ' 清空列表
36                 InetData = Inet1.GetChunk(1024, 0) ' 从缓冲区中依次提取 1024 个字节的回应文本
37                 Combo1.Text = CurrentServerDir ' 设置当前驱动器路径
38                 If Trim(InetData) <> 0 Then ' 判断是否从缓冲区中读取数据
39                     tempArray = Split(InetData, vbCrLf, , vbTextCompare)
40                     ' 对读取的数据返回一个下标从零开始的一维数组
41                     i = 0
42                     Do While i < UBound(tempArray) ' 循环至数组的最大下标
43                         If tempArray(i) <> "" Then ' 数组不为空
44                             DealList (tempArray(i)) ' 显示所选择目录下需上传的文件
45                         End If
46                     i = i + 1

```

```

46         Loop
47     End If
48     Case 2
49         TxtStatus.Text = TxtStatus.Text & vbCrLf & " 成功改变目录 " & " 改变目录状态
50         ListServer
51     Case Else
52     End Select
53 End Select
54 TxtStatus.SelLength = Len(TxtStatus.Text)
55 End Sub

```

(6) 在代码窗口的【对象】下拉列表中选择“List”控件，在【过程】下拉列表中选择“MouseDown”事件，并输入以下代码。

```

01 Private Sub List1_MouseDown(Button As Integer, Shift As Integer, x As Single, y As Single)
02     xpos = x      ' 指定鼠标在当前窗体出现的横坐标
03     ypos = y      ' 指定鼠标在当前窗体出现的纵坐标
04 End Sub

```

(7) 在代码窗口的【对象】下拉列表中选择“List”控件，在【过程】下拉列表中选择“DbClick”事件，并输入以下代码（代码 18-2-4.txt）。

```

01 Private Sub List1_DbClick()
02     Dim item As ListItem
03     If List1.HitTest(xpos, ypos) Is Nothing Then      ' 用 hittest 方法测试点击位置
04         Exit Sub      ' 忽略
05     Else
06         Set item = List1.HitTest(xpos, ypos)      ' 设置新的点击位置
07     End If
08     Select Case item.Icon      ' 选择文件图标
09         Case 1      ' 是文件
10             ' 不进行任何操作
11         Case 2      ' 是目录
12             OperationStyle = 2      ' 设置操作形式
13             If item.Text = "/" Then ' 如果为根目录不再进行任何操作
14             Else
15                 Elself item.Text = "../" Then      ' 如果为上级目录
16                     Inet1.Execute "CD ../"      ' 改变驱动器路径
17                     UpServerDir      ' 返回上一层磁盘路径目录
18                 Else
19                     CurrentServerDir = CurrentServerDir & item      ' 设置当前服务器路径
20                     Inet1.Execute "CD " & CurrentServerDir      ' 改变网络传输控件的路径
21                 End If
22             End Select
23 End Sub

```

(8) 在代码窗口中编写 UpServerDir 子过程、AddDirToList 子过程、DealList 子过程、AddFileToList


子过程和 ListServer 子过程 (代码 18-2-5.txt)。

```

01 Private Sub UpServerDir()
02 Dim tempPos1 As Integer
03 On Error Resume Next
04 If CurrentServerDir <> "/" Then '判断当前服务器路径是否为根路径
05     tempPos1 = InStrRev(CurrentServerDir, "/", Len(CurrentServerDir) - 1, vbTextCompare)
'定位文件路径
06     CurrentServerDir = Mid(CurrentServerDir, 1, tempPos1) '重新设置当前服务器路径
07 End If
08 End Sub
09 Private Sub DealList(tempStr As String)
10 If Right(Trim(tempStr), 1) <> "/" Then
11     AddFileToList (tempStr) '表示接收到的是文件
12 Else
13     AddDirToList (tempStr) '表示接收到的是目录
14 End If
15 End Sub
16 Private Sub AddFileToList(tempStr As String)
17 Dim itmX As ListItem
18 Set itmX = List1.ListItems.Add(, tempStr) '将文件名加入到列表视图中
19 itmX.Icon = 1
20 itmX.SmallIcon = 1
21 End Sub
22 Private Sub AddDirToList(tempStr As String)
23 Dim itmX As ListItem
24 Set itmX = List1.ListItems.Add(, tempStr) '将目录加入到列表视图中
25 itmX.Icon = 2
26 itmX.SmallIcon = 2
27 End Sub
28 Private Sub ListServer()
29 On Error GoTo ErrorH
30 If Not Inet1.StillExecuting Then '判断互联网传输控件是否正忙
31     OperationStyle = 1 '如果忙, 设置操作类型
32     Inet1.Execute, "DIR" '列出服务器指定目录下的文件和子目录
33 End If
34 Exit Sub '退出子过程
35 End Sub

```

## 【运行结果】

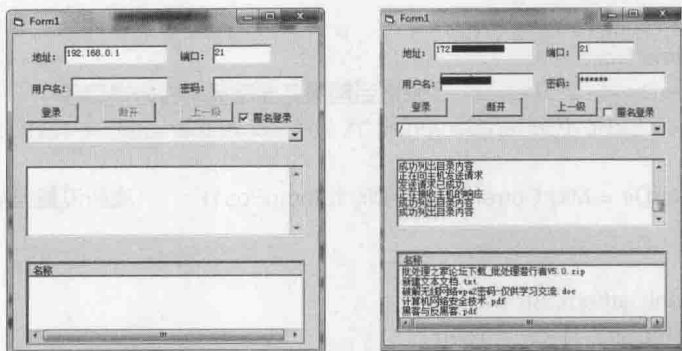
(1) 保存程序, 单击【启动】按钮 , 运行程序。



**提示**

地址“192.168.0.1”为程序初始化地址, 而非 FTP 服务器地址。若要输入用户名和密码, 应撤选【匿名登录】复选框。

(2) 输入 FTP 服务器的地址、端口、用户名和密码, 单击【登录】按钮, 若连接成功, 就会在列表框中列出服务器上的文件目录。



**技巧**

如果要传送比较大的文件时, 应该将此文件根据实际情况分成若干个数据包。在接收端建立数据缓冲区, 依次接收从客户端传送过来的数据包, 并将数据缓冲区的数据写入相应的文件中, 这样就实现了大文件传输。

## 18.3 网页浏览器应用编程



本节视频教学录像: 7 分钟

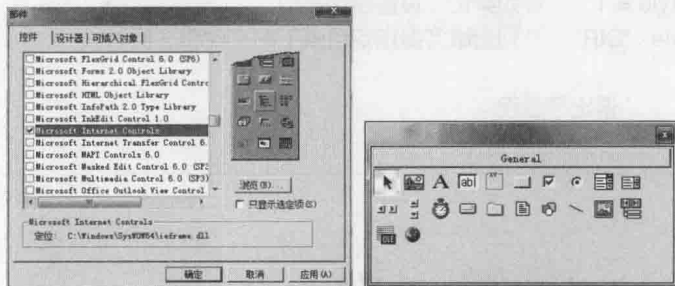
网页浏览器 (Web Browser) 控件是一个基于 IE 内核, 并封装了 IE 诸多功能的浏览器。它不仅支持通过输入 URL 地址进行浏览, 也支持通过单击超链接进行网页浏览, 同时还可以返回到历史浏览记录。

### 18.3.1 网页浏览器控件的属性、事件和方法

在介绍网页浏览器控件之前, 首先需要把网页浏览器控件添加到工具箱中。具体的操作步骤如下。

(1) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令, 在弹出的【部件】对话框中选择【控件】选项卡, 选择【Microsoft Internet Controls】复选框, 然后单击【确定】按钮。

(2) 这样就将网页浏览器 (Web Browser) 控件添加到了工具箱中。



#### 1. 网页浏览器控件的常用属性

##### (1) LocationURL 属性

该属性用于返回或设置控件显示网页页面的 URL 地址。程序运行时此属性为只读状态, 不可修改。

语法如下。

---

```
object.LocationURL = string
```

---

#### (2) LocationName 属性

该属性用于返回或设置控件显示 Web 页面的标题。程序运行时此属性为只读状态，不可修改。语法如下。

---

```
object.LocationName
```

---

### 2. 网页浏览器控件的常用方法

(1) 返回主页方法。该方法用于返回到预设的主页链接页面。语法如下。

---

```
object.GoHome
```

---

(2) 后退方法。该方法用于跳到执行历史列表中的超链接后跳。语法如下。

---

```
object.GoBack
```

---

(3) 前进方法。该方法用于跳到执行历史列表中的超链接前跳。语法如下。

---

```
object.GoForward
```

---

(4) 停止方法。该方法用于停止当前正在下载的网页。语法如下。

---

```
object.Stop
```

---

(5) Navigate2 方法。该方法用于加载所请求的网页。语法如下。

---

```
Object.Navigate2URL[Flags,] [TargetFrameName,] [PostData,] [Headers]
```

---

### 3. 网页浏览器控件的常用事件

(1) 下载开始事件。该事件在下载操作开始时被触发。语法如下。

---

```
Private Sub object _DownloadBegin()
```

---

(2) 下载结束事件。该事件在下载操作结束或者失败时被触发。语法如下。

---

```
Private Sub object _DownloadComplete()
```

---

(3) 标题改变事件。该事件在当前文档标题改变时被触发。语法如下。

---

```
Private Sub WebBrowser1_TitleChange(ByVal Text As String)
```

---

(4) 过程改变事件。该事件用于跟踪下载操作过程并定期触发。语法如下。

---

```
Private Sub WebBrowser1_ProgressChange (ByVal Progress As Long, ByVal ProgressMax As Long)
```

---



### 18.3.2 实现自定义网页浏览器应用

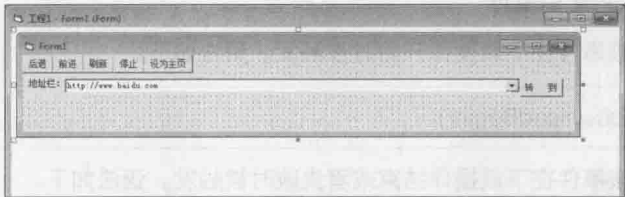
**【范例 18-3】**使用网页浏览器控件制作一个简单的浏览器，能够浏览网页、设置主页、刷新网页等。

第 1 步：界面设计

- (1) 新建一个工程，在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令，在弹出的【部件】对话框中选择【控件】选项卡，选择【Microsoft Internet Controls】复选框，然后单击【确定】按钮。
- (2) 在 Form1 窗体中添加 1 个网页浏览器（WebBrowser）控件，根据需要设置网页浏览器的控件大小。
- (3) 在 Form1 窗体中添加一个标签（Label）控件，设置【Caption】属性为“地址栏：”。
- (4) 在 Form1 窗体中添加一个组合框（ComboBox）控件，设置【Text】属性为“http://www.baidu.com”。
- (5) 在 Form1 窗体中添加 6 个命令按钮（CommandButton）控件，按照下表所示设置它们的属性。

控件名称	Caption	控件功能
Command1	转到	用于转到地址栏所指定的页面
Command2	后退	用于返回或设置上一页浏览过的网页页面
Command3	前进	用于返回或设置下一页浏览过的网页页面
Command4	刷新	用于刷新正在浏览的网页
Command5	停止	用于停止正在打开的网页
Command6	设为主页	用于显示或设置网站的主页

- (6) 将上述所有的控件排列成下图所示的效果。



第 2 步：编写代码

- (1) 在 Form1 窗体的空白处双击鼠标，在打开的代码窗口中声明公有变量。

```
01 Private Sub Form_Load()  
02   WebBrowser1.Navigate Combo1.Text ' 打开默认的浏览器主页  
03 End Sub
```

- (2) 在 Form1 代码窗口的【对象】下拉列表中选择“Combo1”控件，在【过程】下拉列表中选择“KeyPress”事件，然后输入以下代码。

---

```

01 Private Sub Combo1_KeyPress(KeyAscii As Integer)
02   If KeyAscii = 13 Then    ' 如果按下回车键
03     WebBrowser1.Navigate Combo1.Text    ' 转到当前地址栏中网页的地址
04   End If
05 End Sub

```

---

(3) 在 Form1 窗体中双击名称为【转到】的命令按钮，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command1_Click()
02   WebBrowser1.Navigate Combo1.Text    ' 转到当前地址栏中网页的地址
03 End Sub

```

---

(4) 在 Form1 窗体中双击名称为【后退】的命令按钮，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command2_Click()
02   WebBrowser1.GoBack    ' 返回上一页浏览过的网页页面
03 End Sub

```

---

(5) 在 Form1 窗体中双击名称为【前进】的命令按钮，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command3_Click()
02   WebBrowser1.GoForward    ' 返回下一页浏览过的网页页面
03 End Sub

```

---

(6) 在 Form1 窗体中双击名称为【刷新】的命令按钮，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command4_Click()
02   WebBrowser1.Refresh    ' 刷新正在浏览的网页
03 End Sub

```

---

(7) 在 Form1 窗体中双击名称为【停止】的命令按钮，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command5_Click()
02   WebBrowser1.Stop    ' 停止正在打开的网页
03 End Sub

```

---

(8) 在 Form1 窗体中双击名称为【设为主页】的命令按钮，在打开的代码窗口中输入以下代码。

---


```

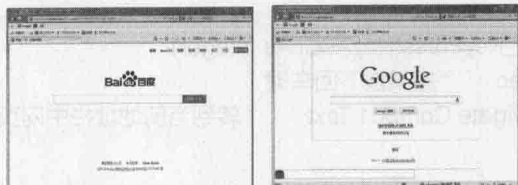
01 Private Sub Command6_Click()
02   WebBrowser1.GoHome    ' 设置当前页面为主页
03   Combo1.Text = WebBrowser1.LocationURL ' 将网页地址赋值给组合框控件
04 End Sub

```

---

## 【运行结果】

- (1) 保存程序，单击【启动】按钮 ，运行程序。
- (2) 在地址栏中输入“www.google.cn”，按回车键或单击【转到】按钮，即可打开谷歌搜索引擎。



(3) 单击【后退】按钮,可返回上一页浏览过的网页页面;单击【前进】按钮,可返回下一页浏览过的网页页面;单击【停止】按钮,可停止正在显示的网页;单击【刷新】按钮,可刷新正在浏览的网页;单击【设为主页】按钮,可设置当前浏览的网页为主页。

## 18.4 高手点拨



本节视频教学录像: 1 分钟

超文本传输协议和文件传输协议的区别就在于 FTP 是 Internet 文件传送的基础。通过该协议,用户可以从一个 Internet 主机向另一个 Internet 主机拷贝文件。超文本传输协议是用于从万维网服务器上传输超文本到本地浏览器的传输协议。它保证计算机正确快速地传输超文本文档,还能确定传输文档中的哪部分内容首先显示。

## 18.5 实战练习

### 一、思考题

试简述超文本传输协议和文件传输协议的区别。

### 二、上机题

编写一个关于 FTP 互联网传输的程序,要求:

1. 可以进行 FTP 服务器上传。
2. 可以进行 FTP 服务器下载。

# 第19章



本章视频教学录像：29 分钟

## Visual Basic 中的视听 ——图形图像与多媒体编程


图形图像与多媒体丰富了应用程序的表现形式，能给用户更好的使用体验。适当地使用图形图像与多媒体技术，可以给应用程序增色不少。

本章讲解 Visual Basic 6.0 中坐标系和颜色的设置方法、常用的绘图方法以及多媒体应用程序的设计方法。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握坐标系概念
- ☐ 了解颜色设置方法
- ☐ 熟悉绘图方法
- ☐ 了解多媒体控制接口控件基本概念
- ☐ 了解多媒体控制接口控件的属性
- ☐ 熟悉多媒体控制接口控件的事件
- ☐ 熟悉动画控件的属性
- ☐ 了解动画控件的方法


# 19.1 图形应用编程

 本节视频教学录像: 13 分钟

总体来讲, 图形要比文字直观、有趣许多, 利用图形可以让程序变得更加美观、个性化, 也可使程序与生活更加贴近。本章介绍如何在 Visual Basic 中进行图形应用编程。

## 19.1.1 坐标系

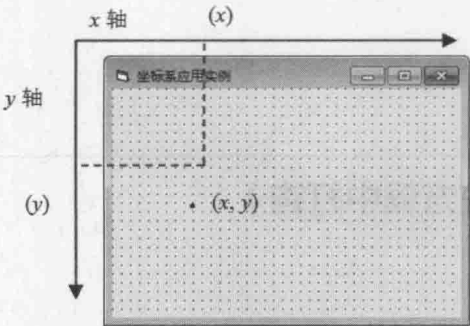
坐标系是设计图形必不可少的工具, 窗体、控件等各种元素的大小和位置等信息都是以坐标系为参照的。构成坐标系的 3 个要素是: 坐标原点、坐标度量单位、坐标轴的长度与方向。



Visual Basic 中除了提供有系统标准坐标系外, 还允许用户自定义坐标系。

**提示**

在 Visual Basic 中, 平面图形用二维坐标系来表示。一般来说, 水平方向作为二维坐标系的 x 轴, 最左端的 x 值为 0; 竖直方向作为二维坐标系的 y 轴, 最上端的 y 值为 0; 这样平面上某个点的坐标就可以用 (x, y) 来表示, 而坐标原点的坐标就是 (0, 0), 位于当前对象的左上角。



### 1. 坐标系的常用属性

(1) 度量单位属性 (ScaleMode): 该属性用于设置对象所在坐标系的度量单位, 用户可以在属性窗口中的【ScaleMode】下拉列表中设置该属性。



该属性中的可选值及其所代表的度量单位如表所示。

可选项	所代表的度量单位
0 - User	用户自定义类型，即通过 ScaleHeight、ScaleWidth、ScaleLeft 和 ScaleTop 等属性中的一个或多个设置为自定义
1 - Twip	缇，1 英寸为 1 440 缇，1cm 为 567 缇
2 - Point	磅，每英寸为 72 磅
3 - Pixel	像素，即监视器或打印机分辨率的最小单位
4 - Character	字符，水平每个单位 =120 缇；垂直每个单位 =240 缇
5 - Inch	英寸
6 - Millimeter	mm
7 - Centimeter	cm

用户也可以在代码中设置该属性，语法如下。

```
object.ScaleMode [= value]
```

(2) ScaleHeight 和 ScaleWidth 属性：这两个属性用于返回或设置对象内部的水平 (ScaleWidth) 或垂直 (ScaleHeight) 度量单位。例如，当我们在 Visual Basic 中通过拖动鼠标改变一个窗体的大小后，该窗体的 ScaleHeight 和 ScaleWidth 属性就会做相应的改变。



用户也可以在代码中设置该属性，语法如下。

```
object.ScaleHeight [= value]
```

```
object.ScaleWidth [= value]
```

如果省略“object”，指的则是当前窗体。



注意

ScaleWidth 和 ScaleHeight 不包括边框厚度或菜单以及标题的高度。因此，ScaleWidth 和 ScaleHeight 总是指对象内的可用空间的大小。内部尺寸和外部尺寸（尺寸由 Width 和 Height 指定）的区别，对于有宽厚边框的窗体特别重要。这些单位也可不同：Width 和 Height 总是按照容器的坐标系来表示，ScaleWidth 和 ScaleHeight 决定了对象本身的坐标系。

(3) ScaleLeft 和 ScaleTop 属性：这两个属性用于返回或设置一个对象左边 (ScaleLeft) 和上边 (ScaleTop) 的水平和垂直的坐标。

语法如下。

object.ScaleLeft [= value]  
object.ScaleTop [= value]

(4) Left 和 Top 属性: Left 属性用于返回或设置对象内部的左边与容器左边之间的距离。语法如下。

object.Left [= value]

Top 属性用于返回或设置对象的内部顶部和容器顶边之间的距离。语法如下。

object.Top [= value]

(5) Height 和 Width 属性: Height 属性用于返回或设置对象的高度。语法如下。

object.Height [= number]

Width 属性用于返回或设置对象的宽度。语法如下。

object.Width [= number]

(6) CurrentX 和 CurrentY 属性: 这两个属性用于返回或设置下一次打印或绘图方法的水平 (CurrentX) 或垂直 (CurrentY) 坐标。语法如下。

object.CurrentX [= x]  
object.CurrentY [= y]

2. 坐标系的常用方法

(1) ScaleX、ScaleY 方法: 这两个方法用于将窗体、图像框等控件的宽度或高度值从一种度量单位转换为另一种度量单位。语法如下。

object.ScaleX (width, fromscale, toscale)  
object.ScaleY (height, fromscale, toscale)

语句中各个关键字的含义如表所示。

关键字	含义
object	对象表达式, 若省略 object, 则带有焦点的 Form 对象默认为 object
width	为 object 指定被转换的度量单位的宽度值
height	为 object 指定被转换的度量单位的高度值
fromscale	为一个常数或数值, 用于指定 object 的 width 或 height 从哪一种坐标系转换
toscale	为一个常数或数值, 用于指定 object 的 width 或 height 转换到哪一种坐标系

fromscale 和 toscale 的值及其所代表的度量单位如表所示。



可选项	所代表度量单位
0 - User	用户定义：指示 object 的宽度和高度设置为自定义值
1 - Twip	缇，1 英寸为 1440 缇，1cm 为 567 缇
2 - Point	磅，每英寸为 72 磅
3 - Pixel	像素，即监视器或打印机分辨率的最小单位
4 - Character	字符，水平每个单位 =120 缇，垂直每个单位 =240 缇
5 - Inch	英寸
6 - Millimeter	mm
7 - Centimeter	cm

(2) Scale 方法：该方法用于定义窗体、图像框或其他控件的坐标系。语法如下。

object.Scale (x1, y1) - (x2, y2)


语句中各个关键字的含义如表所示。

关键字	含义
Object	对象表达式，如果省略 object，则带有焦点的 Form 对象默认为 object
x1, y1	定义 object 左上角的水平（x 轴）和垂直（y 轴）坐标
x2, y2	定义 object 右下角的水平（x 轴）和垂直（y 轴）坐标

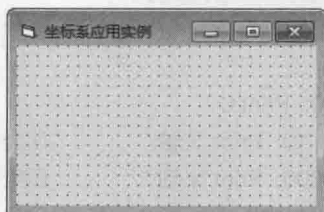
### 3. 坐标系使用实例

下面通过一个实例来学习坐标系的使用方法。

#### 【范例 19-1】通过一个坐标系应用实例来了解默认坐标系与自定义坐标系的区别。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标，然后单击【打开】按钮。

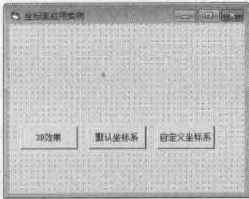
(2) 将 Form1 的 Caption 属性设置为“坐标系应用实例”。



(3) 在窗体中创建 3 个命令按钮（Command Button）控件，控件属性设置如表所示。

名称	Caption	控件作用
Cmd3D	3D 效果	设置文字的立体效果
CmdDef	默认坐标系	在窗体中显示默认坐标系示意图
CmdUser	自定义坐标系	在窗体中显示新建坐标系示意图

(4) 排列 3 个按钮控件的位置，如图所示。



(5) 双击【3D 效果】按钮，在弹出的代码窗口中添加以下代码（代码 19-1-1.txt）。

```
01 Private Sub Cmd3D_Click() ' 设置文字立体效果代码
02 Cls ' 清除屏幕内容
03 FontSize = 30 ' 设置字体大小
04 ForeColor = QBColor(0) ' 设置背景字体颜色
05 CurrentX = 160 ' 设置当前坐标 x 和 y 值
06 CurrentY = 160
07 Print "3D 效果 " ' 以背景色输出字符
08 ForeColor = QBColor(15) ' 设置前景字体颜色
09 CurrentX = 190 ' 设置前景色的坐标 x 和 y 值
10 CurrentY = 180
11 Print "3D 效果 " ' 以前景色输出字符
12 End Sub
```

(6) 双击【默认坐标系】按钮，在弹出的代码窗口中添加以下代码（代码 19-1-2.txt）。

```
01 Private Sub CmdDef_Click() ' 默认坐标系代码
02 Cls ' 清除屏幕内容
03 Form1.Scale ' 设置坐标系
04 FontSize = 10 ' 设置字体大小
05 ForeColor = QBColor(0) ' 设置前景字体颜色
06 Line (10, 10)-(4300, 10) ' 画线
07 Line (10, 10)-(10, 2800)
08 CurrentX = 10: CurrentY = 10: Print 0 ' 在坐标 (10,10) 处输出字符 "0"
09 CurrentX = 4300: CurrentY = 10: Print "x 轴" ' 在坐标 (4300,10) 处输出字符 "x 轴"
10 CurrentX = 10: CurrentY = 2800: Print "y 轴" ' 在坐标 (10,2800) 处输出字符 "y 轴"
11 End Sub
```

(7) 双击【自定义坐标系】按钮，在弹出的代码窗口中添加以下代码（代码 19-1-3.txt）。

```

01 Private Sub CmdUser_Click()      ' 自定义坐标系代码
02 Cls      ' 清除屏幕内容
03 FontSize = 10      ' 设置字体大小
04 ForeColor = QBColor(0)      ' 设置前景字体颜色
05 Form1.ScaleLeft = -500      ' 设置左上角坐标为 (-500,500)
06 Form1.ScaleTop = 500
07 Form1.ScaleWidth = 1000      ' 定义窗体当前宽度的 1/1000 为水平单位, 高度的 1/800 为垂直单位
08 Form1.ScaleHeight = -800
09 Line (-480, 10)-(-480, 10) ' 画线
10 Line (0, 480)-(0, -250)
11 CurrentX = 0: CurrentY = 0: Print 0      ' 输出坐标原点及 "x 轴" 和 "y 轴"
12 CurrentX = 430: CurrentY = 10: Print "x 轴"
13 CurrentX = 10: CurrentY = 480: Print "y 轴"
14 End Sub

```



#### 提示

ForeColor 属性用于设置对象的前景色, QBColor 函数用于为对象设置颜色, Line 方法用在对象上绘制直线或矩形。这些内容将在后面介绍。

## 【代码详解】

第(5)步代码两次以不同的坐标、不同的颜色调用 Print 方法在窗体上打印“3D 效果”文字, 实现文字的 3D 显示。

第(6)步中的代码 Form1.Scale 设置窗体的坐标系为默认, 然后沿  $x$  轴下方和  $y$  轴右方画两条直线, 并绘制 0、 $x$  轴和  $y$  轴。

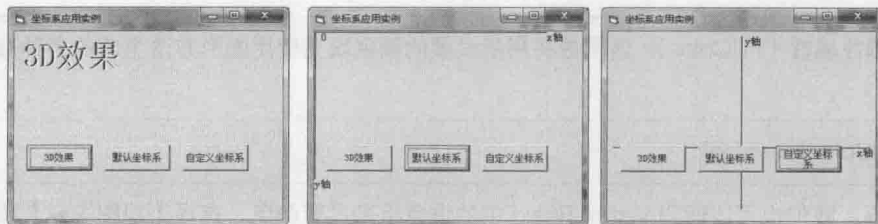
第(7)步中的代码 Form1.ScaleLeft = -500, Form1.ScaleTop = 500, 设置窗体左上角的坐标为 (-500,500), 所以坐标原点位于窗体中央; Form1.ScaleWidth = 1000, Form1.ScaleHeight = -800, 定义窗体当前宽度的 1/1 000 为水平单位, 高度的 1/800 为垂直单位, 然后绘制出 0、 $x$  轴和  $y$  轴。

## 【运行结果】

按快捷键【F5】运行程序。单击【3D 效果】按钮, 将会以坐标原点为起始位置显示文字的 3D 效果。

单击【默认坐标系】按钮, 将会显示系统默认的坐标系位置。

单击【自定义坐标系】按钮, 将会显示窗体自定义的坐标系位置。



【范例分析】

通过本例，我们学习了坐标系的表示方法，自定义坐标系以及控制当前坐标的方法。

19.1.2 颜色设置

我们生活在一个彩色的世界中，美丽的色彩不仅能够使我们在视觉上得到享受，而且合理地使用色彩能够更清晰地传达信息。例如，将一篇文章中的重要部分以红色显示等。

Visual Basic 中提供有多种颜色操作方法，用户既可以通过调色板和颜色通用对话框设置颜色，也可以通过颜色函数如 RGB 函数或者 QBColor 函数设置颜色，还可以通过控件中与颜色相关的属性设置对象颜色。

1. 通过调色板和颜色通用对话框设置颜色

(1) 背景色属性 (BackColor)：该属性用于设置对象的背景色。用户可以在属性窗口中的【BackColor】下拉列表中设置该属性。



(2) 前景色属性 (ForeColor)：该属性用于设置对象的前景色。用户可以在属性窗口中的【ForeColor】下拉列表中设置该属性。

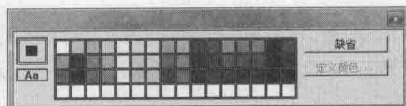
(3) 边框颜色属性 (BorderColor)：该属性用于设置对象的边框颜色。  
语法如下。



```
object.BorderColor [= color]
```

(4) 填充颜色属性 (FillColor)：该属性将用所设置的颜色填充使用图形方法生成的各种形状。  
语法如下。

```
object.FillColor [= value]
```

(5) 调色板：我们也可以使用 Visual Basic 中的调色板来设置颜色。选择【视图】>【调色板】菜单命令，打开调色板。



调色板上的  按钮代表当前控件的前景颜色和背景颜色，调色板上的  按钮代表当前控件中文本的前景颜色和背景颜色。

## 2. 使用 RGB 函数设置颜色

RGB 颜色称为加成色，R 是红色 (Red)、G 是绿色 (Green)、B 是蓝色 (Blue)。通常又将这 3 种颜色叫作三原色，计算机显示器显示的各种颜色都是由三原色经过不同程度的叠加而成的。

RGB 函数的语法如下。

```
RGB(red, green, blue)
```

语句中的 red、green 和 blue 都是范围在 0 到 255 之间的整型数。



### 提示

三原色由三种基本原色构成。原色是指不能透过其他颜色的混合调配而得出的“基本色”。以不同比例将原色混合，可以产生出其他的新颜色。以数学的向量空间来解释色彩系统，则原色在空间内可作为一组基底向量，并且能组合出一个“色彩空间”。由于人类肉眼有三种不同颜色的感光体，因此所见的色彩空间通常可以由三种基本色所表达，这三种颜色称为“三原色”。

## 3. 使用 QBColor 函数设置颜色

使用 QBColor 函数也可以为对象设置颜色，但是其设置的颜色种类很少。该函数返回一个长整型数，用来表示所对应颜色值的 RGB 颜色码。语法如下。

```
QBColor(color)
```

其中，color 参数是范围在 0 到 15 之间的整型数，每个值所代表的颜色如表所示。

color 值	颜色	color 值	颜色
0	黑色	8	灰色
1	蓝色	9	亮蓝色
2	绿色	10	亮绿色
3	青色	11	亮青色
4	红色	12	亮红色
5	洋红色	13	亮洋红色
6	黄色	14	亮黄色
7	白色	15	亮白色



提示

实际上，对于颜色的十六进制数，每两位一组代表一种原色的颜色值，最低两位为红色的值，其次是绿色和蓝色的值。例如，十六进制数 &H00FF00FF 对应 RGB (255, 0, 255)，因此，它表示的颜色为洋红色。

19.1.3 绘图方法

Visual Basic 中提供有一些图形绘制方法，通过这些方法我们能够绘制出丰富多彩的图形。

1. Pset 方法

该方法用来设置指定点处像素的色彩，我们可以使用该方法来绘制任意形状的图形。Pset 方法的语法如下。


```
[object.]PSet [step](x, y)[, color]
```

Pset 语句中各个关键字的描述如表所示。

关键字	描述
Object	可选部分，如果 object 省略，具有焦点的窗体作为 object
Step	可选的关键字，指定相对于由 CurrentX 和 CurrentY 属性提供的为当前图形位置的坐标
(x, y)	必需的参数，为单精度浮点数，指定设置点的水平（x 轴）和垂直（y 轴）坐标
Color	可选的参数，为长整型数，用于为指定点设置颜色。如果省略，则使用当前的前景色属性值

下面通过一个实例来学习 Pset 方法的使用。

【范例 19-2】使用 Pset 方法，在窗体上绘出正弦曲线。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 将 Form1 的 Caption 属性设置为“Pset 方法应用实例”。



- (3) 在 Form1 的空白处右击，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，输入以



下代码（代码 19-2.txt）。

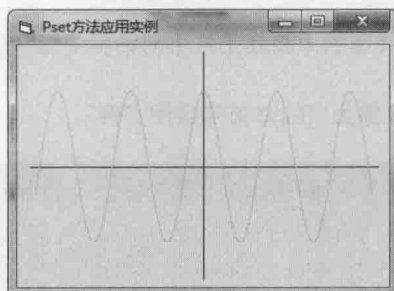
```

01 Private Sub Form_Click() ' 窗体单击事件
02   Dim x, y, i As Single ' 定义坐标值变量 x 和 y 以及循环参数 i
03   Scale (-16, 16) (-16, -16) ' 自定义坐标系
04   Line (0, 15) -(0, -15) ' 绘制垂直直线
05   Line (15, 0) - (-15, 0) ' 绘制水平直线
06   For i = -14.5 To 14.5 Step 0.01
07     x = i ' 正弦曲线参数
08     y = 10 * Cos(i) ' 正弦曲线参数
09     PSet (x, y), vbGreen ' 绘制正弦曲线
10   Next i
11 End Sub
12 Private Sub Form_DblClick() ' 窗体双击事件
13   Cls ' 清除屏幕内容
14 End Sub

```

## 【运行结果】

按快捷键【F5】运行程序。在窗体中的任意位置处单击，将会出现如图所示的图形，双击窗体则可清除窗体上的图形。



### 2. Point 方法

该方法用来返回在窗体或图像边框控件上所指定点的 RGB 颜色。Point 方法的语法如下。

```
object.Point(x, y)
```

### 3. Line 方法

该方法用于在对象上绘制直线和矩形。Line 方法的语法如下。

```
object.Line[Step](x1, y1)[Step](x2,y2),[color].[B] [F]
```


Line 方法语句中各个关键字的描述如表所示。

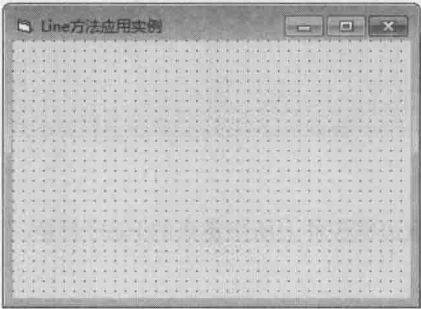


关键字	描述
Object	可选参数，为对象表达式。若 object 省略，具有焦点的窗体作为 object
Step	可选关键字，用于指定起点坐标相对于由 CurrentX 和 CurrentY 属性提供的当前图形位置
(x1, y1)	可选参数，为单精度浮点数，用于指定直线或矩形的起点坐标
Step	可选参数，指定相对于线的起点的终点坐标
(x2, y2)	必需的参数，为单精度浮点数，用于指定直线或矩形的终点坐标
Color	可选参数，为长整型数，设定画线时用的 RGB 颜色。如果被省略，则使用 ForeColor 属性值，用户可以使用 RGB 函数或 QBColor 函数指定颜色
B	可选参数，选中参数则利用对角坐标画出矩形
F	可选参数，如果使用了 B 选项，则 F 选项规定矩形以矩形边框的颜色填充。不能不用 B 选项而用 F 选项。如果不用 F 只用 B，则矩形用当前的 FillColor 和 FillStyle 填充。FillStyle 的默认值为 transparent（透明的）

下面通过一个实例来学习使用 Line 方法。

【范例 19-3】使用 Line 方法，在窗体中绘制图形。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 将 Form1 的 Caption 属性设置为“Line 方法应用实例”。



- (3) 在 Form1 的空白处右击，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，输入以下代码（代码 19-3.txt）。

```
01 Const pi = 3.14159
02 Dim a, b
03 Private Sub Form_Click()
04    Cls      '清空窗体
05     Form1.ScaleMode = 3
06     Form1.Scale (-20, 20)-(20, -20)  '设置坐标系
```

```

07 Form1.DrawWidth = 1 ' 设置绘制宽度
08 Line (-20, 0)-(20, 0), vbRed ' 绘制 x 轴
09 Line (18.5, 0.5)-(20, 0), vbRed
10 Line -(18.5, -0.5), vbRed ' 绘制 x 轴箭头
11 Print "X" 显示 x
12 Line (0, -20)-(0, 20), vbRed ' 绘制 y 轴
13 Line (0.5, 18.5)-(0, 20), vbRed
14 Line -(-0.5, 18.5), vbRed ' 绘制 y 轴箭头
15 Print "Y" 显示 y
16 CurrentX = 1: CurrentY = -1: Print "0" ' 显示原点
17 For b = 0 To 19 Step 1 / 2 ' 绘制曲线
18 For a = 0 To 2 * pi Step pi / 1000
19 PSet (b * Cos(a) ^ 3, b * Sin(a) ^ 1), vbBlue
20 Next a
21 Next
22 End Sub

```

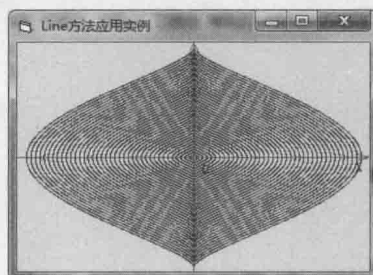
## 【代码详解】

单击窗体后，程序将首先清除窗体上的原有内容，定义窗体中央为坐标系原点，接着绘制出  $x$  轴和  $y$  轴以及箭头，并显示文字，然后调用两个数学函数，以 vbBlue 蓝色为颜色画图。

程序中的 Line 方法用于绘制坐标轴和坐标轴箭头。

## 【运行结果】

按快捷键【F5】运行程序。在窗体中的任意位置处单击，将会出现如图所示的图形。



### 4. Circle 方法

该方法用于在对象上画圆、椭圆、弧以及扇形。

Circle 方法的语法如下：

```
object.Circle [Step] (x, y), radius, [color, start, end, aspect]
```


语句中各个关键字的描述如表所示。

关键字	描述
Object	可选参数，为对象表达式，如果 object 省略，具有焦点的窗体作为 object
Step	可选关键字，用于指定圆、椭圆或弧的中心，相对于当前对象的 CurrentX 和 CurrentY 属性提供的坐标
(x, y)	必需参数，为单精度浮点数，用于指定圆、椭圆或弧的中心坐标。对象的 ScaleMode 属性决定了使用的度量单位
Radius	必需参数，为单精度浮点数，用于指定圆、椭圆或弧的半径
Color	可选参数，为长整型数，用于设置圆轮廓的 RGB 颜色。如果被省略，则使用 ForeColor 属性值，也可以使用 RGB 函数或 QBColor 函数指定颜色
start, end	可选参数，为单精度浮点数。start 和 end 指定（以弧度为单位）弧的起点和终点位置。其范围为 0 ~ 2 像素时为圆弧。当这两个参数的前面均有负号时，绘制的是扇形
Aspect	可选参数，为单精度浮点数。用于指定圆的纵横尺寸比。默认值为 1.0，即为标准圆（非椭圆）。aspect 既可以是整数表达式，也可以是小数表达式，但不能是负数

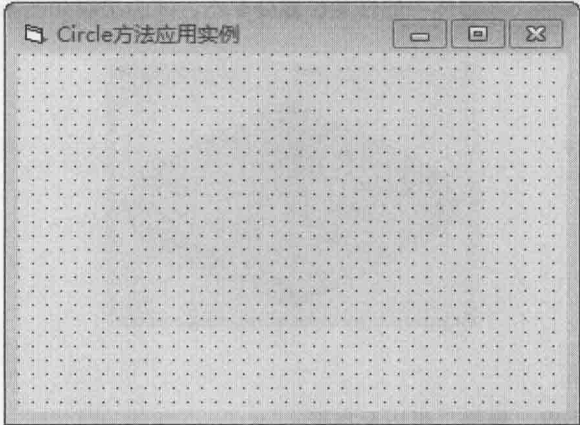
下面通过一个实例来学习 Circle 方法的使用。

【案例 19-4】

【范例 19-4】使用 Circle 方法，在窗体中画出不同的形状。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 将 Form1 的 Caption 属性设置为“Circle 方法应用实例”。

【案例 19-4】



- (3) 在 Form1 的空白处右击，在弹出的快捷菜单中选择【查看代码】选项，进入代码窗口，输入以下代码（代码 19-4.txt）。

```
01 Const PI = 3.14159
02 Private Sub Form_Click()
03    Cls
04     FillStyle = 0      '画一个实心椭圆
```

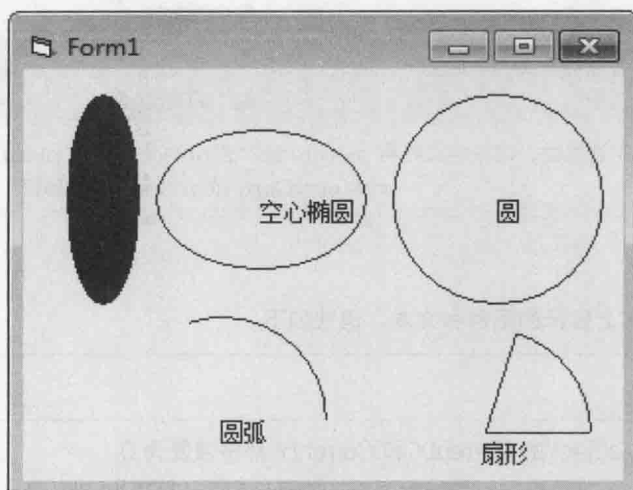
```

05 FillColor = vbBlue
06 Circle (600, 1000), 800, vbBlue, , , 3
07 FillStyle = 1 '画一个空心椭圆
08 Circle (1800, 1000), 800, vbBlue, , , 2 / 3
09 Print " 空心椭圆 "
10 Circle (3500, 1000), 800, vbBlue '画一个圆
11 Print " 圆 "
12 Circle (3500, 2800), 800, vbBlue, -0.0001, -PI * 2 / 5 '画一个扇形
13 Print " 扇形 "
14 Circle (1500, 2700), 800, vbBlue, 0.0001, PI * 3 / 5 '画一个圆弧
15 Print " 圆弧 "
16 End Sub

```

## 【运行结果】

按快捷键【F5】运行程序。在窗体中的任意位置处单击，将会出现如图所示的图形。



### 5. PaintPicture 方法

该方法用于在窗体和图像框等控件上绘制图形内容。语法如下。

```
object.PaintPicture picture, x1, y1, width1, height1, x2, y2, width2, height2, opcode
```

语句中各个关键字的描述如表所示。

部分	描述
Object	可选参数, 为一个对象表达式。如果省略 object, 带有焦点的窗体对象默认为 object
Picture	必需参数, 指定要绘制到对象上的图形源。窗体或图像框必须是 Picture 属性
Width1	可选参数, 为单精度值, 指定 picture 的目标宽度。object 的 ScaleMode 属性决定使用的度量单位。如果目标宽度比原宽度 (width2) 大或小, 将适当地拉伸或压缩 picture。如果该参数省略, 则使用原宽度
x1, y1	必需参数, 为单精度值, 用于指定在 object 上绘制 picture 的目标坐标
Height1	可选参数, 为单精度值, 指定 picture 的目标高度。如果目标高度比原高度 (height2) 大或小, 将适当地拉伸或压缩 picture。如果该参数省略, 则使用原高度
x2, y2	可选参数, 为单精度值, 指定 picture 内剪贴区的坐标。如果该参数省略, 则默认为 0
Width2	可选参数, 为单精度值, 指定 picture 内剪贴区的源宽度。如果该参数省略, 则使用整个源宽度
Height2	可选参数, 为单精度值, 指定 picture 内剪贴区的源高度。如果该参数省略, 则使用整个源高度
Opcode	可选参数, 用来定义在将 picture 绘制到 object 上时对 picture 进行的位操作 (例如 vbMergeCopy 或 vbSrcAnd 操作符)

## 6. Cls 方法

该方法用于清除窗体上显示的图形和文本。语法如下。

```
object.Cls
```

调用 Cls 方法之后, object 的 CurrentX 和 CurrentY 都将恢复为 0。

## 19.2 多媒体应用编程



本节视频教学录像: 8 分钟

除了图形外, 在 Visual Basic 中还可以在程序中加入音频、数字视频、模拟视频、动画、图像和文本等多种多媒体资源。我们可以通过多媒体控制接口控件使用这些多媒体资源。

### 19.2.1 多媒体控制接口控件基本概念

使用 CD 机或者收录机听音乐时, 可通过 CD 机或者收录机上的各种按钮 (如“播放”、“快进”、“暂停”、“停止”等) 来控制音乐的播放。多媒体控制接口 (MCI) 控件就像是一组按钮, 被用来向诸如声卡、MIDI 序列发生器、CD-ROM 驱动器、视频 CD 播放器和视频磁带记录器及播放器等设备发出 MCI 命令,

从而控制多媒体文件的回放。

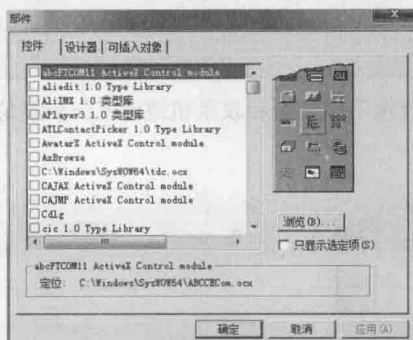


#### 提示

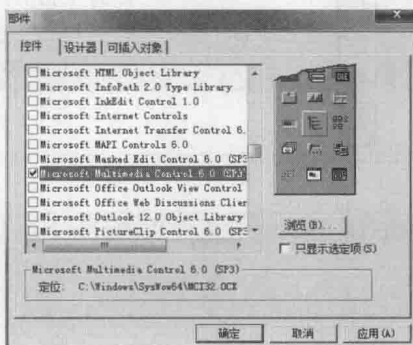
MCI 的全称是 Multimedia Control Interface，即多媒体控制接口。用户可以利用这个接口跳过硬件访问并调用本地符合 MCI 规格的多媒体解码驱动，从而完成对多媒体的回放录制与控制。

在研究多媒体控制接口的各种概念之前，首先需要把多媒体控制接口控件添加到工具栏中。具体的操作步骤如下。

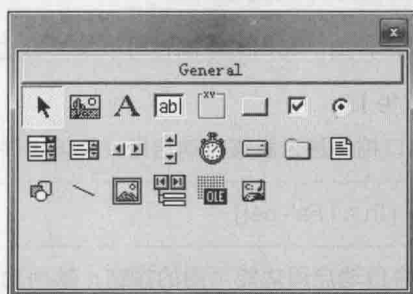
(1) 在 Visual Basic 6.0 中选择【工程】>【部件】菜单命令，弹出【部件】对话框。



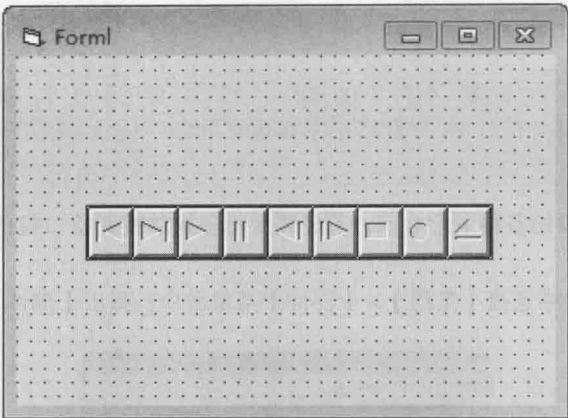
(2) 选择【控件】选项卡，选中【Microsoft Multimedia Control 6.0】复选框，然后单击【确定】按钮。



(3) 这样就将多媒体控制接口控件添加到了工具箱中。



(4) 双击工具箱中的多媒体控制接口控件 (MMControl) 按钮，即可在窗体中创建一个多媒体控制接口控件。



(5) 可以看到多媒体控制接口控件不仅功能和收录机或者 CD 机类似，连外形也十分相似。这些按钮的功能如表所示。

按钮功能	图标	按钮功能	图标
上一个 (Prev)		下一个 (Next)	
播放 (Play)		暂停 (Pause)	
后退 (Back)		向前 (Step)	
停止 (Stop)		录制 (Record)	
弹出 (Eject)			

在一个窗体中可以加入多个 MCI 控件的实例，以提供对多个 MCI 设备的控制，但对每个设备只能使用一个 MCI 控件。

提示

19.2.2 多媒体控制接口控件的属性

MCI 控件提供了非常丰富的属性种类，利用这些属性可以编写出功能强大的多媒体应用程序。

1. 自动启用属性 (AutoEnable)
- 该属性用于设置多媒体控制接口控件是否能够自动启用或关闭控件中的某个按钮。语法如下。

```
[form.]MMControl.AutoEnable[ = {True | False}]
```

当该属性值为 True 时，控件将自动启用功能可用的按钮，禁用功能不可用的按钮；当该属性值为 False 时，不能自动启用或者禁用按钮。除此之外，当 MCI 控件处于不同的模式时，控件中各个按钮的启用状态也是不同的。

用户可以在属性窗口中的【AutoEnable】下拉列表中设置该属性。





## 2. 按钮启用属性 ( ButtonEnabled )

该属性用于设置是否启用或禁用控件中的某个按钮。当该属性值为 True 时，按钮处于启用状态；当该属性值为 False 时，按钮将被禁用。如果某个按钮被设置为禁用，将以灰色的形式显示。语法如下。

```
[form.]MMControl.ButtonEnabled[ = {True | False}]
```

需要注意的是，只有当自动启动属性 ( AutoEnable ) 设置为 False 时，才能使用 ButtonEnabled 属性启用或者禁用一个按钮。

## 3. 可否弹出属性 ( CanEject )

该属性用于设置打开的 MCI 设备能否将其媒体弹出。当该属性值为 True 时，可以将媒体弹出；当该属性值为 False 时，不能将媒体弹出。语法如下。

```
[form.]MMControl.CanEject
```

## 4. 命令属性 ( Command )

该属性用于设置准备执行的 MCI 命令。语法如下。

```
[form.]MMControl.Command[ = cmdstring$]
```

语句中的 cmdstring\$ 参数就是 MCI 的实际命令。

## 5. 设备 ID 属性 ( DeviceID )

该属性是只读的，用于指定或者返回当前打开的 MCI 设备的 ID。在设计阶段，该属性不可用，在运行阶段它是只读的。语法如下。

```
[form.]MMControl.DeviceID[ = id%]
```

# 19.2.3 多媒体控制接口控件的事件

多媒体控制接口控件的常用事件有以下几个。

## 1. ButtonClick 事件

该事件在用户完成一次 Click 操作后被触发。所谓一次 Click 操作，是指用户按下鼠标再释放鼠标的整个过程。语法如下。

```
Private Sub MMControl_ButtonClick (Cancel As Integer)
```

在语句中，Button 可以是以下按钮中的任意一种：Back、Eject、Next、Pause、Play、Prev、Record、Step 或 Stop 等。但是单击这些按钮所进行的默认操作是不同的，下表列出了各个按钮的 ButtonClick 事件所对应的命令。

按钮名称	命令
Back	MCI_STEP
Step	MCI_STEP
Play	MCI_PLAY
Pause	MCI_PAUSE
Prev	MCI_SEEK
Next	MCI_SEEK
Stop	MCI_STOP
Record	MCI_RECORD
Eject	MCI_SET

2. ButtonCompleted 事件

该事件当多媒体控制接口控件激活的 MCI 命令结束时被触发。语法如下。

```
Private Sub MMControl_ButtonCompleted (Err- orcode As Long)
```

3. ButtonGotFocus、ButtonLostFocus 事件

ButtonGotFocus 和 ButtonLostFocus 事件是一对相互的事件，ButtonGotFocus 在多媒体控制接口控件中的按键获得焦点时被触发，而 ButtonLostFocus 则在多媒体控制接口控件中的按键失去焦点时被触发。语法如下。

```
Private Sub MMControl_ButtonGotFocus ()  
Private Sub MMControl_ButtonLostFocus ()
```

4. Done 事件

该事件当 Notify 属性为 True 的 MCI 命令结束时被触发。语法如下。

```
Private Sub MMControl_Done (NotifyCode As Integer)
```

语句中的 NotifyCode 参数代表 MCI 命令是否成功执行。NotifyCode 参数值及其对应的含义如表所示。

值	常量	含义
1	mciSuccessful	命令被成功执行
2	mciSuperseded	命令被其他命令所替代
4	mciAborted	命令被用户中断
8	mciFailure	命令失败

### 5. StatusUpdate 事件


该事件按照 UpdateInterval 属性所给定的时间间隔自动被触发。语法是：

```
Private Sub MMControl_StatusUpdate ()
```

## 19.2.4 多媒体控制接口控件应用实例

本小节通过一个实例来学习多媒体控制接口控件的使用方法。

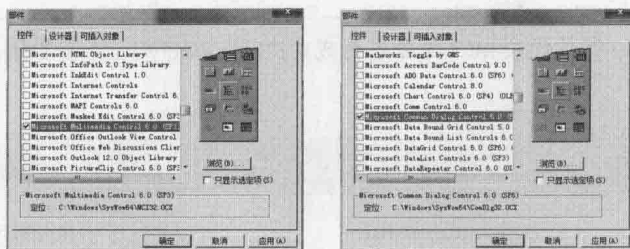
### 【范例 19-5】使用多媒体控制接口控件，设计一个简单的媒体播放器。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

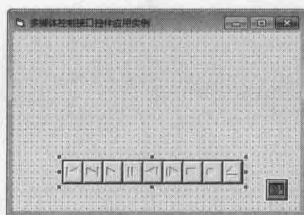
(2) 将 Form1 的 Caption 属性改为“多媒体控制接口控件应用实例”。



(3) 选择【工程】>【部件】菜单命令，弹出【部件】对话框，在【控件】选项卡中选中【Microsoft Multimedia Control 6.0】和【Microsoft Common Dialog Control 6.0】两个复选框，然后单击【确定】按钮，多媒体控件就被添加到了工具箱中。

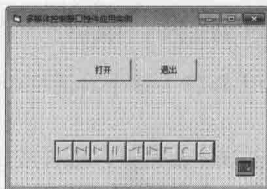


(4) 在窗体中添加 1 个多媒体控制接口 (MMControl) 控件, 将其名称改为 “MCI1”; 添加 1 个通用对话框 (CommonDialog), 将其名称改为 “CommonDLG1”。



(5) 在窗体中添加两个按钮 (CommandButton) 控件, 按照下表设置其属性。

名称	Caption	控件作用
CmdOpen	打开	打开对话框, 浏览并打开文件
CmdExit	退出	退出程序



(6) 双击 “打开” 命令按钮控件, 在弹出的代码窗口中添加以下代码 (代码 19-5.txt)。

```

01 Private Sub CmdOpen_Click()      ' 打开按钮单击事件
02     CommonDLG1.FileName = ""
03     CommonDLG1.ShowOpen
04     If CommonDLG1.FileName <> "" Then    ' 如果选定了文件
05         MCI1.FileName = CommonDLG1.FileName    ' 打开指定的文件
06         MCI1.Command = "open"            ' 使用 MCI 控件打开文件
07     End If
08 End Sub

```

## 【代码详解】

单击【打开】命令按钮后, 调用对话框供浏览并打开文件, 如果打开的是媒体文件, 则调用多媒体

控制接口控件 MCI1 打开文件，开始播放。

## 【运行结果】

按快捷键【F5】运行程序。利用这个程序可以调用 Visual Basic 的多媒体控制接口控件来打开播放多媒体文件。

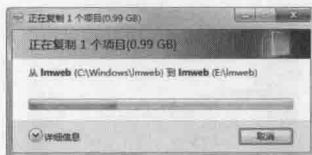


## 19.3 让程序动起来——动画应用编程



本节视频教学录像：6 分钟

动画控件（Animation）用来播放 AVI 文件。在 Windows 操作系统中有不少 AVI 动画文件，例如安装程序时的进度条动画以及复制文件时的【正在复制...】动画。



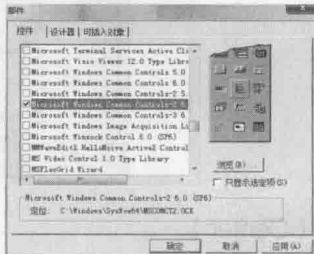
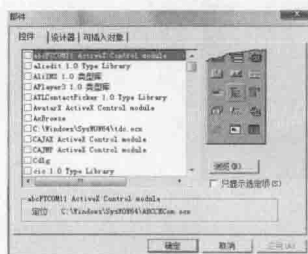
**注意**

有的 AVI 动画带有声音，但这样的动画不能在动画控件中使用，如果试图打开这样的文件，将会产生错误。在动画控件中只能使用无声的 AVI 动画。要播放有声的 AVI 文件，可以使用多媒体控制接口控件。

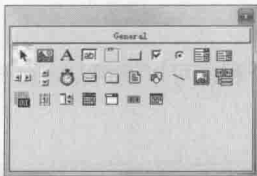
### 19.3.1 添加动画控件

将【动画控件】按钮添加到工具箱中的具体步骤如下。

- (1) 在 Visual Basic 6.0 中选择【工程】>【部件】菜单命令，弹出【部件】对话框。
- (2) 选择【控件】选项卡，选中【Microsoft Windows Common Controls-2 6.0】复选框，单击【确定】按钮。



(3) 这样就将动画（Animation）控件添加到了工具箱中。



在程序运行时，动画控件是不可见的。

### 19.3.2 动画控件的属性

动画控件的主要属性有以下几种。

#### 1. 自动播放属性（AutoPlay）

该属性用于设置或者返回一个值，确定动画控件是否开始播放已加载到该控件上的 .avi 文件并不断重复播放该文件。当该属性值为 True 时，将不断重复播放 AVI 文件；当该属性值为 False 时，将停止播放 AVI 文件。用户可以在属性窗口中的【AutoPlay】下拉列表中设置该属性。



#### 2. 背景风格属性（BackStyle）

该属性返回或设置一个值，该值确定动画控件所使用的背景是否透明。当该值为 0 时，动画控件的背景透明；当该值为 1 时，动画控件的背景不透明。用户可以在属性窗口中的【BackStyle】下拉列表中设置该属性。



#### 3. 居中播放属性（Center）

该属性用于设置播放动画时是否在控件中央播放。当该属性值为 True 时，在控件中央播放动画文件；当该属性值为 False 时，在控件的原点 (0, 0) 处播放动画文件。用户可以在属性窗口中的【Center】下拉列表中设置该属性。



### 19.3.3 动画控件的方法

#### 1. 打开方法 (Open)

该方法用于打开一个 AVI 动画文件，如果该动画控件的 AutoPlay 属性设置为 True，则一旦动画文件成功加载，将不断重复播放。

打开方法的语法如下。

---

```
object.Open file
```

---

#### 2. 播放方法 (Play)

该方法用于播放已经加载的 AVI 动画文件。

播放方法的语法如下。

---

```
object.Play [= repeat, start, end]
```

---

语句中的关键字含义如表所示。

部分	含义
Object	对象表达式，其值是动画控件
Repeat	指定重复剪辑的次数，默认值是 -1，使重复剪辑次数不受限定

#### 3. 停止方法 (Stop)

该方法用于停止播放已经加载的 AVI 动画文件。

停止方法的语法如下。

---

```
object.Stop
```

---



4. 关闭方法（Close）


该方法用于关闭当前加载 AVI 动画文件。  
关闭方法的语法如下。

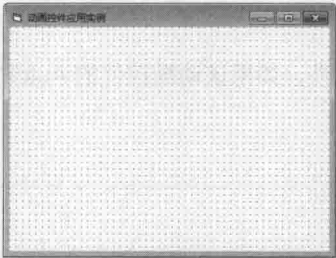
```
object.Close
```

19.3.4 动画控件应用实例

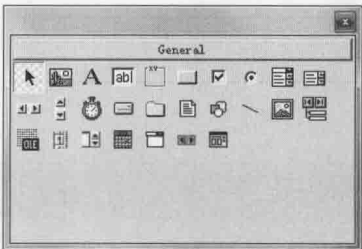
本小节通过一个实例来学习动画控件的使用方法。

【范例 19-6】使用动画控件打开动画文件，并可设定播放次数。

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标，单击【打开】按钮。
- (2) 将 Form1 的 Caption 属性设置为“动画控件应用实例”。



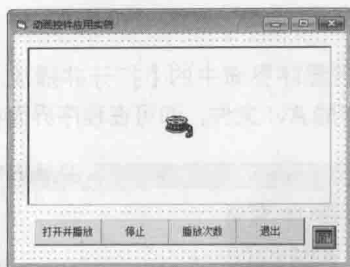
- (3) 选择【工程】>【部件】菜单命令，弹出【部件】对话框，在【控件】选项卡中选择【Microsoft Windows Common Controls-2 6.0】和【Microsoft Common Dialog Control 6.0】两个复选框，然后单击【确定】按钮。控件添加后的工具箱如图所示。



- (4) 在窗体中添加 1 个动画（Animation）控件、1 个 CommomDialog 控件；1 个包含 4 个命令（CommandButton）按钮的控件组，并按照下表所示设置控件的属性。

控件名称	Caption	控件作用
Command1(0)	打开并播放	指定打开文件并播放
Command1(1)	停止	停止播放
Command1(2)	播放次数	指定打开文件并播放指定次数
Command1(3)	退出	退出程序

(5) 将控件添加、设置完成,按照下图所示排列各控件。



(6) 双击【打开并播放】命令按钮控件,在弹出的代码窗口中添加以下代码(代码 19-6.txt)。

```

01 Private Sub Command1_Click(Index As Integer)           '命令控件组代码
02     Dim m As Integer
03     If Index = 0 Then                                   '打开并播放按钮代码
04         CommonDialog1.Filter = "avi 文件 (*.avi)|*.avi"   '选择打开的文件类型
05         CommonDialog1.ShowOpen                             '显示打开对话框
06         Animation1.Open CommonDialog1.FileName           '打开指定文件
07         Animation1.Play                                   '播放
08     End If
09     If Index = 1 Then                                   '停止按钮代码
10         Animation1.AutoPlay = False                     '停止自动播放
11         Animation1.Stop                                  '停止播放
12     End If
13     If Index = 2 Then                                   '播放次数按钮代码
14         CommonDialog1.Filter = "avi 文件 (*.avi)|*.avi"   '选择打开的文件类型
15         CommonDialog1.ShowOpen                             '显示打开对话框
16         Animation1.Open CommonDialog1.FileName           '打开指定文件
17         m = InputBox("请输入播放次数", "2")              '输入播放次数
18         Animation1.Play m                                 '播放指定次数
19     End If
20     If Index = 3 Then                                   '退出按钮代码
21         Animation1.Close                                  '关闭控件
22     End '退出程序
23 End If
24 End Sub

```

## 【代码详解】

CommonDialog1.Filter = "avi 文件 (\*.avi)|\*.avi" 用于指定在【打开】对话框中,只显示文件夹或 AVI 格式的文件。

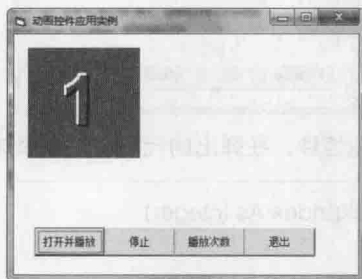
Animation1.Play 调用 Play 方法播放在【打开】对话框中选择的文件。如果按下【停止】按钮,则调用 Animation1.Stop 方法停止播放 AVI 文件。

如果按下【播放次数】按钮,则弹出对话框要求输入次数,然后 Animation1.Play m 方法播放动画

m 次。

## 【运行结果】

按快捷键【F5】运行程序。单击程序界面中的【打开并播放】按钮，浏览并打开随书光盘中的“Sample\ch19\范例 19-6”文件夹中的 AVI 文件，即可在程序界面中播放。



## 19.4 高手点拨



本节视频教学录像：2 分钟

VB 中的动画由两个基本部分组成：一是物体相对于屏幕的运动，即屏幕级动画；二是物体内部的运动，即相对符号的动画。制作动画的原理就是画完一幅图形，清除它的屏幕显示部分，再在新位置画第二幅图形，如此交替下去，利用人眼的视觉效应，就可以产生动画效果，VB 实现动画的原理也是如此，但 VB 不要求编程人员详细了解图形如何再现和清除，这些工作由 VB 提供的工具来做，这样就使 VB 实现动画很方便，编程也很简捷。

## 19.5 实战练习

### 一、思考题

1. 写出在窗体坐标 (3 500,1 000) 处绘制一个半径为 800 的圆形的语句。
2. 写出恢复窗体 Form1 的坐标系为默认坐标系的语句。

### 二、上机题

仿照本章【范例 19-1】和【范例 19-4】设计一个程序，要求完成以下的功能。

- (1) 以窗体中心为窗体坐标原点，绘制坐标系  $x$  轴和  $y$  轴以及原点。
- (2) 以坐标系原点为圆心，绘制一个半径为 200、颜色为蓝色的圆。

# 第20章



本章视频教学录像：28 分钟

## 用 VB 操纵文件——文件系统编程

Visual Basic 提供有很强的文件处理功能，通过使用所提供的文件处理方法、函数及与文件系统有关的语句，可以将数据很容易地读出显示、写入文件并存储等。本章主要讲述文件的类型与结构，以及对顺序文件、随机文件和二进制文件等的基本操作。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解文件的类型和结构
- ☐ 掌握文件操作语句
- ☐ 掌握文件操作函数
- ☐ 熟悉二进制文件
- ☐ 掌握顺序文件的打开、读取、写入和关闭
- ☐ 掌握随机文件的打开、读取、写入和关闭

## 20.1 文件的类型与结构



本节视频教学录像: 4 分钟

文件是存储在外部介质(如磁盘、光盘)上的数据或信息的集合。在日常学习和工作中我们接触最多的便是文件,如 .txt 格式的文件和 .doc 格式的文件等。

### 20.1.1 文件结构

应用程序在对文件中的数据进行读写时,必须根据数据的存放格式来进行。数据的这种存放格式就是文件结构。在 Visual Basic 中,文件由记录组成,记录由字段组成,字段又由字符所组成。

(1) 字符:字符是文件组成中的最基本单位,每一个字节、数字及标点符号等都是一个字符。

(2) 字段:字段的主要构成单位是字符,一个或几个字符所组成的独立的数据就称为一个字段,多个字符的序列就构成了字段值。

(3) 记录:记录是由一组字段所组成的一个逻辑单位。在一个记录中,各个字段之间若有关系,则应使用记录名来表示。

(4) 文件:将具有同一结构的记录按照一定的顺序所组成的集合,就构成了文件。每个文件都有唯一的文件名,它是访问文件数据的唯一手段。

### 20.1.2 文件类型

在 Visual Basic 中根据数据文件的结构和访问方式的不同,可以将文件分为顺序文件、随机文件和二进制文件等 3 种。

(1) 顺序文件。它具有简单的文本结构,按照一定的顺序来存储数据,是为普通的文本文件的使用而设计的。

(2) 随机文件。随机文件的结构与数据库文件的结构极为相似,它是由相同格式的记录组成的。根据这一特点,对随机文件进行数据的读取、插入、修改、删除及查找时,只需要根据记录号进行相关的操作即可。

(3) 二进制文件。二进制文件中的数据是以字节为单位而存取的,它更适用于读写任意有结构的文件。

## 20.2 文件操作语句



本节视频教学录像: 1 分钟

在 Visual Basic 中常用的文件操作语句主要有以下 7 个,主要用来查找文件所处的位置,对文件进行命名、复制、删除和修改属性值等操作。

(1) ChDrive 语句:用于改变当前文件所处的驱动器,语法如下。

ChDrive drive

例如:

ChDrive "E"

' 将 E 盘设置为当前驱动器



#### 注意

如果使用零长度的字符串(""),则当前的驱动器不会改变。如果 drive 参数中有多个字符,则 ChDrive 语句只会使用首字母。

(2) ChDir 语句:用于改变当前文件的目录或文件夹,语法如下。

ChDir path

例如,通过下面的语句可以将驱动器修改至 E 盘上。

ChDir "E:\MyFolder" ' 驱动器在运行时将默认的驱动器重新设置为 E 盘



#### 技巧

ChDir 语句改变默认目录位置,但不会改变默认驱动器位置。

(3) Kill 语句:用于从磁盘中删除文件。Kill 语句支持多字符(\*)的通配符和单字符(?)的通配符来指定多重文件。语法如下。

Kill pathname

例如:

Kill "D:\File.txt" ' 删除 D 盘根目录下的“File.txt”文件

Kill "D:\\*.txt" ' 删除 D 盘根目录下的所有 txt 文件



#### 提示

如果 D 盘下没有 File.txt 文件,则无法删除,程序会报错。

(4) Mkdir 语句:用于创建一个新的目录或文件夹。语法如下。

Mkdir path

例如:

Mkdir "D:\VB" ' 在 D 盘根目录下创建一个 VB 文件夹



#### 注意

Path 参数可以包含驱动器。如果没有指定驱动器, Mkdir 语句则会在当前驱动器上创建新的目录或文件夹。在创建该文件前,应首先确定在该路径下不存在同名的文件夹,如果存在同名的文件夹,程序会报错。

(5) FileCopy 语句：用于复制一个文件。语法如下。

FileCopy source, destination

例如：

FileCopy C:\VB\vb.txt, D:\VB ' 将 C 盘根目录下 VB 文件夹中的 vb.txt 文件复制到 D 盘根目录下的 VB 文件夹中



**注意**

如果对一个已打开的文件使用 FileCopy 语句，程序会报错。

(6) Name 语句：用于重新命名一个文件、目录或文件夹。语法如下。

Name oldpathname As newpathname

例如：

Name D:\old\old.txt As D:\new\new.txt' 将 D 盘根目录下的 old 文件夹更改为 new 文件夹，并将原 old 文件夹下的 old.txt 文件更名为 new.txt 文件



**提示**

由 newpathname 所指定的文件名必须存在。同时，在一个已打开的文件上使用 Name，将会产生错误。必须在改变名称之前，先关闭打开的文件。Name 语句不能包括多字符 (\*) 和单字符 (?) 的通配符。

(7) SetAttr 语句：用于为一个文件设置属性信息。语法如下。

SetAttr pathname, attributes

其中，attributes 参数可以参照如表的值进行设置。


常 数	值	描 述
vbNormal	0	常规 (默认值)
vbReadOnly	1	只读
vbHidden	2	隐藏
vbSystem	4	系统文件
vbArchive	32	上次备份以后，文件已经改变

例如：

SetAttr D:\vb.txt, vbReadOnly ' 设置 vb.txt 文件属性为只读



## 【范例 20-1】通过单击【复制文件】按钮，将一个文件夹下的源文件复制到目标文件夹下。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。


(2) 在 Form1 窗体上添加一个命令按钮 (CommandButton) 控件，将其【Caption】属性设置为“复制文件”。

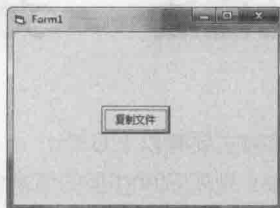


(3) 在 Form1 窗体中双击【复制文件】按钮，在打开的代码窗口中输入以下代码 (代码 20-1.txt)。

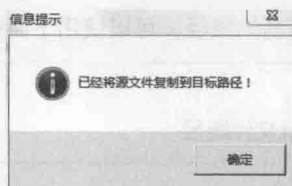
```
01 Private Sub Command1_Click()  
02 Dim SourceFile As String  
03 Dim DestinationFile As String      ' 定义变量  
04 SourceFile = "D:\Final\ch 20\范例 20-1\复制文件\源文件\VB 代码.txt" 访问源文件  
05 DestinationFile = "D:\Final\ch20\范例 20-1\复制文件\目标文件\VB 代码.txt" ' 访问目标文件  
06 FileCopy SourceFile, DestinationFile' 将源文件复制到目标文件  
07 MsgBox "已经将源文件复制到目标路径!", vbInformation, "信息提示" ' 提示信息  
08 End  
09 End Sub
```

## 【运行结果】

(1) 保存程序，单击【启动】按钮 ，运行程序。



(2) 单击【复制文件】按钮，弹出【信息提示】对话框，说明已成功对文件进行复制。

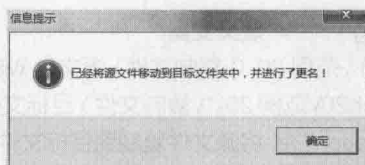


## 【拓展训练 20-1】

在窗体上放置一个命令按钮, 设置其【Caption】属性为“移动文件并重命名”。运行程序后, 单击该命令按钮, 将“源文件\oldfile.txt”移动至“目标文件”夹下, 并将“oldfile.txt”文件更名为“newfile.txt”文件。其 Command 命令按钮的代码如下 ( 拓展代码 20-1.txt )。

```
01 Private Sub Command1_Click()  
02 Dim OldName As String  
03 Dim NewName As String ' 定义变量  
04 OldName = "D:\Finalch16\拓展训练 20-1\移动文件并重命名\源文件\oldfile.txt" ' 访问文件  
05 NewName = "D:\Finalch16\拓展训练 20-1\移动文件并重命名\目标文件\newfile.txt"  
    ' 创建新文件夹并对源文件夹中的文件重命名  
06 Name OldName As NewName ' 文件重命名  
07 MsgBox " 已经将源文件移动到目标文件夹中, 并进行了更名! ", vbInformation, " 信息提示 "  
08 End  
09 End Sub
```

保存程序, 按【F5】快捷键, 运行程序。单击【移动文件并重命名】按钮, 弹出【信息提示】对话框, 说明已成功对文件进行移动和重命名。



**注意**

在移动文件时, 应确保源文件夹下有所指定的文件存在。本实例中移动文件后, 源文件夹已成为了空文件夹。

## 20.3 操纵文件的魔法——文件操作函数



本节视频教学录像: 2 分钟

在 Visual Basic 中常用的文件操作函数主要有以下 6 个。

(1) CurDir 函数。用于返回一个 Variant 型或 String 型的值来代表当前的路径。语法如下。

```
CurDir[(drive)]
```

例如需要使用 CurDir 函数来返回当前的路径, 可以使用下面的语句。

```
Dim MyPath ' 定义变量名  
MyPath = CurDir ' 返回当前系统文件路径
```



如果没有指定驱动器，或 drive 是零长度字符串 ("")，CurDir 函数则会返回当前驱动器的路径。

#### 注意

(2) GetAttr 函数。用于返回一个 Integer 类型的值。语法如下。

GetAttr(pathname)

例如，在下面的 And 表达式中，如果 vbArchive 属性没有设置，返回值则为零；如果文件的 vbArchive 属性已经设置，则返回非零的数值。

Result = GetAttr(FileName) And vbArchive

GetAttr 函数的返回值如表所示。

常 数	值	描 述
vbNormal	0	常规
vbReadOnly	1	只读
vbHidden	2	隐藏
vbSystem	4	系统文件
vbDirectory	16	目录或文件夹
vbArchive	32	上次备份以后，文件已经改变
vbAlias	64	指定的文件名是别名



#### 技巧

若要判断是否设置了某个属性，可在 GetAttr 函数与想要得到的属性值之间使用 And 运算符逐位进行比较。如果所得的结果不为零，就表示设置了这个属性值。

(3) FileDateTime 函数。用于获得文件创建或最近修改的日期与时间值。它返回的是一个 Variant 型或 Date 型的值。语法如下。

FileDateTime(pathname)

例如在 D 盘下，有一个“VB.txt”文件，上次被修改的时间为 2009 年 11 月 25 日 9 时 30 分 25 秒，如果需要获取其最后的修改时间，则可使用以下代码。

FileTime = FileDateTime("D:\VB.txt") ' 运行后变量 FileTime 的值为 2009-11-25 9:20:25

(4) FileLen 函数。用于返回一个 Long 型值。它代表一个文件的长度，单位是字节。  
语法如下：

---

FileLen(pathname)

---

例如在 D 盘的根目录下, 有一个文件名为 “VB.txt”, 该文件中含有数据, 其字节长度为 20。

---

Dim FileSize

FileSize = FileLen("D:\VB.txt") ' 返回文件的字节长度为 20

---


**注意**

当调用 FileLen 函数时, 如果所指定的文件已经打开, 返回的值则是这个文件在打开前的大小。若要取得一个打开文件的长度大小, 可以使用下面介绍的 LOF 函数。

(5) EOF 函数。用于测试文件的结束状态。语法如下。

---

EOF(filenummer)

---

例如在 D 盘下有一个 VB.txt 文件, 使用 EOF 函数检测文件是否已经读到末尾。

---

```
01 Open "D:\VB.txt" For Input As #1 ' 打开文件
02 Do While Not EOF(1) ' 检查是否到文件尾
03     Print #1, "#VB 文件内容" ' 读取文件
04 Loop
05 Close #1 ' 关闭文件
```

---


**提示**

关于文件的打开与关闭, 将在本章 20.5 节和 20.6 节中讲述。

(6) LOF 函数。用于返回一个 Long 型值, 它表示用 Open 语句打开的文件的大小, 以字节为单位。语法如下。

---

LOF(filenummer)

---


例如, 使用 LOF 函数来获取已经打开的 D:\VB.txt 文件的大小。

---

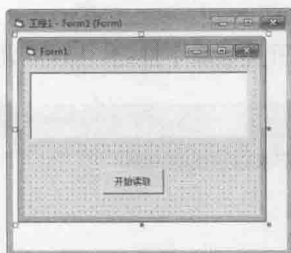
```
01 Dim MyFileLength
02 Open "D:\VB.txt" For Input As #1 ' 打开文件
03 MyFileLength = LOF(1) ' 取得文件长度
04 Close #1 ' 关闭文件
```

---

**【范例 20-2】**从磁盘的 txt 文件中整行读取数据, 然后将读取到的数据在窗体的文本框中显示。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。

(2) 在 Form1 窗体上添加一个命令按钮 (CommandButton) 控件和一个文本框 (TextBox) 控件, 将命令按钮控件的【Caption】属性设置为“开始读取”, 将文本框控件的【MultiLine】属性设置为“True”, 将【Text】属性设置为空, 将控件按下图进行排列。

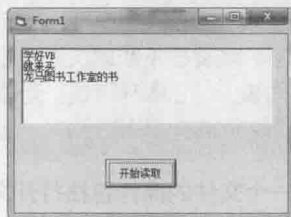


(3) 在 Form1 窗体中双击【开始读取】按钮, 在打开的代码窗口中输入以下代码 (代码 20-2.txt)。

```
01 Private Sub Command1_Click()  
02   Dim InPutData '定义变量  
03   Open "D:\Final\ch20\范例 20-2\整行读取文件\VB.txt" For Input As #1 '打开文件  
04   Do While Not EOF(1) '检查是否到文件末尾  
05     Line Input #1, InPutData '读取一行数据  
06     Text1.Text = Text1.Text + InPutData + vbCrLf '追加到文本框末尾  
07   Loop  
08   Close #1 '关闭文件  
09 End Sub
```

## 【运行结果】

保存程序, 按【F5】快捷键运行程序。单击【开始读取】按钮, 即可从指定文件中整行读取数据。



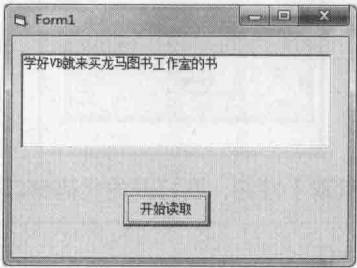
## 【拓展训练 20-2】

对【范例 20-2】中步骤(3)的代码进行如下修改, 使得从磁盘的 txt 文件中逐字读取数据, 然后将读取到的数据在窗体的文本框显示 (拓展代码 20-2.txt)。


```
01 Private Sub Command1_Click()  
02   Dim InPutData '定义变量  
03   Open "D:\Final\ch20\拓展训练 20-2\逐字读取数据\VB.txt" For Input As #1  
      '以绝对路径形式打开文件
```

```
04 Do While Not EOF(1) '检查是否到文件末尾
05   Input #1, InPutData '读取一个字符
06   Text1.Text = Text1.Text + InPutData '追加到文本框的末尾
07 Loop
08 Close #1 '关闭文件
09 End Sub
```

按【F5】快捷键运行程序，单击【开始读取】按钮，即可从指定的文件中逐字读取数据。



## 20.4 顺序文件

 本节视频教学录像：8 分钟

顺序文件是一种文件结构相对简单的文件，是以字符的形式按照先后顺序存储数据的。其结构简单，占用的磁盘空间较少，适用于大量数据的成批处理。

记录 1	记录 2	记录 3	.....	记录 N	EOF
------	------	------	-------	------	-----



**提示**

在实际中常见的磁带、录像带就是一种典型的顺序文件。在顺序文件中，其存储方式只提供了第 1 个记录的位置，其他记录的位置则无法获得。所以，查找顺序文件中的某个记录必须从文件头开始找起，逐个比较，直到找到目标为止。例如，我们在听磁带或者观看录像带的时候，如果想要找到某一段特定的内容，就需要不断地快进和倒带，直到找到为止。顺序文件的优点是：操作简单，占用的磁盘空间较少。它的缺点是：无法任意取出某一个记录进行修改，而必须将全部数据读入到内存中，修改操作才可以进行。

通常情况下，在 Visual Basic 中对一个文件的操作包括打开文件、操作文件内容和关闭文件等 3 个步骤。

### 20.4.1 顺序文件的打开

在对文件进行操作之前都必须先打开文件，同时指明文件所处的位置。打开顺序文件时使用的是 Open 语句，语法如下。

```
Open 文件名 [For 打开模式] [Access 访问方式] [Lock 类型] As[#] 文件号 [Len= 记录长度]
```



在上面的语句中各个参数的含义如下。

(1) “文件名”是指要打开的文件，可包含驱动器名及路径名。文件名必须用双引号括起来。

(2) “For 打开模式”是指文件打开的方式。有 3 种模式可以选择：Output (输出)、Input (输入) 和 Append (添加)。其中，Output 模式是将数据写入文件。如果文件不存在，就创建一个新文件；如果文件已存在，则删除文件中原有的数据，从头开始写入数据。Input 模式是将数据从文件中读入内存。Append 模式是将数据添加在文件尾部。如果文件不存在，就创建一个新文件。

(3) “Access 访问方式”是指文件的存取权限。有 3 个选项，分别是 Read 方式 (只读方式)、Write 方式 (只写方式) 和 Read Write 方式 (读写方式)。

(4) “Lock 类型”是用于设置文件打开后的读写操作特性。有 4 种方式，分别是 Shared (可对该文件读写)、LockRead (不允许读该文件)、LockWrite (不允许写该文件) 和 LockReadWrite (不允许读写该文件)。

(5) “[#] 文件号”是文件缓冲区号，它是一个介于 1 到 511 之间的整数。当打开一个文件并为它指定一个文件号后，文件号就可以代表该文件，直到该文件被关闭后，文件号才可以再供其他的文件使用。在使用文件号时，往往使用 FreeFile 函数获得可利用的文件号。

(6) “Len= 记录长度”是一个小于 32 767 的整数。对于顺序文件来说，它是指缓冲区大小，即所分配的字符个数。系统默认为 512 字节。

## 20.4.2 顺序文件的读取

打开文件后，就可以读取文件了。如果要读取顺序文件中的内容，应该以 Input 模式打开该文件，然后从顺序文件中读取数据。其语法格式有以下 3 种形式。

(1) Input # 文件号, 变量列表。该语句从顺序文件中读取一行数据，并将读出的数据依次放在给出变量列表的各个变量中。变量的顺序必须和文件中各个字段的顺序相同，即每个变量应与其对应的字段数据类型一致。当读取数据到顺序文件末尾时，将显示出错信息。该方法适用于数据类型不一致的数据文件。

(2) Line Input # 文件号, 字符串变量。该语句每次从文件中读取一行信息，以回车换行符作为分界符。其中的变量为字符串变量。该方法适用于以行为单位的文本文件。

(3) 变量 = Input\$(长度, [#] 文件号)。该语句使用 Input 函数进行顺序文件的读取，可以用在赋值语句中或者 Print 语句中。该语句的 Input 函数返回读取的所有字符，包括标点、符号、回车换行符以及空格等字符。该方法适用于文本的编辑操作。

## 20.4.3 顺序文件的写入

读取操作是把计算机中的文件内容取出，与其相对的是写入操作。写入操作是将数据存放到计算机文件中。要将数据写入文件，应该在 Open 语句中使用 Output 模式或者 Append 模式，然后用写入语句把准备好的数据写入该文件。当处于 Output 模式时，将新建一个文件，如果文件已经存在，则删除该文件中的所有数据，新建一个空的文件。如果在打开文件的过程中，想保留原文件的数据，则应该采用 Append 模式打开相应的文件。在 Append 模式下，写入语句会将新的数据追加到该文件的末尾存储。写入操作的语法格式有以下两种形式。



(1) Print # 文件号, [Spc(n) | Tab(n)][输出表达式列表] [; | ,]

在该语法格式中,“文件号”是 Open 语句打开文件时的文件号;“Spc(n)”表示插入 n 个空格;“Tab(n)”表示写入表达式内容时定位于第 n 列(从对象界面最左端第 1 列开始计算的第 n 列);“输出表达式列表”是要写入到文件中的数据,可以是数值或字符串表达式。

(2) Write # 文件号, [表输出表达式列表]

在该语法格式中,无论输出表达式列表中的分隔符是什么,写入文件的数据在任意两个数据项之间会自动插入一个逗号(,);使用该语句写入数据时,如果输出表达式列表中的表达式为字符串类型,则需要给字符串数据两端加上双引号(")。使用该语句写入数据时,如果输出表达式列表中的表达式为日期、时间或逻辑类型,则会自动给数据两端加上“#”号。



**提示**

使用“Input # 文件号, 变量列表”语句读取的是以 Write# 命令写入数据的文件,因为以 Write# 命令写入数据时,已经将各个数据项以“,”正确地区分。变量列表中各变量的数据类型必须和文件中要读取的数据的类型相一致,才能正确地获取文件中的数据,因此要注意读取文件中数据存放的格式。

## 20.4.4 顺序文件的关闭


当我们对文件的读、写或者其他操作结束以后,要将文件关闭。因为只有在关闭文件时,文件系统才能将内存缓冲区中的数据全部写入到文件中,所以不关闭文件将会造成文件数据的丢失。另外,及时关闭不再使用的文件也可避免无谓地计算机资源占用。关闭顺序文件的语法如下。

Close [[#] 文件号] [, [#] 文件号]

其中,“文件号”是我们要关闭的文件的文件号,文件号可以是一个文件号,也可以是多个文件号的文件号列表。文件号之间以逗号分隔,当文件号被省略时,所有 Open 语句打开的数据文件将全部关闭,并释放被关闭文件对应缓冲区所占的内存空间。文件关闭后,被关闭的文件的文件号将不再代表该文件。

## 20.4.5 顺序文件使用实例

**【范例 20-3】**在窗体中添加两个按钮,其中一个用来将输入的数据以顺序文件读写形式写入“花名册.txt”文件中,同时将“花名册.txt”文件中的数据分别读取到窗体中不同的文本框中显示。

(1) 启动 Visual Basic 6.0,在弹出的【新建工程】对话框中选择【标准 EXE】图标 ,然后单击【打开】按钮。

(2) 在 Form1 窗体中添加两个命令按钮(CommandButton)控件、1 个包含 4 个文本框的文本框控件组(TextBox)和 1 个包含 4 个标签的标签控件组(Label),然后按照下表设置控件的属性。

控件名称	名称	Caption/Index	控件作用
Command1	Command1	保存	用于向顺序文件“花名册”中添加记录
Command2	Command2	显示	用于读取顺序文件“花名册”中的记录
Label1 ( 0 )	Label1	姓名	提示记录中字段内容的意义
Label1 ( 1 )	Label1	班级	提示记录中字段内容的意义
Label1 ( 2 )	Label1	学号	提示记录中字段内容的意义
Label1 ( 3 )	Label1	专业	提示记录中字段内容的意义
Text1 ( 0 )	Text1	0	显示记录中字段的内容
Text1 ( 1 )	Text1	1	显示记录中字段的内容
Text1 ( 2 )	Text1	2	显示记录中字段的内容
Text1 ( 3 )	Text1	3	显示记录中字段的内容

(3) 将上述控件在 Form1 窗体上按照下图所示排列。



(4) 在 Form1 窗体上用右键单击，在弹出的快捷菜单中选择【查看代码】菜单项，进入代码窗口，输入以下代码。

```
Private k As Integer ' 设置一个记录号变量
```

(5) 在 Form1 窗体上双击【保存】按钮，进入代码窗口，输入以下代码（代码 20-3-1.txt）。

```
01 Private Sub Command1_Click()
02   Dim i As Integer ' 定义一个整型变量，用于循环计数器
03   Dim ans As String ' 定义一个字符串变量，用于获取消息对话框返回值
04   For i = 0 To 3 ' 检验信息的完整性
05     If Text1(i).Text = "" Then ' 判断各文本框中的值是否为空
06       ans = MsgBox("请填写完整的花名册信息！", vbOKOnly, "警告")
07     If ans = 1 Then Exit Sub
```

```

08 End If
09 Next i
10 If k > 0 Then '判断文件是否已经打开,正在读取
11     MsgBox "正在读取文件!当另一按钮为显示时,读取文件结束。", vbOKOnly, "警告"
12 Exit Sub
13 End If
14 Open App.Path & "\花名册.txt" For Append As #1
15     '使用 Append 模式,用于向顺序文件中添加记录
16 For i = 0 To 3 '保存花名册信息
17     Print #1, Text1(i).Text '写入数据
18 Next i
19 Close 1
20 For i = 0 To 3 '保存后清除文本框中的内容
21     Text1(i).Text = ""
22 Next i
23 End Sub

```

(6) 在 Form1 窗体上双击【显示】按钮,进入代码窗口,输入以下代码(代码 20-3-2.txt)。

```

01 Private Sub Command2_Click()
02     Dim i As Integer '定义一个整型变量,用于循环计数器
03     Dim str As String '定义一个整型变量,用于存储读取的数据
04     k = k + 1 '读取的记录条数加 1
05     If Command2.Caption = "重新显示" Then
06         Command2.Caption = "显示"
07         k = 0 '清空读取数据的记录数
08     Exit Sub
09 End If
10 If k = 1 Then
11     Command2.Caption = "下一条"
12     Open App.Path & "\花名册.txt" For Input As #1 '打开文件
13     For i = 0 To 3 '开始读取花名册信息
14         Line Input #1, str '整行读取
15         Text1(i).Text = str '将读取的数据赋给文本框显示
16     Next i
17 ElseIf EOF(1) = False Then '判断是否到记录结尾
18     For i = 0 To 3 '读取每一条花名册信息
19         Line Input #1, str '整行读取
20         Text1(i).Text = str '用文本框显示数据记录
21     Next i
22 Else
23     For i = 0 To 3 '读取完花名册后,清除记录显示
24         Text1(i).Text = "" '清空各文本框的值

```

```

25     Next i
26     Cls '清除记录数显示
27     Command2.Caption = "重新显示"
28     Close 1
29     Exit Sub
30 End If
31 Print "第"; k; "条记录" '在窗体中输入数据记录的条数
32 End Sub

```

## 【运行结果】

(1) 保存程序，按【F5】快捷键运行程序。单击【显示】按钮，可查看“花名册.txt”文件中的数据记录。

(2) 当【下一条】按钮变为【显示】按钮时，可以在文本框中输入数据，并单击【保存】按钮，向“花名册.txt”文件中添加数据。



### 提示

如果“花名册.txt”文件为只读属性，则不能进行记录的保存。

(3) 单击【显示】按钮，可以查看新添加的记录。

## 20.5 随机文件



本节视频教学录像: 7 分钟

随机文件是以随机的方式进行存取的文件。随机文件和数据库的结构比较类似, 它是以固定长度的记录为单位进行存储的, 每条记录中所包含的字段数和数据类型都是相同的。

# 记录 1	# 记录 2	# 记录 3	.....	# 记录 N
--------	--------	--------	-------	--------

从表中可以看到, 当我们从随机文件中搜索某一内容的时候, 并不是将所有数据一一和搜索内容匹配, 而是从随机文件的记录中搜索, 搜索到符合条件的记录后, 根据记录中所对应的文件数据位置直接找到文件。随机文件的查找过程就像在图书馆找书一样, 通常会使用图书馆中的检索计算机检索馆藏书目, 从书目中找到所需的图书后, 将该书的索取号告诉图书管理员, 图书管理员根据索取号直接找到该书交给读者。顺序文件的查找过程就像我们在图书馆的书架中找书一样, 从第 1 个书架开始, 一本一本寻找, 直到找到为止。如果所要的书恰巧是在最后几个书架中, 无疑就会浪费大量的时间。

随机文件和顺序文件一样, 都有自身的优缺点。随机文件的优点是, 可以按任意顺序访问文件中的数据, 可以方便地修改各个记录, 而不需要重写全部记录; 它的缺点是, 由于在每个记录前增加了记录号, 从而使其占用的存储空间增大, 同时也增加了编程的工作量。

### 20.5.1 随机文件的打开

在打开随机文件之前, 首先要在模块中用 Type 定义记录中所含变量各个字段的类型和长度, 例如:

```
Type Record
Name As String *8
Age As Integer
Sex As Boolean
End Type
```

在定义好记录中所含变量各个字段的类型和长度之后, 就可以打开该文件了, 语法格式如下。

```
Open 文件名 For Random As # 文件号 [Len= 记录长度]
```

在这个语句中, 文件的读取、存储都以这一模式打开, 对随机文件进行某种操作后不必关闭该文件就可以进行另一种操作。在 Open 语句中要给出记录的长度, 如果省略, 则默认为 128 个字节。记录长度是一条记录实际所占的字节数, 可以使用 Len() 函数获取。

### 20.5.2 随机文件的读取

从随机文件中读取数据可以使用 Get 语句, 语法格式如下。

```
Get [#] 文件号, [记录号], 记录变量
```

在这个语句中, 记录变量的数据类型必须同文件中记录的数据类型一致。随机文件的存取操作与顺序文件存取的不同之处在于: 打开一个随机文件之后, 在关闭随机文件之前, 既可读取也可存储。

### 20.5.3 随机文件的写入

向随机文件中写入数据可以使用 Put 语句，语法格式如下。

Put [# 文件号], [记录号], 记录变量

在这个语句中，“记录号”是大于 1 的整数，可以是常量、变量或表达式，表示写入的是第几条记录。若随机文件中没有指定的记录号，则写入文件中；如果已存在指定记录号的记录，则刷新该记录；如果不指定记录号，则表示将变量内容写入下一个记录位置。

### 20.5.4 随机文件的关闭

随机文件的关闭和顺序文件的关闭相似，语法如下。


Close [# 文件号 1], [# 文件号 2]

如果 Close 语句的后面无文件号，则关闭所有打开的文件。

### 20.5.5 随机文件使用实例

本小节使用随机文件制作一个简易班级花名册。

#### 【范例 20-4】读取花名册。

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

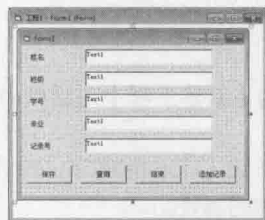
(2) 在 Form1 窗体中添加 4 个命令按钮 (CommandButton) 控件和 1 个含有 5 个标签的标签控件组 (Label)，然后按照下表设置这些控件的属性。

控件名称	名称	Caption	控件作用
Command1	Command1	保存	用于向随机文件“花名册”中添加记录
Command2	Command2	查询	用于读取随机文件“花名册”中的记录
Command3	Command3	结束	用于关闭随机文件
Command4	Command4	添加记录	用于在随机文件末尾添加记录
Label1 ( 0 )	Label1	姓名	提示记录中字段内容的意义
Label1 ( 1 )	Label1	班级	提示记录中字段内容的意义
Label1 ( 2 )	Label1	学号	提示记录中字段内容的意义
Label1 ( 3 )	Label1	专业	提示记录中字段内容的意义
Label1 ( 4 )	Label1	记录号	提示记录中字段内容的意义



(3) 在 From1 窗体上添加 1 个含有 5 个文本框的文本框控件组 ( TextBox ), 然后按照下表设置这些控件的属性。

控件名称	名称	Index	控件作用
Text1 ( 0 )	Text1	0	显示记录中字段的名称
Text1 ( 1 )	Text1	1	显示记录中字段的名称
Text1 ( 2 )	Text1	2	显示记录中字段的名称
Text1 ( 3 )	Text1	3	显示记录中字段的名称
Text1 ( 4 )	Text1	4	显示记录中字段的名称



(4) 在 Form1 窗体上用右键单击, 在弹出的快捷菜单中选择【查看代码】菜单项, 进入代码窗口, 输入以下代码 ( 代码 20-4-1.txt )。

```
01 Private Type record      ' 自定义数据类型
02     name As String * 8
03     add As String * 34
04     tel As String * 12
05     post As String * 34
06 End Type
07 Dim j, k As Integer      ' 变量 j 用于记录个数, 变量 k 用于记录一个记录号
08 Dim txl As record        ' txl 用于记录一个数组名
```

(5) 在 Form1 窗体的空白处双击, 进入代码窗口, 输入以下代码。

```
01 Private Sub Form_Load()
02     k = 0      ' 记录号初始为零
03     Open App.Path & "/ 花名册 .txt" For Random As #1 Len = 88      ' 打开文件
```



```
04 j = LOF(1) / 88 '统计文件记录条数
05 End Sub
```

(6) 在 Form1 窗体上双击【保存】按钮，进入代码窗口，输入以下代码（代码 20-4-2.txt）。

```
01 Private Sub Command1_Click()
02   Dim ans As String '变量 ans 用于接收消息对话框的返回值
03   Dim i As Integer '变量 i 用于循环计数器
04   For i = 0 To 4 '检验信息的完整性
05     If Text1(i).Text = "" Then '判断文本框内容是否为空
06       ans = MsgBox("请填写完整的花名册信息及记录号!", vbOKOnly, "警告")
07       If ans = 1 Then Exit Sub
08     End If
09   Next i
10   j = LOF(1) / 88 '统计文件中数据记录条数
11   k = Val(Text1(4).Text) '获取文件记录号
12   If k <= j Then '如果当前记录号小于总的数据记录条数则刷新记录
13     ans = MsgBox("是否确认刷新花名册中的记录?", vbYesNo, "警告")
14     If ans = vbNo Then MsgBox "没有刷新记录, 请重新输入记录号!", vbOKOnly, "提示": Exit
Sub
15   End If
16   txl.name = Text1(0).Text '保存姓名
17   txl.add = Text1(1).Text '保存地址
18   txl.tel = Text1(2).Text '保存电话
19   txl.post = Text1(3).Text '保存邮编
20   Put #1, k, txl '写入数据记录
21   For i = 0 To 4
22     Text1(i).Text = "" '保存后清除文本框中的内容
23   Next i
24 End Sub
```

(7) 在 Form1 窗体上双击【查询】按钮，进入代码窗口，输入以下代码（代码 20-4-3.txt）。

```
01 Private Sub Command2_Click()
02   j = LOF(1) / 88 '统计文件中数据记录条数
03   k = Val(Text1(4).Text) '获取文件记录号
04   If k > j Or k = 0 Then MsgBox "超出记录范围!", vbOKOnly, "警告": Exit Sub
05   Get #1, k, txl '读出数据记录
06   Text1(0).Text = txl.name '将记录中的姓名赋给文本框显示
07   Text1(1).Text = txl.add '将记录中的地址赋给文本框显示
08   Text1(2).Text = txl.tel '将记录中的电话赋给文本框显示
09   Text1(3).Text = txl.post '将记录中的邮编赋给文本框显示
10 End Sub
```

(8) 在 Form1 窗体上双击【结束】按钮，进入代码窗口，输入以下代码。

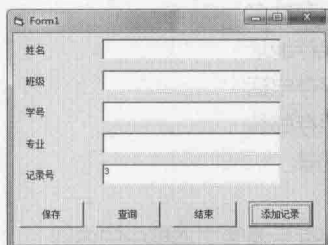
```
01 Private Sub Command3_Click()  
02     Close #1      ' 关闭文件  
03     End           ' 退出系统  
04 End Sub
```

(9) 在 Form1 窗体上双击【添加记录】按钮，进入代码窗口，输入以下代码 (代码 20-4-4.txt)。

```
01 Private Sub Command4_Click()  
02     j = LOF(1) / 88 ' 统计文件中数据记录条数  
03     j = j + 1 ' 记录总数加 1  
04     For i = 0 To 3  
05         Text1(i).Text = "" ' 清空各文本框  
06     Next i  
07     Text1(4).Text = j ' 显示当前记录号  
08 End Sub
```

## 【运行结果】

(1) 保存程序，按【F5】快捷键运行程序。单击【添加记录】按钮，系统会自动产生一个记录号。



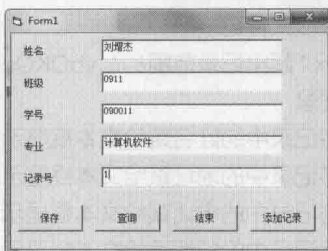
The screenshot shows a Windows form titled 'Form1'. It contains five text boxes labeled '姓名' (Name), '班级' (Class), '学号' (ID), '专业' (Major), and '记录号' (Record Number). Below the text boxes are four buttons: '保存' (Save), '查询' (Query), '结束' (End), and '添加记录' (Add Record). The '记录号' box contains the value '0'.



### 提示

程序运行后，如果没有“花名册.txt”文件，系统会默认在工程所在的文件夹创建一个没有任何数据的“花名册.txt”。

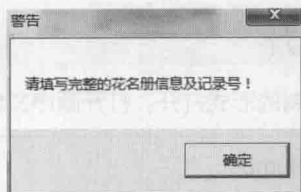
(2) 输入“姓名”、“班级”、“学号”和“专业”等数据，然后单击【保存】按钮，可以保存数据。



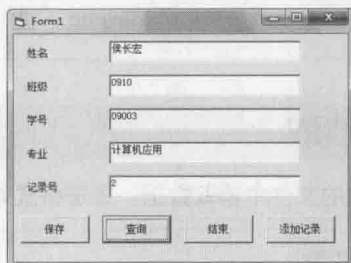
The screenshot shows the same 'Form1' window as before, but now the text boxes are filled with data: '姓名' (Name) is '刘耀杰', '班级' (Class) is '0911', '学号' (ID) is '090011', and '专业' (Major) is '计算机软件'. The '记录号' (Record Number) box still contains '0'. The buttons remain the same.



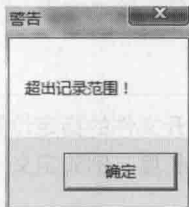
如果没有输入“姓名”、“班级”、“学号”和“专业”等数据，或数据输入的不完整，则会弹出【警告】对话框，提示用户填写完整的花名册信息及记录号。



(3) 输入相关【记录号】，单击【查询】按钮，可以查询相关的数据记录。



如果没有输入【记录号】而单击【查询】按钮，则会弹出【警告】对话框，提醒所要查询的数据超出了记录范围。



## 20.6 二进制文件



本节视频教学录像：4 分钟

二进制文件中的数据均以二进制方式存储，存储单位是字节。我们知道，计算机从根本上来说是以二进制方式进行运算的，所以在二进制文件中，能够存取任意需要的字节，可以把文件指针移到文件的任何地方。还以图书馆的例子来说，可以认为二进制的存储方式就像是把所有的书都打碎成了一块块碎片集中存放，当我们使用这些书的时候，再把这些碎片组合成书。目前，这在现实生活中是不可能的，但是在计算机中却是一种非常常见的存储方式。二进制文件的优点是：存储密集，空间利用效率高，存取方式最为灵活。它的缺点是：操作起来比较困难，工作量较大。

**提示**

ASCII 码文件是指在文件中存储的是每个字符的 ASCII 码, 这些文件被称为文本文件, 可以利用文本编辑工具对其进行编辑和查看, 例如源程序代码文件。二进制读写既可以读写二进制文件, 也可以读写 ASCII 码文件。

## 20.6.1 二进制文件的打开

任何一种类型的文件都可以以二进制的形式打开, 打开顺序文件时使用的是 Open 语句。语法如下。

```
Open pathname For Binary As FileNumber
```

例如, 以二进制形式打开 D 盘根目录下的“VB.txt”文件, 使用下面的语句即可。

```
Open "D:\VB.txt" For Binary As #1 '以二进制形式打开文件
```

## 20.6.2 二进制文件的读取

二进制文件使用 Get 语句从指定的文件中读取数据。语法格式如下。

```
Get [#] 文件号, [记录号], 记录变量
```

由于二进制文件的读取操作与随机文件的读取操作基本相似, 这里不再赘述。

## 20.6.3 二进制文件的写入

二进制文件的写入使用的是 Put 语句。其语法格式如下。

```
Put [#]filenumber,[recnumber],varname
```


使用 Put 语句将变量的内容写入到所打开文件的指定位置, 而当文件刚打开时, 指向的是文件中的第 1 个位置。它一次写入的长度等于变量的长度。例如定义一个整型变量 i, 当向文件中写入 i 时, 就需要写入两个字节的数。

## 20.6.4 二进制文件的关闭

二进制文件的关闭与顺序文件和随机文件的关闭方法相同, 都是利用 Close #filenumber 语法格式来关闭文件的。

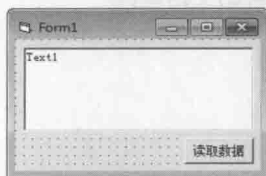
## 20.6.5 二进制文件使用实例

### 【范例 20-5】以二进制形式实现数据的读取。

(1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单

击【打开】按钮。

(2) 在 Form1 窗体中添加 1 个文本框 (TextBox) 控件和一个命令按钮 (CommandButton) 控件, 设置命令按钮控件的【Caption】属性为“读取数据”, 设置文本框控件的【MultiLine】属性为“True”, 然后将各控件按下图进行放置。

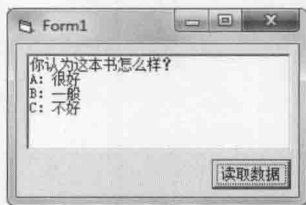


(3) 在 Form1 窗体上双击【读取数据】按钮, 进入代码窗口, 输入以下代码 (代码 20-5.txt)。

```
01 Private Sub Command1_Click()  
02   Dim str As String      ' 字符串变量 str 用于获取文件路径  
03   str = App.Path & "/ 答卷 .txt"    ' 获取文件路径  
04   Open str For Binary As #1        ' 打开文件  
05   Get #1, , str      ' 读出数据记录  
06   Text1.Text = str ' 显示数据  
07   Close #1          ' 关闭文件  
08 End Sub
```

## 【运行结果】

保存程序, 按【F5】快捷键运行程序。单击【读取数据】按钮, 即可将“答卷.txt”文件中的内容以二进制的形式读取出来。



## 20.7 高手点拨



本节视频教学录像: 2 分钟

在实际应用中, 经常需要对文件系统上的驱动器、目录和文件进行处理, 如收集驱动器的相关信息, 创建、添加、移动或删除目录和文件等。实现这些操作需要 Visual Basic 6.0 提供的文件对象模型 (FSO, 即 FileSystemObject) 来对文件系统进行访问处理。该模型提供了一个基于对象的工具, 通过其所提供的一系列属性和方法, 可以在程序中更简单、更灵活地对文件系统进行各种操作。FSO 包含了许多其他的对象。如:

1. Derive 对象: 允许收集系统物理或通过 LAN 与系统逻辑连接的硬盘、CD-ROM 等驱动器的可

用空间和共享名等信息。

2. Folder 对象：允许创建、删除或移动目录，并向系统查询目录的命名称、路径等。

3. Files 对象：允许创建、删除或移动文件，并向系统查询文件的名称、路径等。

4. TextSystemObjec 对象：允许创建和读写文本文件。

另外 FSO 还有一系列方法可以通过帮助文档查询。

## 20.8 实战练习

### 一、思考题

读下面两段程序，分析用 write# 语句和 print# 语句建立顺序文件时的格式有什么本质区别？

程序 A：

```
Private Sub Command1_Click()
    Dim a As Integer
    Open "C:\b.txt" For Output As 1
    For a = 1 To 10
        Print #1, Trim(Str(a));
    Next
    Close
End Sub
```

程序 B：

```
Private Sub Command1_Click()
    Dim a As Integer
    Open "C:\b.txt" For Output As 1
    For a = 1 To 10
        Write #1, Trim(Str(a));
    Next
    Close
End Sub
```

### 二、操作题

使用 LOF 函数获得“D:\ 跟我上机 \ch20”文件夹下的“VB.txt”文件的长度，并将其显示在窗体上。

# 第21章



本章视频教学录像：10 分钟

## 让你的程序去旅行——应用程序打包

在 Visual Basic 6.0 开发环境下设计的程序，特别是大的应用程序，包含有许多与应用程序相关的文件，在不具备 Visual Basic 6.0 开发环境的计算机中是无法运行的。

为了解决这一问题，Visual Basic 6.0 提供有应用程序打包工具，可以将与应用程序有关的文件全部集中起来，生成一个安装包文件。用户通过安装该文件就能在别的计算机中正常运行程序了。本章介绍打包前的准备、打包应用程序的步骤、安装卸载应用程序的方法以及打包应注意的问题。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解打包前的准备
- ☐ 掌握打包应用程序的步骤
- ☐ 熟悉安装应用程序的方法
- ☐ 熟悉卸载应用程序的方法
- ☐ 了解打包应注意的问题



## 21.1 打包前的准备



本节视频教学录像: 3 分钟

在使用 Visual Basic 设计完成一个应用程序后, 开发人员往往需要把它发布给其他人使用。如果要使程序脱离 Visual Basic 6.0 的集成环境而独立运行, 就必须对应用程序进行打包, 打包后可以利用磁盘、CD、网络等途径, 将使用 Visual Basic 创建的应用程序发布给任何人。

在准备发布前, 应仔细规范精简程序的代码, 一般来说, 精简程序要注意以下几个方面。

(1) 检查代码中是否存在无用的变量、过程、标识符和空行等元素。

(2) 检查注释是否清楚、完整, 方便精简程序后的调试。

(3) 检查代码中是否存在可精简语句。

(4) 检查是否存在无用或可整合的窗体和控件。

(5) 检查代码中变量、控件等使用完成后, 是否释放内存。

(6) 在打包前, 应该确保工程文件已经妥善编译和保存。如果正在使用一个工程组, 或者在 Visual Basic 6.0 中加载了多个工程, 那么在打包前应确保当前的工程就是要打包的工程。

(7) 调试程序, 保证程序功能完整。

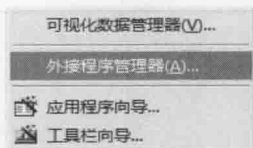
## 21.2 打包应用程序



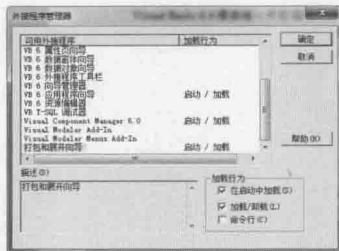
本节视频教学录像: 2 分钟

Visual Basic 6.0 提供有一个打包向导供打包。在使用打包向导前, 首先要把它加载到 Visual Basic 6.0 的工作环境中。加载打包向导的具体步骤如下。

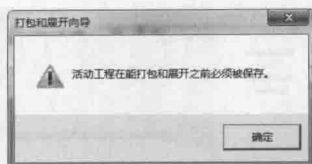
(1) 打开 Visual Basic 6.0, 选择【外接程序】▶【外接程序管理器】菜单命令。



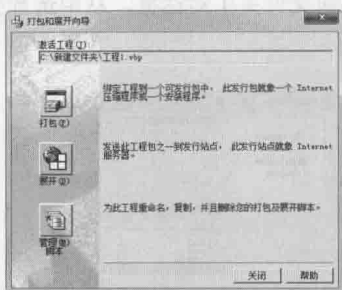
(2) 弹出【外接程序管理器】对话框, 在【可用外接程序】列表框中选中“打包和展开向导”选项, 选中【在启动中加载】和【加载/卸载】两个复选框, 然后单击【确定】按钮。



(3) 这样, 打包向导就加载到 Visual Basic 6.0 中了。但此时还不能打开打包向导, 只有工程被保存后才能打开打包向导。否则, 打开时会提示工程必须先进行保存。



(4) 工程保存后选择【外接程序】>【打包和展开向导】菜单命令，即可打开【打包和展开向导】对话框。

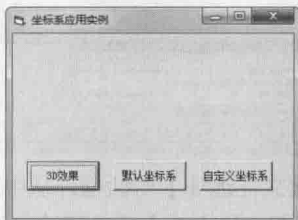


下面通过一个实例来讲解如何进行应用程序的打包。

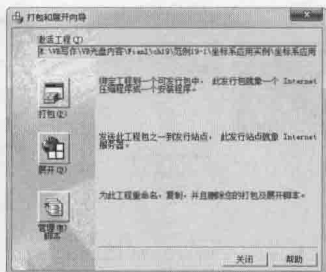
### 【范例 21-1】使用打包向导，打包光盘“Final\ch19\范例 19-1\坐标系应用实例\坐标系应用实例.vbp”工程文件。

第 1 步：打开打包向导


(1) 打开随书光盘中的“坐标系应用实例.vbp”工程文件，程序调试通过后保存工程。

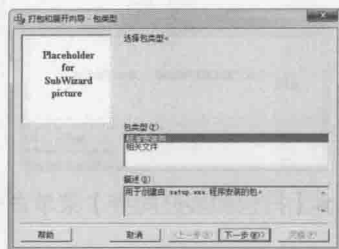


(2) 选择【外接程序】>【打包和展开向导】菜单命令，打开【打包和展开向导】对话框。

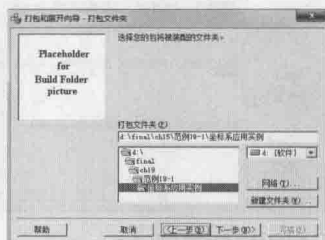


第 2 步：选择打包设置

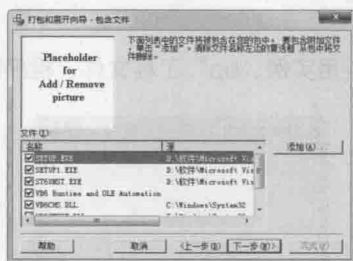
(1) 单击【打包】按钮 ，弹出【打包和展开向导 - 包类型】对话框，在【包类型】列表框中选择“标准安装包”选项，然后单击【下一步】按钮。



(2) 弹出【打包和展开向导 - 打包文件夹】对话框，从中选择要存放打包文件的文件夹，然后单击【下一步】按钮。



(3) 在弹出的【打包和展开向导 - 包含文件】对话框中，打包向导会自动选择工程的应用程序、应用的控件和 DLL 等文件。如果有缺失文件，则可单击【添加】按钮将文件添加到列表中。检查确认没有缺失后，单击【下一步】按钮。



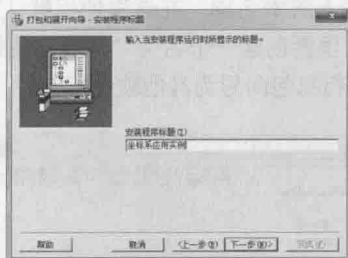
(4) 弹出【打包和展开向导 - 压缩文件选项】对话框，从中选择打包类型。根据不同的实际情况，在【压缩文件选项】中选中【单个的压缩文件】或【多个压缩文件】单选按钮，这里选择【单个的压缩文件】单选按钮，然后单击【下一步】按钮。



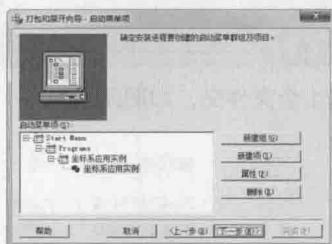
#### 提示

多个压缩文件指安装完成后产生多个占用空间较小的安装包文件，当用户使用软盘存储该安装包文件时，选择该项。

(5) 弹出【打包和展开向导 - 安装程序标题】对话框，【安装程序标题】文本框中的文字将显示在安装程序的背景中。在这里输入“坐标系应用实例”，然后单击【下一步】按钮。



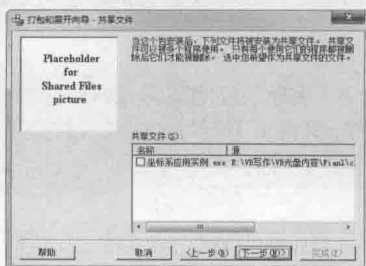
(6) 弹出【打包和展开向导 - 启动菜单项】对话框，在这里可以设置在【开始】菜单中显示哪些项目，默认情况下只有“坐标系应用实例”一项，然后单击【下一步】按钮。



(7) 弹出【打包和展开向导 - 安装位置】对话框，在这里可以更改文件夹的安装位置，例如要安装到 ProgramFiles 下的 Package 文件夹里，则在安装位置框里输入 \$(ProgramFiles)\Package，在这里我们不再重新指定安装位置，单击【下一步】按钮。

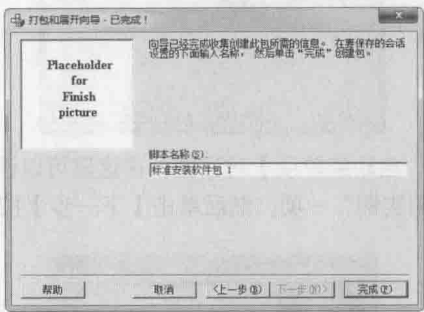


(8) 弹出【打包和展开向导 - 共享文件】对话框，如果文件需要被多个应用程序使用，则勾选上要共享的文件。例如，有多个应用程序都使用了同一个 ActiveX 控件，那么要将它设置为共享文件，在这里可以设置为共享文件的是“坐标系应用实例.exe”，只是一个一般应用程序，不需要多个程序共享使用，所以直接单击【下一步】按钮。



第3步：打包应用程序

(1) 打包向导此时已经收集完成打包所需要的所有信息，在弹出的【打包和展开向导 - 已完成！】对话框中的【脚本名称】文本框中输入脚本名称，在这里使用默认的脚本名称，然后直接单击【完成】按钮，即可完成应用程序的打包。这里会创建一个名为“标准安装软件包 1”的脚本，该脚本记住了本次打包过程中所做的选择，在以后运行打包向导为其他软件打包时，使用该脚本可以避免重复的选择。



(2) 打包完成，会生成 3 个文件和 1 个文件夹，如图所示。

名称	修改日期	类型	大小
Support	2014/1/2 14:35	文件夹	
setup	2004/3/10 0:00	应用程序	137 KB
SETUP.LST	2010/1/22 4:32	LST 文件	4 KB
坐标系应用实例	2010/1/22 4:32	360压缩 CAB 文件	1,325 KB

文件及文件夹的信息说明如表所示。

文件 / 文件夹	信息说明
Support 文件夹	压缩包中包含的所有文件
坐标系应用实例	数据文件文件包，安装所需的文件全部在包里
setup	安装程序的主文件
SETUP.LST	安装信息文件

在安装过程中包含两个安装程序——setup.exe 和 setup1.exe。setup.exe 程序在用户计算机上执行预安装处理，包括安装 setup1.exe 程序以及运行主安装程序所需的任何其它文件。Setup1.exe 是可以自定义的，位于 Visual Basic 主目录的 \Wizards\PDWizard\Setup1 子目录中。关于 setup.exe 和 setup1.exe 区别的考察在一些考试中经常出现。

Support 文件夹中有一个“坐标系应用实例.BAT”批处理文件，当工程重新改动后，将工程重新编译，然后将新的执行文件拷贝到 Support 文件夹下，执行这个批处理文件，就可以重新打包，不需要每次都运行打包向导了。



**提示**

打开打包向导除了上面介绍的方法，选择【开始】>【所有程序】>【Microsoft Visual Basic 6.0 中文版】>【Microsoft Visual Basic 6.0 中文版工具】>【Package & Deployment 向导】菜单命令，也可以打开打包向导。

## 21.3 安装应用程序



本节视频教学录像：1 分钟

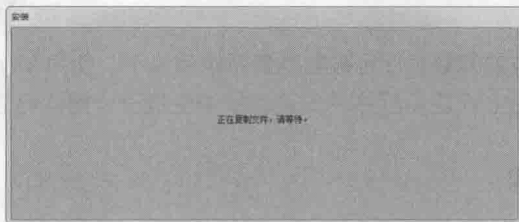
在使用打包向导完成了应用程序的打包并产生了安装程序后，应该对安装程序进行测试，也就是进行实际安装。在测试过程中，应确保在一台没有 Visual Basic 集成环境以及应用程序所需的控件的机器上测试。

下面以安装打包后的“坐标系应用实例”为例，讲解如何安装应用程序。

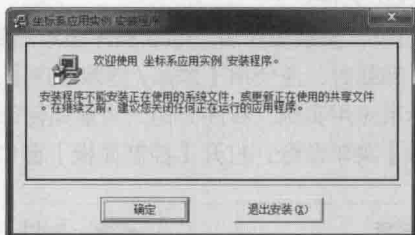
(1) 双击打包后生成的“setup”安装文件。

名称	修改日期	类型	大小
Support	2014/1/2 14:35	文件夹	
setup	2004/3/10 0:00	应用程序	137 KB
SETUP.LST	2010/1/22 4:32	LST 文件	4 KB
坐标系应用实例	2010/1/22 4:32	360压缩 CAB 文件	1,325 KB

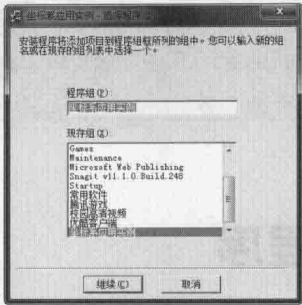
(2) 弹出【安装】窗口，提示“正在复制文件，请等待。”



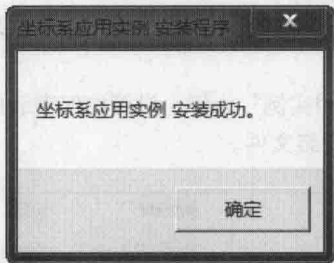
(3) 随即弹出【坐标系应用实例 安装程序】对话框，单击【更改目录】按钮，可以不使用默认的目录，自定义安装的目标目录。这里使用默认目录，单击 按钮。



(4) 弹出【坐标系应用实例 - 选择程序组】对话框，在这里可以设定程序安装时在【开始】菜单中创建的程序组的名称。设定完成单击【继续】按钮。



(5) 程序安装完成。



## 21.4 卸载应用程序



本节视频教学录像：1 分钟

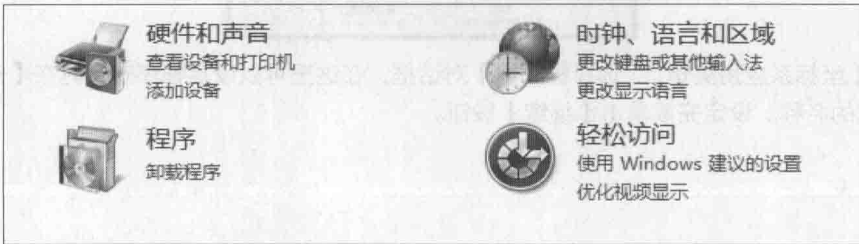
安装完应用程序后，相应的卸载程序也被复制到系统目录下。使用 Visual Basic 打包向导创建的安  
装程序，在完成安装工作之后会在应用程序的安装目录中生成一个删除日志文件 (st6unst.log)，该日志  
文件包含以下一些信息。

- (1) 安装过程中创建的目录。
- (2) 安装的文件及位置。
- (3) 创建或修改的注册表项。
- (4) 安装程序创建的【开始】菜单项目。
- (5) 程序自注册的 dll、exe 或 ocx 文件。

例如，在 Windows 7 操作系统下，安装程序后，卸载程序被添加到【控制面板】中的【添加 / 删除  
程序】已安装应用程序的列表中。卸载时，应使用【添加 / 删除程序】来卸载应用程序。

下面以卸载安装完成的“坐标系应用实例”程序为例，讲解如何卸载应用程序。

- (1) 选择【开始】>【控制面板】菜单命令，打开【控制面板】窗口，从中双击【卸载程序】图标。

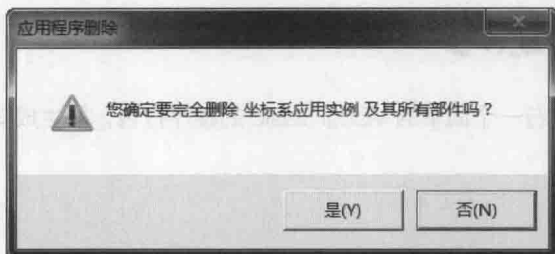




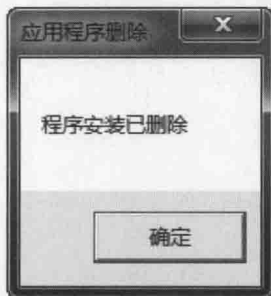
(2) 在弹出的【程序和功能】窗口的【卸载或更改程序】列表中选中“坐标系应用实例”选项。



(3) 单击【卸载/更改】按钮，弹出【坐标系应用实例 删除】窗口，在弹出的【应用程序删除】对话框中单击【是】按钮。



(4) 卸载程序完成删除“坐标系应用实例”程序，并弹出对话框报告“程序安装已删除”。



## 21.5 打包应注意的问题



本节视频教学录像：2 分钟

一个程序设计完成后，需要进行打包才能发布给别人。但是可能花费很多时间和精力设计出来的程序，在打包后由于自己的失误，出现了很多问题。下面简要介绍程序打包时应注意的问题。

- (1) 程序包含有第三方控件时，打包过程中一定要把 OCX 文件包含进去。
- (2) 对于工程中引用的文件，应查看是否含有绝对路径，如果有则改为相对路径。
- (3) 如果程序中包含数据库，则数据库的引用代码也要改为相对路径。
- (4) 打包前应通过 Visual Basic 的【工程】>【引用】和【部件】命令，将所有加载到工程中的文件复制出来。
- (5) 打包后应尽可能在各种操作系统中测试一下安装程序能否正常运行。

## 21.6 高手点拨



本节视频教学录像: 1 分钟

Support 目录中有一个“坐标系应用实例.bat”文件, 可以使用它帮助我们简化打包程序。当工程经过改动之后, 我们可以将工程重新编译一下, 然后直接将可执行文件复制到此 support 目录下, 执行这个批处理文件, 就可以重新打包。这样每次改动后就不再需要重新运行一次打包向导, 从而简化了重新打包的过程。

## 21.7 实战练习

依照【范例 21-1】, 进行一个简单的 Visual Basic 的程序打包, 并生成安装文件。

# 第4篇

## 应用开发

本篇包括项目规划、使用 Visual Basic 实现远程控制、仿 Windows 画图程序、开发自己的播放器、文件分割与合并程序以及连连看等各种实用程序的开发。学习完本篇，读者应能开发实用的高级应用程序。

第 22 章 项目实战前的忠告——项目规划

第 23 章 网络通信应用开发——VB 实现远程控制

第 24 章 图形图像应用开发——仿 Windows 画图程序

第 25 章 多媒体应用开发——开发自己的播放器

第 26 章 文件系统应用开发——文件分割与合并程序

第 27 章 游戏开发——VB 连连看

# 第22章



本章视频教学录像：28 分钟

## 项目实战前的忠告——项目规划

一个项目系统从无到有要经历策划、分析、开发、测试和维护等阶段，我们将这样的一个阶段过程称为项目的生命周期。本章介绍面对项目的开发，如何对项目进行规划。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解项目开发流程
- ☐ 掌握客户需求有哪些
- ☐ 熟悉项目开发团队的组成
- ☐ 了解项目计划说明书
- ☐ 了解项目开发阶段运作过程

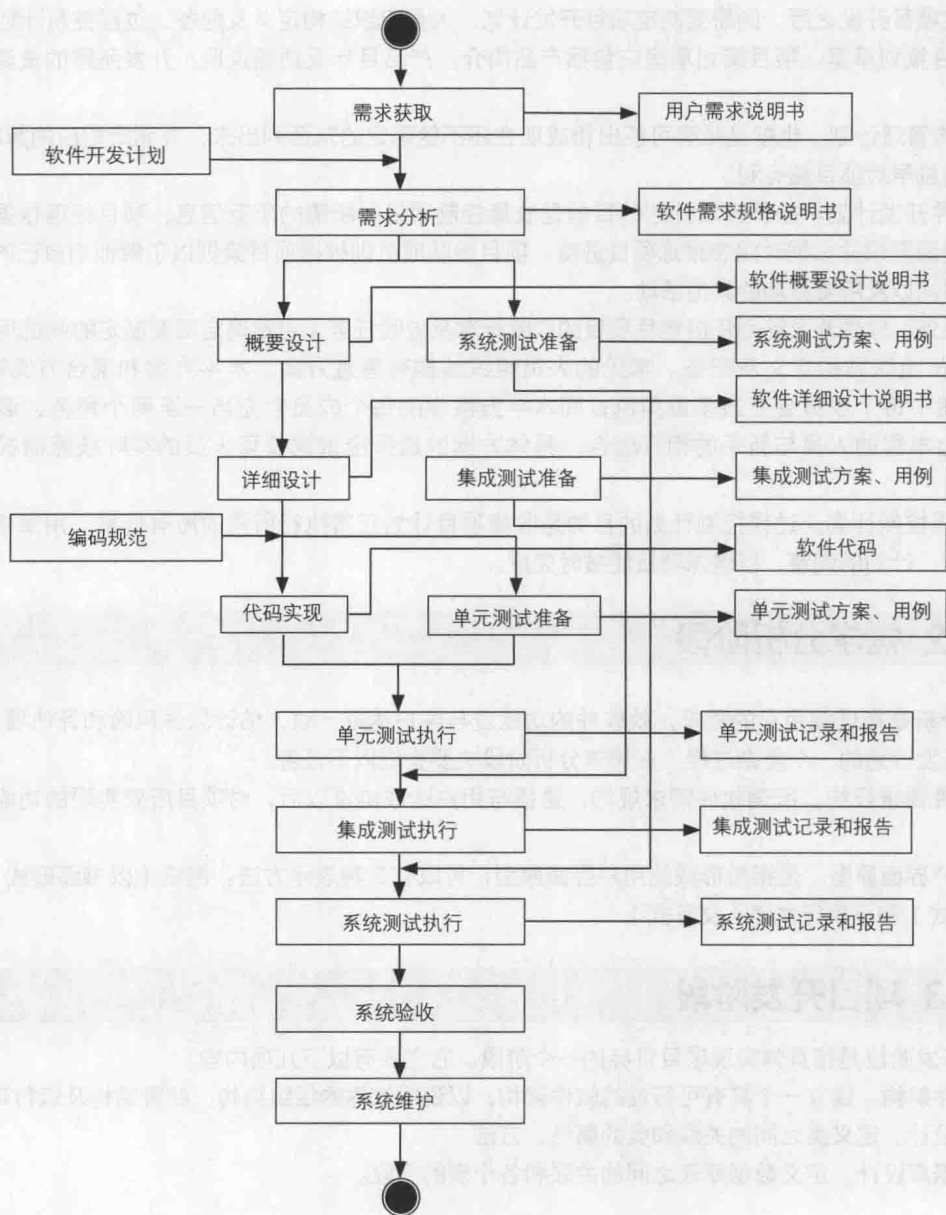
## 22.1 项目开发流程



本节视频教学录像：6 分钟

每一个项目的开发都不是一帆风顺的。为了避免软件开发过程中的混乱，也为了提高软件的质量，需要按照项目开发的流程操作。一个项目的开发往往会被分成很多步骤来实现，每一个步骤都有自己的起点和终点。

项目开发的流程如图所示。



下面从项目整体划分介绍项目开发过程中各阶段的主要任务。

### 22.1.1 项目策划阶段

项目策划草案和风险管理策划往往作为一个项目开始的第1步。当接到一个项目后,应根据公司高层负责人所制定的初步商业计划书来完成项目的策划草案,并确定、分析项目的风险和项目风险的优先级,同时还要制定出一套面对项目风险的解决方案。项目策划阶段的主要目的是确立产品开发的经济理由。

当确定项目开发之后,则需要制定项目开发计划、人员组织结构定义及配备、过程控制计划等。

(1) 项目策划草案。项目策划草案应包括产品简介、产品目标及功能说明、开发所需的资源、开发时间等。

(2) 风险管理计划。也就是把有可能出错或现在还不能确定的东西列出来,并制定相应的解决方案。风险发现得越早对项目越有利。

(3) 软件开发计划。软件开发计划的目的是收集控制项目时所需的所有信息,项目经理根据项目策划来安排资源需求并根据时间表跟踪项目进度。项目团队成员则根据项目策划以了解他们自己的工作任、工作时间以及所要依赖的其他活动。

除此之外,软件开发计划还应包括项目的应收标准及应收任务(包括确定需要制定的测试用例)。

(4) 人员组织结构定义及配备。常见的人员组织结构有垂直方案、水平方案和混合方案等3种。在垂直方案中每个成员会充当多重角色;而水平方案中的每个成员会充当一至两个角色,混合方案则包括经验丰富的人员与新手的相互融合。具体方案的选择应根据公司人员的实际技能情况调整安排。

(5) 过程控制计划。过程控制计划的目的是收集项目计划正常执行所需的所有信息,用来指导项目进度的监控、计划的调整,以确保项目能按时完成。

### 22.1.2 需求分析阶段

需求分析是指理解用户的需求,就软件的功能应与客户达成一致,估计软件风险和评估项目代价,最终形成开发计划的一个复杂过程。在需求分析阶段主要完成以下任务。

(1) 软件需求规约。所谓软件需求规约,是指与用户达成协议以后,对项目所要实现的功能进行的详细说明。

(2) 用户界面原型。是指所形成的用户界面原型,可以有3种表示方法:图纸(以书面形式)、位图(以图片形式)和可执行文件(交互式)。

### 22.1.3 项目开发阶段

项目开发阶段是指具体实现项目目标的一个阶段。它主要有以下几项内容。

(1) 软件架构。建立一个具有可行性的软件架构,以说明软件的组织结构、部署结构及运行环境。

(2) 类设计。定义类之间的关联和类的属性、方法。

(3) 数据库设计。定义数据库表之间的关联和各个表的字段。

(4) 编码和单元测试。按照设计文档进行代码的编码，每完成一个模块都应进行相应的单元测试，以避免后期发现了错误不好处理。

(5) 集成系统。按软件组织结构的要求将各个子系统组合起来，以形成最终的项目。

## 22.1.4 项目测试阶段

项目测试的目的是在项目投入使用之前找出项目所存在的错误。它主要包括：根据设计文档，核实每个模块能否正常运行；并根据需求文档说明，核实需求是否被正确实施。

(1) 测试计划。收集和整理测试相关信息，为测试工作提供指导。

(2) 测试数据。在选择所测试的数据时，应尽量使用真实数据，以免在交付项目后不能正常使用。

(3) 测试报告。记录每一次进行项目测试的结果，详细描述项目中存在的问题，并提出相关的解决方案。

## 22.1.5 项目后期维护

项目维护是指在已完成对项目的研制（分析、设计、编码和测试）工作并交付使用以后，对项目产品所进行的一些项目工程的活动。即根据软件运行的情况，对软件进行适当的修改，以适应新的要求，以及纠正运行中发现的错误等。同时，还需要编写软件问题报告和软件修改报告。

# 22.2 满足客户需求



本节视频教学录像：4 分钟

满足客户的需求也就是在项目开发流程中所提到的需求分析。如果一个项目经过大量的人力、物力、财力和时间的投入后，所开发出的软件没人要，这种遭遇是很让人痛心疾首的。比如，用户需要一个人事管理的软件，而你在软件开发前期忽略了向用户询问软件中是否增加岗位培训的功能，而是想当然地认为岗位培训一般都用于现实生活中，没有哪家公司会用软件进行员工的岗位培训。然而，当你千辛万苦地将软件开发完成，并向用户提交时，才发现岗位培训功能在这个软件中对用户来说是多么的渴望与需要，这个时候是开发人员最为苦恼的时候。

需求分析之所以重要，就因为它具有决策性、方向性和策略性的作用，它在软件开发的过程中占据着举足轻重的地位。在一个大型软件系统的开发中，它的作用要远远大于程序设计。那么该如何做，才能满足客户的需求呢？

(1) 了解客户业务目标。只有在需求分析时更好地了解客户的业务目标，才能使产品更好地满足需求。充分了解客户业务目标，将有助于程序开发人员设计出真正满足客户需要并达到期望的优秀软件。

(2) 撰写高质量的需求分析报告。需求分析报告是分析人员对从客户那里获得的所有信息整理，主要用以区分业务需求及规范、功能需求、质量目标、解决方法和其他信息，它使程序开发人员和客户之间针对要开发的产品内容达成了共识和协议。需求分析报告应以一种客户认为易于翻阅和理解的方式组织编写。同时程序分析师可能会采用多种图表作为文字性需求分析报告的补充说明，虽然这些图表很容易让客户理解，但是客户可能对此并不熟悉，因此对需求分析报告中的图表进行详细的解释说明也是



很有必要的。

(3) 使用符合客户语言习惯的表达方式。在与客户进行需求交流时,要尽量站在客户的角度去使用术语,而客户却不需要懂得计算机行业方面的术语。

(4) 要多尊重客户的意见。客户与程序开发人员,偶尔也会碰到一些难以沟通的问题。如果客户与开发人员之间产生了不能相互理解的问题,要尽量多听听客户方的意见。能满足客户的需求,就要尽可能地满足客户的需求,如果实在是因为某些技术所限而无法实现,可以合理地地向客户说明。

(5) 划分需求的优先级。绝大多数项目没有足够的时间或资源实现功能性上的每一个细节。如果需要对哪些特性是必要的,哪些是重要的等问题做出决定,最好询问一下客户所设定的需求优先级。程序开发人员不应猜测客户的观点,然后去决定需求的优先级。

## 22.3 项目开发团队



本节视频教学录像: 8 分钟

所谓团队是指有限的一些人为了共同的目标而在一起工作,这些人将分担不同的角色,他们有着独特的贡献,一个组织很好的团队将包含所有的团队角色。所谓的项目团队是指,在明确的目标和共同价值观之下的一种特殊形式的团队,是为了完成某个一次性的特定任务(独特的产品或服务)而临时组建起来的团队。项目团队特征是指一种临时性的柔性组织,具有明确的生命周期,其成员是因某项具体的工作而加入团队的,在这个团队中不存在冗余成员。

### 22.3.1 项目团队组成

一个高效的项目开发团队,应拥有以下一些人员。

(1) 项目经理。项目经理根据需要可以有多位,包括设计项目经理、发行项目经理和协助项目经理等。其中设计项目经理主要负责具体的产品设计;发行项目经理主要负责整个项目的流程和进度管理,制定进度表,协调整个团队的工作;协助项目经理主要负责产品开发其他需要照顾到的事情,如与客户和市场开发人员交流,负责初版试行等。

(2) 开发团队。开发团队主要包括开发团队领导、开发组长、开发工程师和构架师等。其中开发团队领导主要负责管理各个开发小组,并对开发编程的工作做总体的规划;开发组长主要负责管理开发工程师,也参加对开发编程的工作做总体的规划;开发工程师主要负责具体的编程开发;构架师主要专门做整体系统的设计规划。

(3) 测试团队。测试团队主要包括测试团队领导、测试组长、测试工程师和测试开发工程师等。其中,测试团队领导主要负责管理测试小组;测试组长主要负责管理测试工程师,制定测试计划等;测试工程师主要负责具体的测试工作;测试开发工程师主要负责测试工具的开发。

(4) 产品可用性团队。产品可用性团队主要包括产品可用性工程师、界面设计师和产品设计师等。其中产品可用性工程师主要做使用性能的调查和测试,采访客户或将客户邀请来做调查等;界面设计师主要负责具体的界面设计;产品设计师主要负责产品的总体设计,特别是硬件产品。

(5) 文档团队。文档团队主要包括文档组长和文档编辑。文档组长主要负责文档小组的管理,文档

编辑主要负责具体的文档编辑和撰写。

一个项目开发团队并不是所有的产品队伍都要有，比较小的队伍就没有必要面面俱到。有的时候可以向别的队伍中借用职员，或者雇用临时工。

### 22.3.2 项目团队特征

一个高效的软件开发团队是需要建立在合理的开发流程及团队成员密切合作的基础之上的，每一个成员共同迎接挑战，有效地计划、协调和管理各自的工作以至完成明确的目标。高效的开发团队具有以下一些特征。

(1) 具有明确且有挑战性的共同目标。一个具有明确的且有挑战性目标的团队其工作效率会很高。因为在通常情况下，技术人员往往会为了完成了某个具有挑战性意义的任务而感到自豪，而反过来技术人员为了获取这种自豪的感觉会更加积极地工作，从而带来团队开发的高效率。因此高效的软件开发团队具有挑战性的共同目标。

(2) 团队具有很强的凝聚力。在一个高效的软件开发团队中，成员们的凝聚力表现为相互支持、互相交流和互相尊重，而不是相互推卸责任、保守、相互指责。例如，某个程序员明明知道另外的模块中需要用到一段与自己已经编写完成且有些难度的程序代码，但他就是不愿拿出来让其他的程序员共享，也不愿意与系统设计人员交流，这样就会为项目的顺利开展带来不良的影响。

(3) 具有融洽的交流环境。在一个开发团队中，每个开发小组人员行使各自的职责，例如系统设计人员做系统概要设计和详细设计，需求分析人员制定需求规格说明，项目经理配置项目开发环境并且制订项目计划等。但是由于种种原因，每个组员的工作不可能一次性做到位，如系统概要设计的文档可能有个别地方会出现词不达意，做详细设计的时候就有可能造成误解，因此高效的软件开发团队是具有融洽的交流环境的，而不是那种简单的命令执行式的。

(4) 具有共同的工作规范和框架。高效软件开发团队具有规范性及共同框架的工作特点。对于项目管理具有规范的项目开发计划，对于分析设计具有规范和统一框架的文档及审评标准，对于代码具有程序规范条例，对于测试有规范且可推理的测试计划及测试报告，等等。

(5) 采用合理的开发过程。软件项目的开发不同于一般商品的研发和生产，开发过程中面临着各种难以预测的风险，比如客户需求的变化、人员的流失、技术的瓶颈、同行的竞争，等等。高效的软件开发团队往往会采用合理的开发过程，去控制开发过程中的风险、提高软件的质量、降低开发的费用等。

## 22.4 项目计划说明书



本节视频教学录像：4 分钟

软件项目计划编制的目的是制定一个合理的实施软件工程及管理软件项目的计划。软件项目计划编制着重于对要实施的工作进行估计，建立必要的承诺并定义工作计划。一个好的软件项目计划可为项目的成功实施打下坚实的基础，因此在编辑项目计划说明书的时候要注意以下3点。

### 1. 重视项目计划的层次性和客户的沟通

在制订软件项目计划的时候,要合理地划分小组,减少组织的层次,这样有利于项目计划的制订和实施。一般情况下,项目的计划分为整体计划、详细计划和个人计划等。整体计划主要是进行项目的阶段划分,分配所需相关的资源,包括人力资源、设备资源和资金资源等。有了整体计划之后,还应做好下一阶段的详细计划,详细计划要确定各项任务的负责人、开始时间、结束时间、任务之间的依赖关系、设备资源、小的事件点等。开发人员的个人计划是由开发人员根据自己的任务自行制定。在项目开发的过程中与客户的沟通是很重要的。应该让客户知道项目的开发计划,和客户共享这些信息,特别是对项目目前的进度情况一定要与客户及时进行沟通。

### 2. 项目计划的实现过程

(1) 项目的初始需求,对范围、时间和成本等进行初步的估计,成立项目高级计划组以制定高级计划等,由项目经理负责完成。

(2) 对项目整体计划的审查要由客户方代表来参与,因为客户是项目风险的承担者之一。待项目计划通过审核后要进行封存。

(3) 在实施计划的过程中,要执行保证项目质量的要求,进行项目的监督和控制,以确保计划得以顺利实施。同时,在项目的实际开发过程中,可以根据项目的实际需要修正项目计划,在关键的地方还应就项目的执行情况与最初所制订的计划进行比较和分析,适时进行调整。已封存的项目计划也并不是不可以更改的,在更改时需要征得项目管理部门的同意。

### 3. 注重历史数据的积累

为了更好地实施项目计划,应该将项目计划的原始数据记录下来,以备以后使用。曾经失败的项目对后来的项目研发具有重要的参考价值,这些都是经验数据。有了历史记录,就可以根据它来制订计划,从而对以后的工作做进一步的改进。

## 22.5 项目开发阶段的运作



本节视频教学录像: 4 分钟

软件项目开发最根本的目标就是让客户最大化地使用软件产品,达到客户满意的效果。软件的生命周期在时间上被分解为 4 个顺序的阶段,分别是初始阶段、细化阶段、构建阶段和交付阶段。每个阶段的结束就相当于一个里程碑,同时也是下一个阶段的开始。

### 22.5.1 初始阶段

初始阶段的目标是为系统建立商业案例并确定项目的边界。为了达到这一目标,必须识别所有与系统交互的外部实体,在较高层次上定义交互的特性。在初始阶段最为关注的是整个项目进行中的业务和需求方面的主要风险,这些对于建立在原有系统基础上的开发项目来讲,初始阶段有可能很短。虽然初始阶段很短暂,但是意义同样重大,因为初始阶段做得好坏直接决定了项目的基本生存能力。

## 22.5.2 细化阶段

细化阶段的目标是分析问题领域，建立健全的体系结构基础，编制项目计划，淘汰项目中最高风险的元素。为了达到这一目的，必须在理解整个项目的基础上，对体系结构做出决策，主要包括对体系结构的范围、主要功能和主要性能等非功能的需求。同时为项目建立支持环境，包括创建开发案例，创建模板、准则并准备工具。

## 22.5.3 构建阶段

在构建阶段，所有剩余的构件和应用程序功能被开发并集成为产品，同时，所有的功能也将被详细测试。从某种意义上说，构建阶段是一个制造过程，其重点应放在管理资源及控制运作，以优化成本、进度和质量。

## 22.5.4 交付阶段

交付阶段的重点是确保软件对最终用户是可用的。交付阶段可以跨越几次迭代，包括为项目发布做出准备的产品测试，基于用户反馈的少量调整。在生命周期的这一点上，客户反馈可能会主要集中在产品调整、设置、安装和可用性方面，所有主要的结构问题应该已经在项目生命周期的早期阶段解决了，因此在这里不会刻意关注产品的结构问题。

## 22.5.5 维护阶段

软件维护主要是指根据需求变化或硬件环境的变化对应用程序进行部分或全部的修改，修改时应充分利用源程序，修改后要填写程序修改登记表，并在程序变更通知书上写明新旧程序的不同之处。软件维护的最终目标是使产品具有正确性、适应性、完善性和预防性。

# 22.6 高手点拨



本节视频教学录像：2 分钟

项目的开发过程并不是一两天就可以做好的。对于一个复杂的项目来说，其开发过程更是充满了曲折和艰辛，其问题也是层出不穷、接连不断的。

### 1. 项目进度难于控制，项目管理难度加大

大量的软件错误通常只有到了项目后期，在进行系统测试时才能够被发现。解决问题所花费的时间通常很难预料，经常导致项目进度无法控制。同时在整个软件开发的过程中，项目管理人员缺乏对软件质量状况的了解和控制，也会加大项目管理的难度。

面对这种情况，较好的解决方法是尽早进行测试，当软件的第 1 个过程结束后，测试人员要马上基于它进行测试脚本的实现，按项目计划中的测试目的执行测试用例，对测试结果进行评估报

告。这样就可以通过各种测试指标实时监控项目得质量状况，以提高对整个项目的控制和管理的能力。

2. 软件项目开发费用超出预算

在整个项目开发的过程中，错误发现得越晚，单位错误修复成本就会越高，错误地延迟解决问题，必然会导致整个项目成本的急剧增加。

解决这个问题的最好方法是采取多种测试手段，尽早发现潜在的问题。

# 第23章



本章视频教学录像：59 分钟

## 网络通信应用开发——VB 实现远程控制


随着时代的发展，网络与通信技术的应用也越来越广泛，各种编程语言对网络通信的开发更是百家争鸣。本章详细介绍了如何利用 Visual Basic 6.0 实现网络间的相互会话，并实现服务器端远程控制客户端关机、重启等。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解网络通信的相关知识
- ☐ 实现网络间的相互会话
- ☒ 实现远程控制重启和关机



## 23.1 系统分析

 本节视频教学录像: 3 分钟

对于大型的公司来讲,要检测某个员工是否在岗或者要与其互通信息,需要一个内部的检测和通信工具。在本章我们使用 Visual Basic 6.0 设计一个简单的网络通信工具,从对系统的要求来讲,该工具要实现以下三个功能。

第一,检测用户是否在线。

第二,实现用户之间可以自由通信。

第三,设计一个管理端,实现对用户的远程关机、重启。

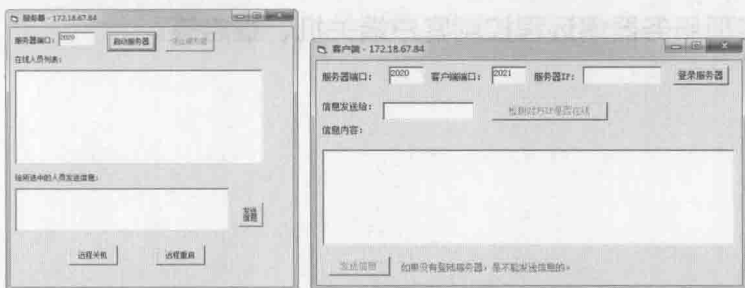
通过对上述要求功能的分析,在设计时可以将系统分为两个模块:客户端模块和服务器端模块,两模块间主要负责的工作为:

(1) 服务器端可以发出指令给客户端,客户端不能够发出指令给服务器端。

(2) 服务器端可以对客户端进行远程控制,如关机、重启等。

(3) 客户端可以与客户端进行会话通信。

服务器端和客户端运行后的效果如图所示。



下面分别对服务器端和客户端两个模块的设计思想进行相关说明。

### 1. 服务器端

当启动服务器后,服务器开始处于监听状态,当有客户端根据服务器的端口号和服务器 IP 地址进行登录后,服务器端会列出已经连接上的客户端的 IP 地址。建立连接后,服务器端可以根据需要对客户端发出指令,如提醒客户端注意保存文件,5 分钟后将会关机等,同时,也可以根据客户端的 IP 地址,直接进行关机、重启等操作。



注意

服务器端若断开服务器,所有的客户端均不能进行通信会话,服务器对客户端也不能发出任何指令和进行操作,待服务器启动后,客户端重新登录才可以再次进行相互会话。

### 2. 客户端


客户端根据服务器的端口号和服务器 IP 地址登录后,便开始对服务器发出连接请求,同时,客户端开始处于正在连接服务器状态,等待服务器的响应。待服务器端给出应答后,双方便可以建立连接。客户端可以根据 IP 地址进行检索其他客户机是否也连接到了服务器上,如果其他客户端也得到了服务器的连接应答,那么该客户端便可以根据 IP 地址对其进行双方会话通信。

在 Visual Basic 6.0 中对网络程序的编写,主要使用的是 Winsock 控件。Winsock 控件能够通过



UDP（用户数据报协议）或 TCP（数据传输协议）连接到远程的机器并进行数据交换，这两种协议都能用来创建客户端和服务端应用程序。其工作原理是，客户端向服务器端发出连接请求，服务器端则不停地监听客户端的请求，当两者的协议沟通时，客户端和服务端之间就建立了连接，这时客户端和服务端就可以实现双向数据传输。

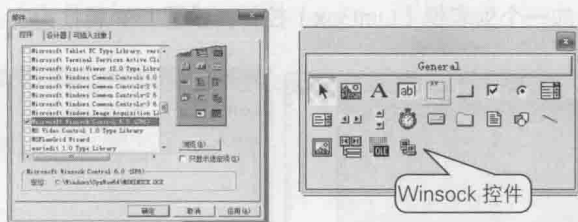
## 23.2 系统设计

 本节视频教学录像：12 分钟

在对系统分析之后，明确两个模块需要实现的具体功能以及所用的控件。本节分 7 个步骤详细地讲述了服务器端和客户端的界面设计和代码编写过程。

### 第 1 步：添加部件

- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。
- (2) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令，弹出【部件】对话框。
- (3) 在【控件】选项卡中选中【Microsoft Winsock Control 6.0】复选框，然后单击【确定】按钮。
- (4) 这样就将 Winsock 控件添加到了工具箱中。



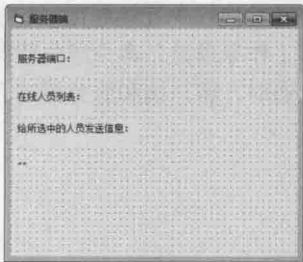
在进行客户端和服务端的设计时均需要添加 Winsock 控件。

注意

### 第 2 步：服务器端界面设计

- (1) 将 Form1 窗体的 Caption 属性设置为“服务器端”。
- (2) 在 Form1 窗体中添加 4 个标签控件，分别按下表设置其属性。

控件名称	名称	Caption	控件作用
Label1	Label1	服务器端口：	提示记录中字段内容的意义
Label2	Label2	在线人员列表：	提示记录中字段内容的意义
Label3	Label3	给所选中的人员发送信息：	提示记录中字段内容的意义
Label4	Label4	“ ”	提示记录中字段内容的意义



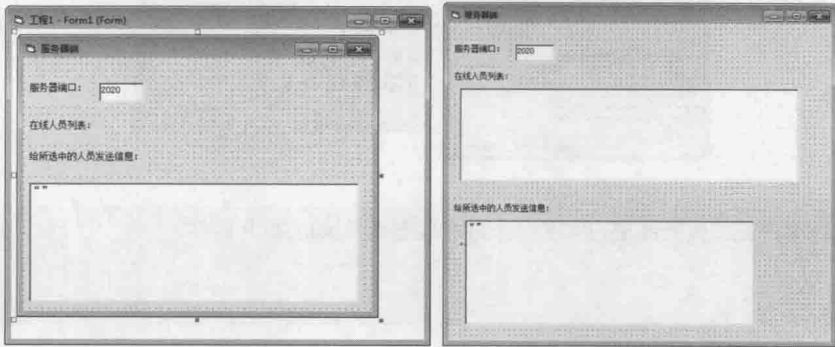
提示

将 Label4 的 AutoSize 属性设置为 True，用于根据文本内容自动调整文本大小。

(3) 在 Form1 窗体中添加两个文本框控件，分别按下表设置其属性。

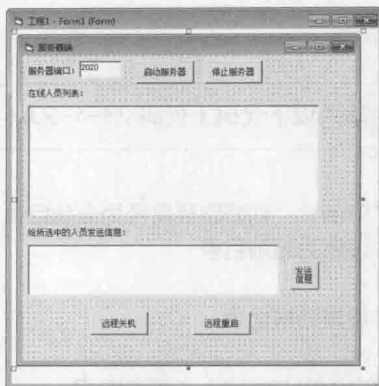
控件名称	名称	Text	控件作用
TextBox	Text1	2020	显示服务器端口号
TextBox	Text2	“ ”	用于向客户端发送文本信息

(4) 在 Form1 窗体上添加一个列表框（ListBox）控件，设置 List 属性为空。



(5) 在 Form1 窗体中添加 5 个按钮控件，分别按下表设置其属性。

控件名称	名称	Caption	控件作用
Command1	Command1	启动服务器	用于服务器的启动
Command2	Command2	停止服务器	用于断开服务器的运行
Command3	Command3	发送信息	用于向客户端发送信息
Command4	Command4	远程关机	用于关闭客户端的计算机
Command5	Command5	远程重启	用于重启客户端的计算机



(6)在 Form1 窗体中添加一个 Timer 控件和一个 Winsock 控件，将 Winsock 的【名称】属性设置为“W1”，【Index】属性设置为“0”，并将 Timer 控件的【Interval】属性设置为“3000”。各控件按下图进行排列。将 Label4 控件放置在界面的最下端用来显示相关的提示信息。



### 第 3 步：编写服务器端代码

(1) 在 Form1 窗体上用右键单击，在快捷菜单中选择【查看代码】菜单项，进入代码窗口，输入以下代码（代码 23-1-1.txt）。

```
01 Option Explicit
02 Private Declare Sub InitCommonControls Lib "comctl32.dll" () '使用通用控件
03 Dim MaxIndex As Long '最大连接数
04 Dim WinSockSendOK As Boolean '值为 True 表示上次发送的信息完成
05 Dim intFreeFileName As String '当前要保存的文件名
06 Dim intFreeFile As Long '当前使用的文件号
07 Dim myPath As String '程序所在位置
08 Const QuitMsg = "CMD:QUITLINK"
09 Const CMD1 = "CMD:QUITWINDOWS" '关机
10 Const CMD2 = "CMD:RESETWINDOWS" '重启
11 Const CMD3 = "CMD:TESTUSER=" '测试人员是否在线
```

(2) 在 Form 窗体的 Initialize 事件中输入以下代码。

```
Private Sub Form_Initialize()
```

```
InitCommonControls ' 初始化控件
End Sub
```

(3) 在 Form 窗体的 Load 事件中输入以下代码 (代码 23-1-2.txt)。

```
01 Private Sub Form_Load()
02 If Right(App.Path, 1) <> "\" Then ' 判断路径是否为当前目录
03     myPath = App.Path & "\" 设置为当前目录
04 Else
05     myPath = App.Path ' 设置为根目录
06 End If
07 Me.Caption = "服务器 - " & W1(0).LocalIP ' 取本地 IP
08 List1.Clear ' 清空列表
09 Command1.Enabled = True ' 【启动服务器】按钮可用
10 Command2.Enabled = False ' 【启动服务器】按钮不可用
11 End Sub
```

(4) 在 Form 窗体的 QueryUnload 事件中输入以下代码。

```
01 Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
02 Call CloseUserLc ' 断开所有连接
03 DoEvents ' 转让控制
04 End ' 退出系统
05 End Sub
```

(5) 在 Form1 窗体中双击【启动服务器】按钮，在打开的代码窗口中输入以下代码 (代码 23-1-3.txt)。

```
01 Private Sub Command1_Click()
02 List1.Clear ' 清空在线人员列表
03 Command1.Enabled = False ' 【启动服务器】按钮不可用
04 If Val(Text1.Text) > 0 Then ' 判断服务器端口号是否大于零
05 W1(0).LocalPort = Val(Text1.Text) ' 获取服务器端口号
06 Else
07 W1(0).LocalPort = 2020 ' 设置默认服务器端口号
08 End If
09 Text1.Text = W1(0).LocalPort ' 显示服务器端口号
10 W1(0).Bind ' 绑定
11 W1(0).Listen ' 开始监听
12 Command2.Enabled = True ' 【停止服务器】按钮可用
13 End Sub
```

(6) 在 Form1 窗体中双击【停止服务器】按钮，在打开的代码窗口中输入以下代码。

```
01 Private Sub Command2_Click()
02 List1.Clear ' 清空在线人员列表
03 Call CloseUserLc ' 断开所有连接
04 Command2.Enabled = False ' 【停止服务器】按钮不可用
05 Command1.Enabled = True ' 【启动服务器】按钮可用
```

06 End Sub

(7) 在 Form1 窗体中双击【发送信息】按钮，在打开的代码窗口中输入以下代码（代码 23-1-4.txt）。

```
01 Private Sub Command3_Click()
02   Dim tmp As String, i As Integer
03   If Len(Text2.Text) = 0 Then Exit Sub      ' 判断消息内容长度
04   If List1.ListIndex < 0 Then Exit Sub      ' 判断人员列表有无人员
05   tmp = List1.List(List1.ListIndex)        ' 获取人员列表信息
06   tmp = Mid(tmp, 1, InStr(1, tmp, " ") - 1)
07   Label4.Caption = ""                      ' 清空信息提示文本
08   i = BIUserState(tmp)                    ' 调用函数
09   If i <> -1 Then
10     Call SendString(i, "MSGFORM: 服务器 MSG:" & Text2.Text) ' 发送消息
11     Label4.Caption = "消息发送成功!"      ' 信息提示
12   Else
13     Label4.Caption = "消息发送失败!"      ' 信息提示
14   End If
15 End Sub
```

(8) 在 Form1 窗体中双击【远程关机】按钮，在打开的代码窗口中输入以下代码（代码 23-1-5.txt）。

```
01 Private Sub Command4_Click()
02   Dim tmp As String, i As Integer          ' 定义变量
03   If List1.ListIndex < 0 Then Exit Sub      ' 判断客户列表为空，不执行关机
04   If MsgBox("确定要关闭此客户端计算机吗?", 32 + 4) <> vbYes Then Exit Sub
05   tmp = List1.List(List1.ListIndex)        ' 获取客户机器列表信息
06   tmp = Mid(tmp, 1, InStr(1, tmp, " ") - 1)
07   Label4.Caption = ""                      ' 清空消息提示信息
08   i = BIUserState(tmp)                    ' 调用函数，获取 IP
09   If i <> -1 Then
10     Call SendString(i, CMD1)               ' 调用函数进行关机
11     Label4.Caption = "正在进行远程关机……" ' 提示信息
12   Else
13     Label4.Caption = "远程关机失败!"      ' 提示信息
14   End If
15 End Sub
```

(9) 在 Form1 窗体中双击【远程重启】按钮，在打开的代码窗口中输入以下代码（代码 23-1-6.txt）。

```
01 Private Sub Command5_Click()
02   Dim tmp As String, i As Integer          ' 定义变量
03   If List1.ListIndex < 0 Then Exit Sub      ' 用户列表为空不进行重启操作
04   If MsgBox("确定要重新启动此客户端计算机吗?", 32 + 4) <> vbYes Then Exit Sub
05   tmp = List1.List(List1.ListIndex)        ' 获取客户端列表信息
06   tmp = Mid(tmp, 1, InStr(1, tmp, " ") - 1)
07   Label4.Caption = ""                      ' 清空提示信息
```

```

08 i = BIUserState(tmp) '调用函数, 获取 IP
09 If i <> -1 Then
10     Call SendString(i, CMD2) '调用函数, 进行重启
11     Label4.Caption = " 正在进行远程重启....." '提示信息
12 Else
13     Label4.Caption = " 远程重启失败! " '提示信息
14 End If
15 End Sub

```

#### 第4步: 编写服务器端的自定义函数

(1) 在代码窗口中编写关闭与所有机器进行链接的 CloseUserLc() 函数 (代码 23-1-7.txt)。

```

01 Private Sub CloseUserLc()
02     On Error Resume Next '存在连接错误, 暂不理睬
03     Dim i As Integer '定义变量
04     For i = 0 To MaxIndex
05         W1(i).Close '断开连接
06     Next i
07     For i = 1 To MaxIndex
08         Unload W1(i) '删除动态添加的连接数
09     Next i
10 End Sub

```

(2) 在代码窗口中编写时钟控件的 Timer 事件代码, 用于信息提示随着时间的变化而改变。

```

01 Private Sub Timer1_Timer()
02     If Label4.Caption <> "" Then
03         Label4 = "" '清空提示信息
04     End If
05 End Sub

```

(3) 编写 Winsock 控件的 Close 事件代码, 用于在关闭与客户端的连接时执行相关程序。

```

01 Private Sub W1_Close(Index As Integer)
02     On Error Resume Next '存在连接错误, 暂不理睬
03     W1(Index).Close '断开连接
04     If Index <> 0 Then Unload W1(Index) '立即删除未使用的 Winsock 控件把所占资源还给系统
05     Call RefurbishList '调用函数, 刷新客户端列表
06 End Sub

```

(4) 编写 Winsock 控件的 ConnectionRequest 事件代码, 用于当远程计算机请求连接时所执行的相关程序 (代码 23-1-8.txt)。

```

01 Private Sub W1_ConnectionRequest(Index As Integer, ByVal requestID As Long)
02     On Error Resume Next '存在连接错误, 暂不理睬
03     If W1(Index).State <> sckClosed Then '判断 State 状态
04         W1(Index).Close '关闭 Winsock 连接

```

```

05 End If
06 W1(Index).Accept requestID ' 接受连接
07 If err.Number <> 0 Then Exit Sub ' 如果出现错误就退出过程
08 MaxIndex = MaxIndex + 1
09 Load W1(MaxIndex) ' 再添加一个线程
10 W1(MaxIndex).Listen ' 继续监听
11 Call RefurbishList ' 刷新在线人员列表
12 End Sub

```

(5) 编写 Winsock 控件的 DataArrival 事件代码，用于接收数据时执行相关程序（代码 23-1-9.txt）。

```

01 Private Sub W1_DataArrival(Index As Integer, ByVal bytesTotal As Long)
02 Dim strCommand As String ' 接收到的数据
03 Dim sendTO As String
04 Dim sendMsg As String
05 Dim sendForm As String
06 Dim i As Long ' 计数器
07 On Error Resume Next ' 存在连接错误，暂不理睬
08 W1(Index).GetData strCommand ' 获取客户端发出的命令
09 strCommand = RTrim(strCommand) ' 去除多的空格
10 If strCommand = QuitMsg Then ' 判断并执行客户端发出的信息
11 W1(Index).Close ' 关闭连接
12 Unload W1(Index)
13 Call RefurbishList ' 刷新在线人员列表
14 ElseIf Left(strCommand, Len(CMD3)) = CMD3 Then
15 sendTO = Right(strCommand, Len(strCommand) - Len(CMD3))
16 If Len(sendTO) = 0 Then Exit Sub ' 人员列表为零，不进行操作
17 If UCase(sendTO) = "EVERYONE" Then ' 向所有人员发送是否在线的请求
18 Call SendString(Index, "TESTYES"): Exit Sub ' 直接回答在
19 End If
20 i = BUUserState(sendTO) ' 单个发送在线请求
21 If i <> -1 Then
22 Call SendString(Index, "TESTYES") ' 在
23 Else
24 Call SendString(Index, "TESTNO") ' 不在
25 End If
26 ElseIf Left(strCommand, 3) <> "TO:" Or InStr(1, strCommand, "MSG:") = 0 Then
27 Call SendString(Index, "Server : 消息格式错误! ") ' 发送消息
28 Else
29 sendTO = Mid(strCommand, 4, InStr(1, strCommand, "MSG:") - 4) ' 获取消息的目的地
30 sendMsg = Right(strCommand, Len(strCommand) - InStr(1, strCommand, "MSG:") - 3)
' 消息内容
31 If Len(sendMsg) = 0 Then ' 判断消息是否为空
32 Call SendString(Index, "Server : 不能发送空消息! "): Exit Sub ' 不能发送消息
33 End If
34 If W1(Index).RemoteHost <> "" Then

```



```

35     sendForm = W1(Index).RemoteHost      ' 发送计算机名
36     Else
37     sendForm = W1(Index).RemoteHostIP    ' 发送计算机 IP
38     End If
39     If UCase(sendTO) = "EVERYONE" Then ' 判断是否发送给所有在线人员
40         For i = 0 To MaxIndex
41             If W1(i).State = 7 Then
42                 Call SendString(i, "MSGFORM:" & sendForm & "MSG:" & sendMsg)
' 发送给所有在线人员
43             End If
44         Next
45     Else
46         i = BIUserState(sendTO)          ' 判断人员是否在线
47         If i <> -1 Then Call SendString(i, "MSGFORM:" & sendForm & "MSG:" & sendMsg)
' 发送消息
48     End If
49     End If
50     DoEvents
51     Call SendString(Index, Chr(0) & "OK") ' 成功发送消息
52     Exit Sub
53     err: ' 错误处理
54     Call SendString(Index, "Server : 超时错误! ") ' 发送消息超时
55     End Sub

```

(6) 编写 Winsock 控件的 Error 事件代码, 用于连接错误时执行的相关程序。

```

01 Private Sub W1_Error(Index As Integer, ByVal Number As Integer, Description As String,
ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long,
CancelDisplay As Boolean)
02 On Error Resume Next ' 存在连接错误, 暂不理睬
03 W1(Index).Close ' 断开连接
04 W1(MaxIndex).Listen ' 继续监听
05 Call RefurbishList ' 调用函数, 刷新列表
06 End Sub

```

(7) 编写 Winsock 控件的 SendComplete 事件和 SendProgress 事件代码, 用于处理发送消息过程和完成发送消息后执行的相关程序。

```

01 Private Sub W1_SendComplete(Index As Integer)
02 WinSockSendOK = False ' 发送成功后关闭
03 End Sub
04 Private Sub W1_SendProgress(Index As Integer, ByVal bytesSent As Long, ByVal bytesRemaining
As Long)
05 WinSockSendOK = True ' 正在发送中要打开
06 End Sub

```

(8) 编写 SendString 函数, 用于正确地发送一个字符串 (代码 23-1-10.txt)。

```

01 Public Sub SendString(ByVal Index As Long, ByVal strSend As String)
02   Do While WinSockSendOK = True ' 如果正在发送
03     DoEvents      ' 转让控制权
04     Loop
05   If W1(Index).State = sckConnected Then W1(Index).SendData strSend      ' 如果已连接就
发送数据
06   DoEvents      ' 转让控制权
07 End Sub

```

(9) 编写 RefurbishList 函数，用于刷新在线客户列表（代码 23-1-11.txt）。

```

01 Public Sub RefurbishList()
02   List1.Clear      ' 清空客户端列表
03   On Error Resume Next ' 存在连接错误，暂不理睬
04   Dim i As Integer
05   For i = 0 To MaxIndex
06     If W1(i).State = 7 Then ' 判断服务器是否启动
07       If W1(i).RemoteHost <> "" Then
08         List1.AddItem W1(i).RemoteHost & " 远程端口：" & W1(i).RemotePort
' 读取客户端主机端口
09       Else
10         If W1(i).RemoteHostIP <> "" Then List1.AddItem W1(i).RemoteHostIP & " 远程端口：" &
W1(i).RemotePort ' 读取客户主机端口和 IP
11       End If
12     End If
13   Next
14 End Sub

```

(10) 编写 BIUserState 函数，用于判断一个人员是否在线，如果在线就返回连接的 ID（代码 23-1-12.txt）。

```

01 Public Function BIUserState(user As String) As Integer
02   Dim i As Integer
03   On Error Resume Next ' 存在连接错误，暂不理睬
04   BIUserState = -1
05   For i = 0 To MaxIndex
06     If W1(i) = 7 Then ' 判断服务器是否启动
07       If UCase(W1(i).RemoteHost) = UCase(user) Or W1(i).RemoteHostIP = user Then BIUserState
= i: Exit Function
08     End If
09   Next
10 End Function

```

### 第 5 步：客户端界面设计

- (1) 新建一个工程，在新建的工程中将 Form1 窗体的 Caption 属性设置为“客户端”。
- (2) 在 Form1 窗体中添加 6 个标签控件，分别按下表设置其属性。

控件名称	名称	Caption	控件作用
Label1	Label1	服务器端口：	提示记录中字段内容的意义
Label2	Label2	客户端端口：	提示记录中字段内容的意义
Label3	Label3	服务器 IP：	提示记录中字段内容的意义
Label4	Label4	信息发送给：	提示记录中字段内容的意义
Label5	Label5	信息内容：	提示记录中字段内容的意义
Label6	Msg1	如果没有登录服务器，是不能发送信息的	程序运行过程中的信息提示

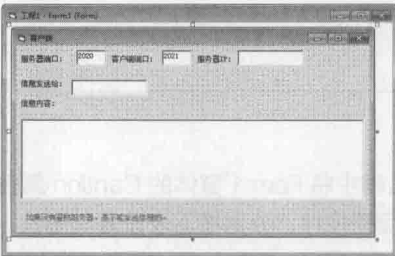


可以将 Label6 的【ForeColor】属性设置为红色，以增加提示信息的注意力度。

技巧

(3) 在 Form1 窗体中添加 5 个文本框控件，分别按下表设置其属性。

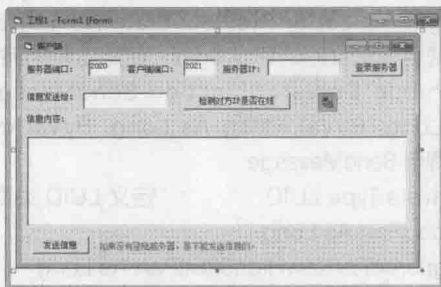
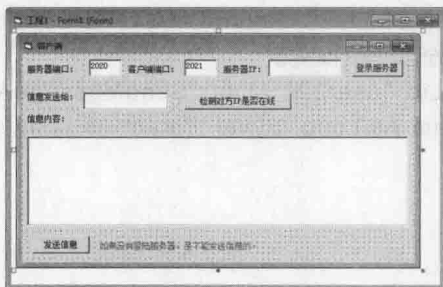
控件名称	名称	Text	控件作用
TextBox	Text1	2020	显示服务器端口号
TextBox	Text2	2021	显示客户端端口号
TextBox	Text3	“ ”	输入发送信息的内容
TextBox	Text4	“ ”	输入服务器端 IP 地址
TextBox	Text5	“ ”	输入客户端 IP 地址



(4) 在 Form1 窗体上添加 3 个命令按钮控件，分别按下表设置其属性。

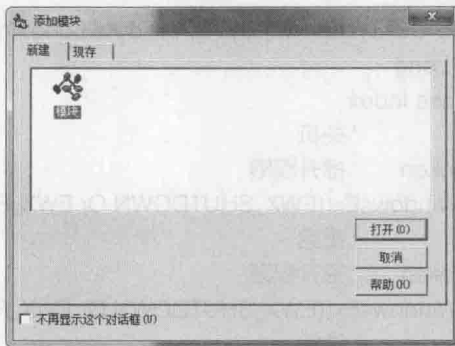
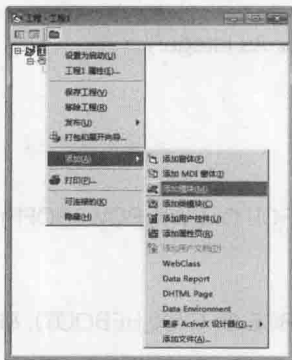
控件名称	名称	Caption	控件作用
Command1	Command1	登录服务器	用于登录服务器
Command2	Command2	检测对方 IP 是否在线	用于检测与对方 IP 是否可以通信
Command3	Command3	发送信息	用于向客户端发送信息

(5) 在 Form1 窗体中添加一个 Winsock 控件，将 Winsock 的【名称】属性设置为“W1”，各控件按下图进行排列。



#### 第 6 步：编写客户端公用模块

- (1) 在【工程】窗口中右击鼠标，在弹出的快捷菜单中选择【添加】>【添加模块】菜单命令。
- (2) 在打开的【添加模块】对话框中单击【打开】按钮。



- (3) 在模块的代码窗口中输入以下代码（代码 23-1-13.txt）。

```

01 Option Explicit ' 对各部分变量进行声明
02 Private Const EWX_SHUTDOWN As Long = 1
03 Private Const EWX_FORCE As Long = 4
04 Private Const EWX_REBOOT = 2
05 Private Const EWX_LOGOFF = 0
06 Private Const EWX_POWEROFF = 8
07 Public Const WM_SYSCOMMAND = &H112&
    
```

```

08 Public Const SC_SCREENSAVE = &HF140&
09 Private Declare Function ExitWindowsEx Lib "user32" (ByVal dwOptions As Long, ByVal dwReserved
As Long) As Long '调用 ExitWindowsEx, 以关闭计算机
10 Private Declare Function GetCurrentProcess Lib "kernel32" () As Long
'调用 GetCurrentProcess, 以获取当前进程的句柄
11 Private Declare Function OpenProcessToken Lib "advapi32" (ByVal
ProcessHandle As Long, ByVal DesiredAccess As Long, TokenHandle As Long) As Long
'调用 OpenProcessToken, 以打开一个进程的访问代号
12 Private Declare Function LookupPrivilegeValue Lib "advapi32" Alias "LookupPrivilegeValueA"
(ByVal lpSystemName As String, ByVal lpName As String, lpLuid As LUID) As Long
'调用 LookupPrivilegeValue, 获得本地唯一的标示符 (LUID)
13 Private Declare Function AdjustTokenPrivileges Lib "advapi32" (ByVal TokenHandle
As Long, ByVal DisableAllPrivileges As Long, NewState As TOKEN_PRIVILEGES, ByVal
BufferLength As Long, PreviousState As TOKEN_PRIVILEGES, ReturnLength As Long) As Long
'调用 AdjustTokenPrivileges, 使能或者禁用指定访问记号的优先权
14 Public Declare Function SendMessage Lib "user32" Alias "SendMessageA" (ByVal
hwnd As Long, ByVal wParam As Long, ByVal lParam As Long, ByVal lParam As Long) As Long
'调用 SendMessage
15 Private Type LUID '定义 LUID 类型
16 UsedPart As Long
17 IgnoredForNowHigh32BitPart As Long
18 End Type
19 Private Type TOKEN_PRIVILEGES '定义 TOKEN_PRIVILEGES 类型
20 PrivilegeCount As Long
21 TheLuid As LUID
22 Attributes As Long
23 End Type
24 Public Sub QuitSystemWin(ByVal hwnd As Long, Index As Integer)
25 Dim i As Long
26 Select Case Index
27 Case 0 '关机
28 AdjustToken '提升权限
29 i = ExitWindowsEx((EWX_SHUTDOWN Or EWX_FORCE Or EWX_POWEROFF), &HFFFF)
30 Case 1 '重启
31 AdjustToken '提升权限
32 i = ExitWindowsEx((EWX_SHUTDOWN Or EWX_FORCE Or EWX_REBOOT), &HFFFF)
33 Case 2 '注销
34 AdjustToken '提升权限
35 i = ExitWindowsEx((EWX_FORCE Or EWX_LOGOFF), &HFFFF)
36 End Select
37 End Sub
38 Private Sub AdjustToken() '当需要提升权限时, 执行以下命令
39 Const TOKEN_ADJUST_PRIVILEGES = &H20
40 Const TOKEN_QUERY = &H8
41 Const SE_PRIVILEGE_ENABLED = &H2
42 Dim hdlProcessHandle As Long

```

```

43 Dim hdlTokenHandle As Long
44 Dim tmpLuid As LUID
45 Dim tkp As TOKEN_PRIVILEGES
46 Dim tkpNewButIgnored As TOKEN_PRIVILEGES
47 Dim lBufferNeeded As Long
48 hdlProcessHandle = GetCurrentProcess() ' 得到当前进程的句柄, 并赋给 hdlProcessHandle
49 OpenProcessToken hdlProcessHandle, (TOKEN_ADJUST_PRIVILEGES Or _
50 TOKEN_QUERY), hdlTokenHandle ' 打开 hdlTokenHandle 进程的访问代号
51 LookupPrivilegeValue "", "SeShutdownPrivilege", tmpLuid
52 tkp.PrivilegeCount = 1
53 tkp.TheLuid = tmpLuid
54 tkp.Attributes = SE_PRIVILEGE_ENABLED
55 AdjustTokenPrivileges hdlTokenHandle, False, tkp, Len(tkpNewButIgnored),
tkpNewButIgnored, lBufferNeeded
56 End Sub

```



#### 提示

该模块主要用于支持关机、重启等操作。

(4) 将公用模块以 “mod\_exitwindows” 为名, 保存在 “Final\ch23\客户端” 文件夹下。

#### 第 7 步: 编写客户端代码

(1) 在 Form1 窗体上用右键单击, 在弹出的快捷菜单中选择【查看代码】菜单项, 进入代码窗口, 输入以下代码 (代码 23-1-14.txt)。

```

01 Option Explicit
02 Private Declare Sub InitCommonControls Lib "comctl32.dll" () ' 声明 API 函数
03 Private Declare Function GetSystemDirectory Lib "kernel32" Alias "GetSystemDirectoryA" (ByVal
lpBuffer As String, ByVal nSize As Long) As Long
04 Dim mIndex As Integer
05 Dim SendBusy As Boolean
06 Const QuitMsg = "CMD:QUITLINK"
07 Const CMD1 = "CMD:QUITWINDOWS" ' 关机
08 Const CMD2 = "CMD:RESETWINDOWS" ' 重启
09 Const CMD3 = "CMD:TESTUSER=" ' 测试用户是否在线

```

(2) 在 Form 窗体的 Initialize 事件中输入以下代码。

```

01 Private Sub Form_Initialize()
02 InitCommonControls ' 初始化控件
03 End Sub

```

(3) 在 Form 窗体的 Load 事件中输入以下代码。

```

01 Private Sub Form_Load()

```

```

02 Me.Caption = "客户端 - " + W1.LocalIP      ' 显示客户端 IP
03 Command2.Enabled = False                  ' 不允许检测对方 IP
04 Command3.Enabled = False                  ' 不允许发送信息
05 Command1.Caption = "登录服务器"           ' 设置按钮显示内容
06 End Sub

```

(4) 在 Form 窗体的 QueryUnload 事件中输入以下代码 (代码 23-1-15.txt)。

```

01 Private Sub Form_QueryUnload(Cancel As Integer, UnloadMode As Integer)
02 If Command1.Caption = "断开" Then          ' 判断是否允许断开
03 W1.SendData QuitMsg ' 发送数据
04 DoEvents                                   ' 转让控制权, 以便让操作系统处理其他的事件
05 End If
06 Unload Me                                  ' 卸载窗体
07 End Sub

```

(5) 在 Form1 窗体中双击【登录服务器】按钮, 在打开的代码窗口中输入以下代码 (代码 23-1-16.txt)。

```

01 Private Sub Command1_Click()
02 If Command1.Caption = "登录服务器" Then
03 On Error GoTo err ' 发现错误去处理
04 If Len(Text4.Text) = 0 Then Exit Sub      ' 服务器 IP 为空不处理
05 If Val(Text2.Text) < 0 Then Exit Sub      ' 客户端端口号小于零不处理
06 If Val(Text1.Text) < 0 Then Exit Sub     ' 服务器端口号小于零不处理
07 W1.LocalPort = Val(Text2.Text) ' 获取本地端口号
08 W1.RemotePort = Val(Text1.Text) ' 获取服务器端口号
09 W1.RemoteHost = Text4.Text ' 获取对方 IP 地址
10 Command1.Enabled = False ' 【登录服务器】按钮不可用
11 Msg1.Caption = "正在登录到服务器 " & W1.RemoteHost & "....." 提示信息
12 W1.Connect ' 连接服务器
13 Else
14 W1.SendData QuitMsg ' 发送关闭连接的命令
15 End If
16 Exit Sub
17 err:
18 Msg1.Caption = err.Description ' 显示错误信息
19 err.Clear ' 清空错误
20 On Error Resume Next ' 发现错误暂不理睬
21 W1.Close ' 关闭与服务器的链接
22 Command2.Enabled = False ' 不能检测对方 IP
23 Command3.Enabled = False ' 不能发送信息
24 Command1.Caption = "登录服务器" ' 设置按钮控件的属性
25 Command1.Enabled = True ' 【登录服务器】按钮可用
26 End Sub

```

(6) 在 Form1 窗体中双击【检测对方 IP 是否在线】按钮, 在打开的代码窗口中输入以下代码。



```
01 Private Sub Command2_Click()
02 W1.SendData CMD3 & Text5.Text '发送数据, 等待回应
03 End Sub
```

(7) 在 Form1 窗体中双击【发送信息】按钮, 在打开的代码窗口中输入以下代码 (代码 23-1-17.txt)。

```
01 Private Sub Command3_Click()
02 If Len(Text3.Text) < 1 Then
03     MsgBox "不能发送空消息!", 48 '提示信息
04     Exit Sub
05 End If
06 Msg1.Caption = "正在发送....." '显示状态
07 W1.SendData "TO:" & Text5.Text & "MSG:" & Text3.Text '发送消息, 让服务器处理
08 End Sub
```

(8) 在代码窗口中编写 Winsock 控件的 Connect 事件和 Close 事件的相关代码, 用于连接服务器和关闭与服务器的连接 (代码 23-1-18.txt)。

```
01 Private Sub W1_Close() '关闭连接
02 On Error Resume Next '存在连接错误, 暂不理睬
03 Msg1.Caption = "连接中断, 服务器已经结束了此连接。" '提示信息
04 W1.Close '关闭连接
05 Command2.Enabled = False '不能检测 IP
06 Command3.Enabled = False '不能发送消息
07 Command1.Caption = "登录服务器" '设置按钮控件的属性
08 Command1.Enabled = True '按钮控件可用
09 End Sub
10 Private Sub W1_Connect() '建立连接
11 Command2.Enabled = True '允许检测 IP
12 Command3.Enabled = True '允许发送消息
13 Command1.Caption = "断开服务器" '设置按钮控件的属性
14 Msg1.Caption = "登录成功" '显示信息
15 Command1.Enabled = True '按钮控件可用
16 End Sub
```

(9) 在代码窗口中编写 Winsock 控件的 DataArrival 事件的代码, 用于接收到信息后的处理 (代码 23-1-19.txt)。

```
01 Private Sub W1_DataArrival(ByVal bytesTotal As Long)
02 Dim strCommand As String '命令字符串
03 Dim getMsg As String
04 Dim getForm As String
05 Dim i As Long
06 On Error Resume Next '发现错误, 暂不处理
07 W1.GetData strCommand '获取客户端发出的命令
08 strCommand = RTrim(strCommand) '去除多余的空格
09 If strCommand = "TESTYES" Then '判断并执行服务端发出的信息
```

```

10  MsgBox "用户 " & Text5.Text & " 在线 ", 64 ' 提示信息
11  Exit Sub
12  ElseIf strCommand = "TESTNO" Then
13  MsgBox "用户 " & Text5.Text & " 不在线 ", 64 ' 提示信息
14  Exit Sub
15  ElseIf Left(strCommand, 8) = "MSGFORM:" Then
16  getMsg = Right(strCommand, Len(strCommand) - InStr(1, strCommand, "MSG:") - 3)
'显示消息
17  getForm = Mid(strCommand, 9, InStr(1, strCommand, "MSG:") - 9) ' 获取消息内容
18  If Right(getMsg, 3) = Chr(0) & "OK" Then
19  getMsg = Mid(getMsg, 1, Len(getMsg) - 3)
20  Msg1.Caption = "消息发送成功! " ' 提示信息
21  End If
22  MsgBox "来自 " & getForm & " 的消息 " & Time & vbCrLf &
***** & vbCrLf & vbCrLf & getMsg, vbSystemModal + vbMsgBoxSetForeground, "信息窗口" ' 号为
获取消息的分隔线
23  Exit Sub
24  ElseIf strCommand = Chr(0) & "OK" Then
25  Msg1.Caption = "消息发送成功! ": Exit Sub ' 提示消息发送成功
26  ElseIf Left(strCommand, 7) = "Server : " Then
27  Msg1.Caption = Right(strCommand, Len(strCommand) - 7): Exit Sub
28  ElseIf strCommand = CMD1 Then ' 关机
29  W1.SendData QuitMsg ' 断开连接
30  DoEvents
31  Call QuitSystemWin(Me.hwnd, 0) ' 调用函数进行关机
32  ElseIf strCommand = CMD2 Then ' 重启
33  W1.SendData QuitMsg ' 断开连接
34  DoEvents
35  Call QuitSystemWin(Me.hwnd, 1) ' 调用函数进行重启
36  Else ' 不处理任何事件
37  End If
38  Exit Sub
39  err:
40  End Sub

```

(10) 在代码窗口中编写 Winsock 控件的 Error 事件的代码, 用于处理连接中的错误 (代码 23-1-20.txt)。

```

01 Private Sub W1_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long,
ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
02 On Error Resume Next ' 发现错误暂不处理
03 Select Case Number
04 Case 10061 ' 判断错误类别号
05 Msg1.Caption = Description & ", 可能是服务端已关闭或正忙, 请稍后再试! "
'提示错误信息
06 Case 11001 ' 判断错误类别号
07 Msg1.Caption = "在网络中找不到主机 " & Text5.Text & ", 请稍后重试! " ' 提示错误信息

```

```

08 Case 10065 '判断错误类别号
09 Msg1.Caption = "没有与主机相连的路由器，不能建立连接！" '提示错误信息
10 Case Else '判断错误类别号
11 Msg1.Caption = "发生了错误，错误代号是" & Number & "，" & Description & "，" & Scode
    '提示错误信息
12 End Select
13 W1.Close '关闭连接
14 Command2.Enabled = False '不允许检测 IP
15 Command3.Enabled = False '不允许发送消息
16 Command1.Caption = "登录服务器" '显示信息
17 Command1.Enabled = True '按钮控件可用
18 End Sub
    
```

常数	值	含义
SckClosed	0	缺省值，表示关闭
SckOpen	1	打开
SckListening	2	正在监听连接
sckConnectionPending	3	连接请求已抵达，但尚未完成
sckResolvingHost	4	主机名正在解析
sckHostResolved	5	主机名解析已完成
sckConnecting	6	连接请求已经启动，但尚未完成
sckConnected	7	连接建立（完成）
sckClosing	8	对方正在初始化一次“关闭”
sckError	9	产生某个错误



#### 提示

Winsock 的控件状态有以下几种：SckClosed、SckOpen、SckListening、sckConnectionPending、sckResolvingHost、sckHostResolved、sckConnecting、sckConnected、sckClosing、sckError，其值分别为从 0 到 9，上表中说明了各个状态和含义。在各种考试中，通常会在一个案例中考察多个状态，分清每个状态的具体含义是答对这类题型的基础。

## 23.3 运行系统

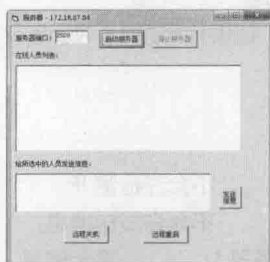


本节视频教学录像：12 分钟

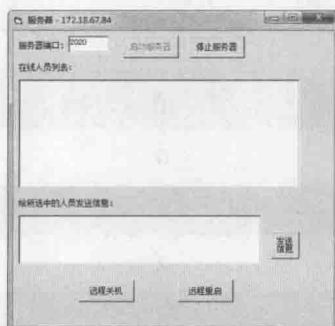
经过上述的 7 个步骤，完成服务器端和客户端的程序设计。为了检验程序的正误，需要选择两台没

有安装 Visual Basic 6.0 的机器进行试验,一台可以做服务器端又可以做客户端,另一台仅做客户端。

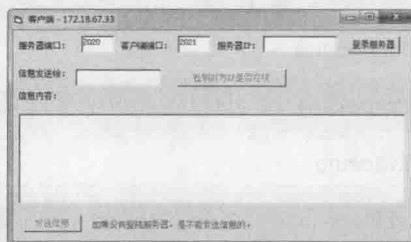
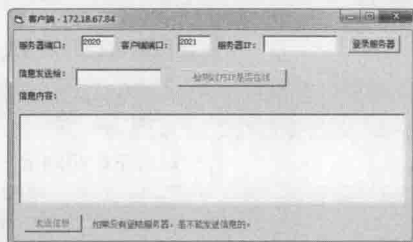
(1) 按【F5】快捷键运行服务器端程序。



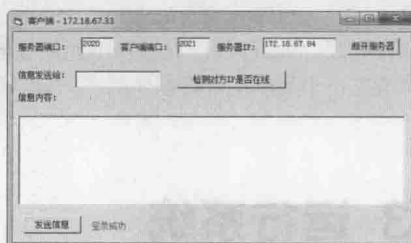
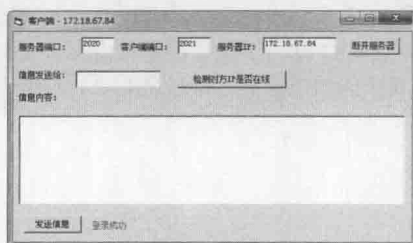
(2) 单击【启动服务器】按钮,开启服务器(服务器端 IP: 172.18.67.84)。



(3) 按【F5】快捷键运行客户端程序(客户端以 IP: 172.18.67.84 和 IP: 192.168.16.6 为例来进行演示)。

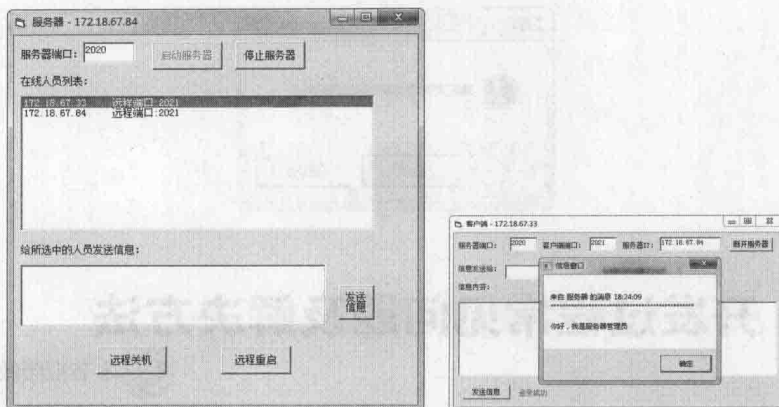


(4) 在客户端输入服务器的 IP 地址,单击【登录服务器】按钮连接服务器。

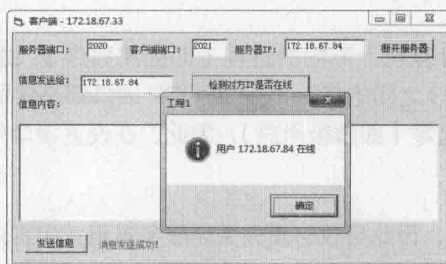


(5) 在服务器端输入信息内容,并选择 IP 为 192.168.16.6 的客户端,单击【发送信息】按钮。

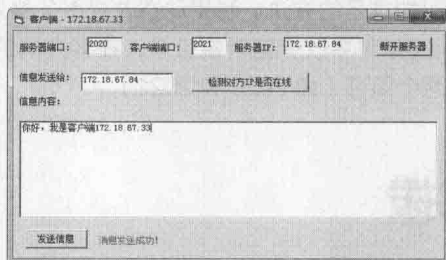
(6) 客户端(IP 为 192.168.16.6)会接收到服务器所发出的信息。



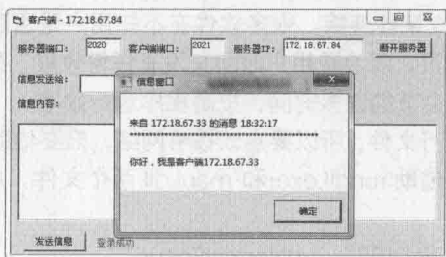
(7) 如果客户端 (IP 为 192.168.16.6) 向其他客户端发送信息, 需要先检测看其他客户端的 IP 是否连接至服务器, 如果没有连接上, 是不可以发送信息的。



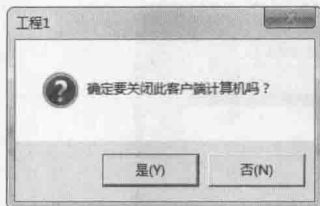
(8) 在客户端 (IP 为 192.168.16.6) 中输入信息内容, 单击【发送信息】按钮。




(9) 客户端 (IP 为 172.18.67.84) 便可以收到客户端 (IP 为 192.168.16.6) 所发来的消息。



(10) 在服务器端选择客户端的 IP 地址, 单击【远程关机】按钮, 即可关闭客户端的计算机。



## 23.4 开发过程常见问题及解决方法

 本节视频教学录像: 23 分钟

在程序开发过程中, 总会有或多或少的问题出现在我们的面前。切不要被这些问题所吓倒, 这些问题的出现都会成为我们以后编程的经验财富。

### 1. 端口号不可以随便设置

端口号是用于表示同一台计算机上不同的进程 (即应用程序)。由于系统自动分配的端口号位于 1024 ~ 5000 之间, 而 1 ~ 1023 之间的任一 TCP 或 UDP 端口都是保留的, 系统不允许任一进程使用保留端口, 除非其有效用户 ID 是零 (即超级用户), 因此, 在设置端口号的时候应尽量超过 1024。

### 2. 无法正常连接服务器

由于设置了客户端的端口号, 因此导致必须先关闭服务器再关闭客户端才能在下次正常连接; 或者客户端异常退出时 (比如客户端突然停电), 也会导致下次无法正常连接。出现这种错误的原因, 是因为没有释放连接端口号所造成的。为此可以将服务器端停止服务, 再打开服务; 也可以利用动态改变服务器监听端口号和客户端呼叫端口号的方法。利用动态改变服务器监听端口号和客户端呼叫端口号的方法是: 在服务器的 Form\_Load 事件中改变监听端口号, 在客户端的 Winsock 控件的错误事件中改变呼叫端口号, 端口号只要使用两个即可 (如 2020 和 2021)。

## 23.5 高手点拨

 本节视频教学录像: 9 分钟

许多应用程序常常需要在程序中直接进行联网操作, 以便进行一些必要的处理 (如在线注册和在线帮助等), 这就要求在程序中建立某些连接。很多软件在不知用户是否联网的情况下就启动浏览器查找网址, 结果只能查出一个错误网页, 既浪费用户时间又没有任何效果。如果应用程序在查找网页之前能自动判断用户是否已经联网, 就会节约许多时间, 提高程序运行效率。

由于拨号网络不是一个可执行文件, 所以要启动拨号网络, 需要借助 explorer.exe。但若是要启动拨号网络中的某一个连接, 则要借助 rundll.exe 和 mui.dll 两个文件。启动方法如下 (假定此连接名称为 163)。

```
Shell "rundll mui.dll, Rndial 163", vbNormalFocus
```

上面假定了连接名称, 但在实际编程中通常是不知道连接名称的。在窗体上放置一个命令按钮 (cmdCallConnect), 在其单击事件中进行连接处理。

# 第24章



本章视频教学录像：8 分钟

## 图形图像应用开发——仿 Windows 画图程序

本章讲解在 Visual Basic 6.0 中如何开发一个仿 Windows 画图程序，通过这个画图程序可以完成一些基本的绘图操作。


本章的内容包含常用图形图像控件的应用、坐标系和颜色的设置，以及常用的绘图方法。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握常用图形图像控件的应用
- ☐ 熟悉坐标系
- ☐ 熟悉颜色设置
- ☐ 掌握常用的绘图方法



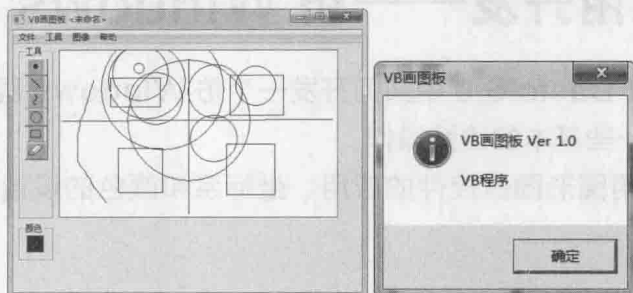
## 24.1 系统分析

 本节视频教学录像: 1 分钟


本章通过一个“VB 画图板”程序的设计来学习图形图像的应用开发。系统主要功能分析如下。

- (1) 在程序界面中有一个画图板，可以在上面画点、直线、线条、圆形和矩形等形状。
- (2) 可以将绘制的图形用橡皮工具擦除。
- (3) 可以新建、打开和保存图形。
- (4) 可以选取使用不同的颜色。
- (5) 可以水平、垂直翻转图像，可以拉伸图像。
- (6) 提供一个用于显示软件信息的对话框。

程序的效果如图所示。




## 24.2 系统设计

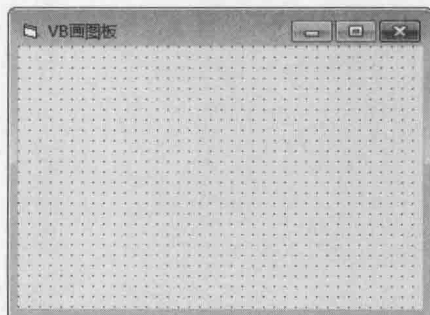
 本节视频教学录像: 3 分钟

从上一节程序的功能和效果图图中可以看到，在该程序中包含有主窗体、软件信息窗体、若干个按钮控件以及相关的操作菜单等。本节介绍具体的设计步骤。

### 第 1 步：主窗体设计

(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

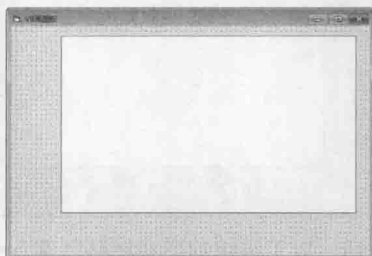
(2) 将 Form1 的名称设置为“frmDraw”，Caption 属性设置为“VB 画图板”，并调整大小。



## 第 2 步：图像框控件设计

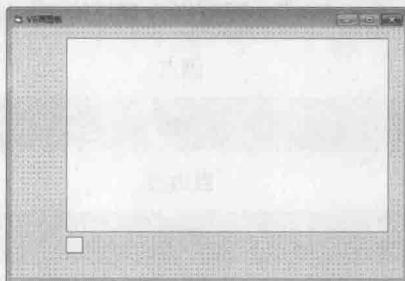
(1) 在窗体中添加 1 个图像框控件，并按下表设置其属性。

属性名称	属性值
名称	picDraw
Appearance	0 - Flat
AutoRedraw	True
ScaleMode	3 - Pixel



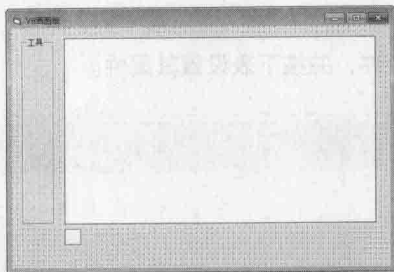
(2) 在窗体中再添 1 个图像框控件，并按下表设置其属性。

属性名称	属性值
名称	picTool
Appearance	0 - Flat
AutoRedraw	True
Visible	False

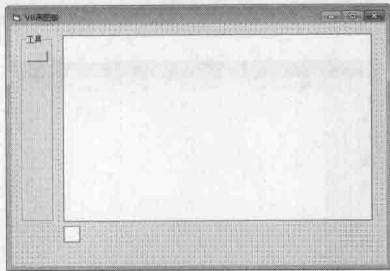


## 第 3 步：工具控件设计

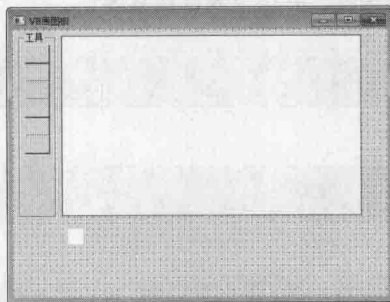
(1) 在窗体中添加 1 个框架控件，将其 Caption 属性值设置为“工具”，调整其大小和位置如下图所示。



(2) 在窗体中添加 1 个命令按钮，将其名称设置为“cmdTool”，Caption 属性值设置为空，Style 属性值设置为“1 - Graphical”。




(3) 复制添加的 cmdTool 命令按钮，以工程组的形式粘贴创建 5 个命令按钮，按照下图排列整齐。



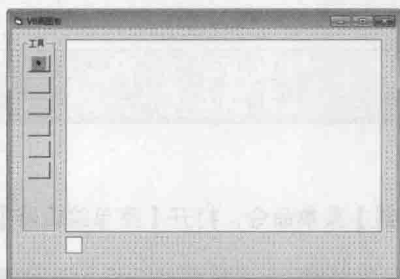
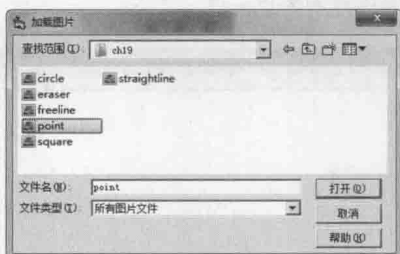
(4) 按照下表设置 cmdTool 命令按钮的属性值。

命令按钮名称	ToolTipText 属性值	控件功能
cmdTool(0)	画点	选择画点工具
cmdTool(1)	直线	选择直线工具
cmdTool(2)	自由线	选择自由线工具
cmdTool(3)	圆形	选择圆形工具
cmdTool(4)	矩形	选择矩形工具
cmdTool(5)	橡皮擦	选择橡皮擦工具

(5) 单击选中 cmdTool(0) 命令按钮，单击属性窗口中 Picture 属性值后面的  按钮。

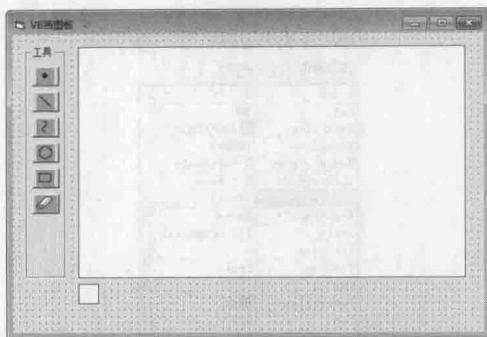


(6) 弹出【加载图片】对话框，选择随书光盘中的“Sample\ch24\point.bmp”图片文件，然后单击【确定】按钮。

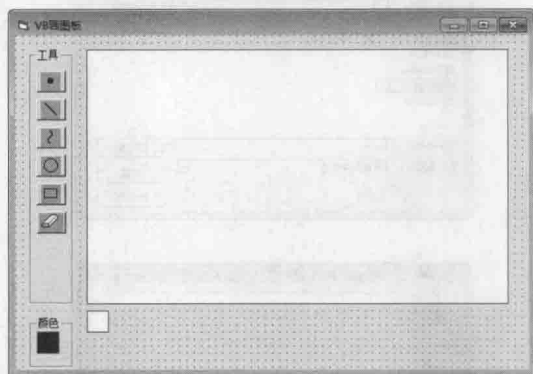


(7) 按照上述方法，参照下表为其他按钮添加图片。

控件名称	图片	添加后按钮
cmdTool(1)	Sample\ch24\straightline.bmp	
cmdTool(2)	Sample\ch24\freeline.bmp	
cmdTool(3)	Sample\ch24\circle.bmp	
cmdTool(4)	Sample\ch24\square.bmp	
cmdTool(5)	Sample\ch24\eraser.bmp	

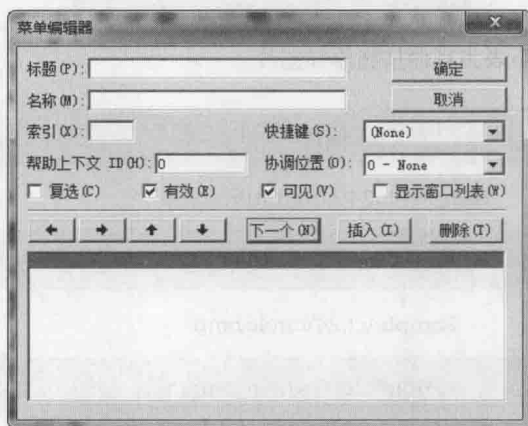


(8) 在窗体中添加 1 个框架控件，将其名称设置为“FrameColor”，Caption 属性值设置为“颜色”。在 FrameColor 框架控件中添加 1 个图像框控件，将其名称设置为“picColor”，Appearance 属性值设置为“0 - Flat”，BackColor 属性值设置为黑色。



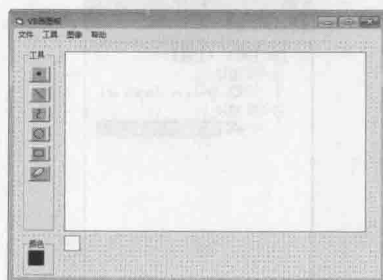
#### 第 4 步：编辑菜单

(1) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。



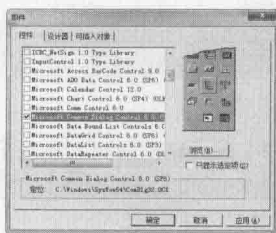
(2) 按下表为窗体添加菜单，标题前面带有“...”的为上一级的子菜单，添加时只需输入后面文字即可。

标题	名称	索引	菜单作用
文件	mnuFile		文件菜单
…新建	mnuNew		新建文件
…打开	mnuOpen		打开文件
…保存	mnuSave		保存文件
…另存为	mnuSaveAs		另存为文件
…退出	mnuExit		退出程序
工具	mnuTools		工具菜单
…画点	mnuTool	0	画点工具
…直线	mnuTool	1	直线工具
…自由线	mnuTool	2	自由线工具
…圆形	mnuTool	3	圆形工具
…矩形	mnuTool	4	矩形工具
…橡皮擦	mnuTool	5	橡皮擦工具
图像	mnuPicture		图像菜单
…水平翻转	mnuHVer		水平翻转
…垂直翻转	mnuVer		垂直翻转
…图像拉伸	mnuScale		图像拉伸
…清除图像	mnuCls		清除图像
帮助	mnuHelp		帮助菜单
…关于	mnuAbout		显示关于对话框

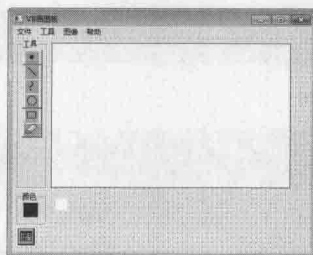
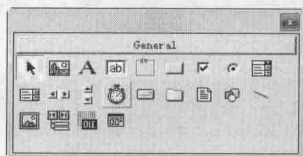


第 5 步：添加部件和模块

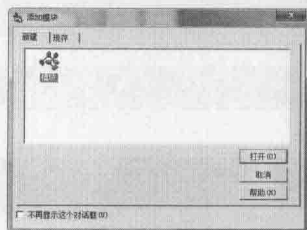
(1) 选择【工程】>【部件】菜单命令，在弹出的【部件】对话框中选择【控件】选项卡，选中【Microsoft Common Dialog Control 6.0】复选框，然后单击【确定】按钮。



(2) 双击刚添加的 CommonDialog 控件，把它添加到窗体中。



(3) 选择【工程】>【添加模块】菜单命令，在弹出的【添加模块】对话框中选择【新建】选项卡，选中“模块”图标，单击【打开】按钮。

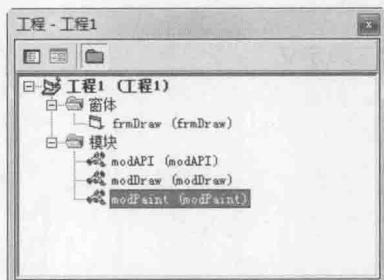




(4) 在【属性】窗口中将其名称改为“modAPI”。



(5) 按照上述方法再添加两个模块，将其名称分别设置为“modDraw”和“modPaint”。



#### 提示

图像框 (Image) 控件和图片框 (PictureBox) 控件的区别是：① 图像框占用内存少。② 图片框内可以包括其他控件，图像框则不能。③ 装入图片框的图形文件不随图片框的大小尺寸调整大小，当 AutoSize=true 时，图片框可以自己调整大小以适应图片文件。④ 图像框有一个 Stretch 属性，当其为 true 时，图形能自动变化大小以适应图像框的尺寸。

#### 第 6 步：添加模块代码

(1) 双击 modAPI 模块，在代码窗口中添加以下代码（代码 24-1.txt）。

```
01 Option Explicit
02 Public Declare Function ExtFloodFill Lib "gdi32" _ '定义填充区域函数
03 (ByVal hdc As Long, ByVal X As Long, ByVal Y As Long, _
04  ByVal crColor As Long, ByVal wFillType As Long) As Long
05 Public Const SRCCOPY = &HCC0020
06 Public Const FLOODFILLSURFACE = 1
07 Public Const DSTINVERT = &H550009
08 Public Const PATPAINT = &HFB0A09
```

(2) 双击 modPaint 模块，在代码窗口中添加以下代码（代码 24-2.txt）。

```
01 Option Explicit
02 Public gDragPen& '定义全局变量
```

```
03 Public Const strTitle = "VB 画图板"
04 Public Const strVersion = "1.0"
05 Public Const strNewFile = "未命名"
06 Public lngLColor&
07 Public lngOutColor&
08 Public intFillStyle As Integer
09 Public CurentWidth As Single
10 Public blnModified As Boolean
11 Public strFilename As String
12 Public mImage As ImageProp
13 Public Type ImageProp
14     iWidth As Long
15     iHeight As Long
16     iBackColor As OLE_COLOR
17 End Type
18 Public Type UDT_Tool '工具定义
19     Line As Boolean
20     FreeLine As Boolean
21     Circle As Boolean
22     Point As Boolean
23     Square As Boolean
24     Eraser As Boolean
25 End Type
26 Public Type UDT_Square '矩形定义
27     mintSquareX1 As Single
28     mintSquareX2 As Single
29     mintSquareY1 As Single
30     mintSquareY2 As Single
31     blnFill As Boolean
32 End Type
33 Public Type UDT_Line '线定义
34     mintLineX1 As Single
35     mintLineX2 As Single
36     mintLineY1 As Single
37     mintLineY2 As Single
38 End Type
39 Public Type UDT_Circle '圆形定义
40     mintCircleX1 As Single
41     mintCircleY1 As Single
42     mintCircleX2 As Single
43     mintCircleY2 As Single
44     mdblCircleR As Single
45     mblnFill As Boolean
```

## 46 End Type

(3) 双击 modDraw 模块，在代码窗口中添加以下代码（代码 24-3.txt）。

```
01 Option Explicit
02 Private CurentX As Single ' 当前坐标 x 值
03 Private CurentY As Single ' 当前坐标 y 值
04 Private mudtLine As UDT_Line ' 画线方法定义
05 Private mudtCircle As UDT_Circle ' 画圆方法定义
06 Private mudtSquare As UDT_Square ' 画矩形方法定义
07 Public Sub InitiateLine(mudtP As UDT_Point, frmName As Form) ' 初始化画线方法
08     With mudtLine
09         .mintLineX1 = mudtP.msglX
10         .mintLineY1 = mudtP.msglY
11         .mintLineX2 = mudtP.msglX
12         .mintLineY2 = mudtP.msglY
13     End With
14 End Sub
15 Public Sub DrawLine(mudtP As UDT_Point, frmName As Form) ' 执行画线
16     With mudtLine
17         gDragPen& = frmName.picDraw.BackColor Xor QBColor(0)
18         frmName.picDraw.DrawMode = 7 'to enable live drawing
19         frmName.picDraw.Line (.mintLineX1, .mintLineY1)-(.mintLineX2, .mintLineY2), gDragPen&
20         .mintLineX2 = mudtP.msglX
21         .mintLineY2 = mudtP.msglY
22         frmName.picDraw.Line (.mintLineX1, .mintLineY1)-(.mintLineX2, .mintLineY2), gDragPen&
23     End With
24 End Sub
25 Public Sub FinalizeLine(mudtP As UDT_Point, frmName As Form) ' 生成线
26     With mudtLine
27         .mintLineX2 = mudtP.msglX
28         .mintLineY2 = mudtP.msglY
29         blnModified = True
30         frmName.picDraw.Line (.mintLineX1, .mintLineY1)-(.mintLineX2, .mintLineY2), lngOutColor&
31     End With
32 End Sub
```

第 7 步：添加窗体及菜单命令代码

(1) 双击 frmDraw 窗体空白处，在弹出的代码窗口中添加以下代码（代码 24-4.txt）。

```
01 Private Sub Form_Load()
02     frmDraw.MousePointer = vbDefault ' 鼠标指针为默认形状
03     strFilename = strNewFile
```

```

04 frmDraw.Caption = strTitle & " <" & strFilename & ">"
05 CurentWidth = 1
06 picDraw.DrawWidth = CurentWidth
07 lngLColor& = vbBlack
08 picDraw.FillColor = vbBlack
09 picDraw.FillStyle = vbSolid
10 blnModified = False ' 编辑改动布尔值初始为 False
11 End Sub

```

(2) 单击【文件】>【新建】菜单命令，在弹出的代码窗口中添加以下代码（代码 24-5.txt）。

```

01 Private Sub mnuNew_Click()
02     If blnModified Then
03         Select Case MsgBox(" 当前是否保存?", vbInformation + vbYesNoCancel, "新建画图")
04             Case vbYes ' 按下“是”按钮，保存图片
05                 If strFilename <> strNewFile Then
06                     SavePicture picDraw.Image, strFilename
07                 Else
08                     mnuSaveAs_Click
09                 End If
10             Case vbNo ' 按下“否”按钮，清除图片
11                 picDraw.Cls
12                 frmDraw.Caption = strTitle & " <" & strNewFile & ">"
13             Case vbCancel ' 按下“取消”按钮，结束过程
14                 End Select
15         End If
16     End Sub

```

(3) 单击【文件】>【打开】菜单命令，在弹出的代码窗口中添加以下代码（代码 24-6.txt）。

```

01 Private Sub mnuOpen_Click() ' 打开按钮单击事件
02     On Error GoTo Err ' 如果出错转到这里
03     CommonDialog1.Filter = "Bitmap" & "(*.bmp)|*.bmp|Jpeg Files (*.jpg)|*.jpg"
04     CommonDialog1.FilterIndex = 1
05     CommonDialog1.ShowOpen ' 显示打开对话框
06     CommonDialog1.CancelError = False
07     strFilename = CommonDialog1.FileName
08     picDraw.Picture = LoadPicture(strFilename) ' 加载图片到控件
09     frmDraw.Caption = strTitle & " <" & CommonDialog1.FileTitle & ">" ' 设置窗体标题
10 Exit Sub
11 Err:
12 End Sub

```

(4) 单击【文件】>【保存】菜单命令，在弹出的代码窗口中添加以下代码。

```

01 Private Sub mnuSave_Click()      ' 保存按钮单击事件
02     If strFilename <> strNewFile Then      ' 如果不是新文件
03         SavePicture picDraw.Image, strFilename      ' 直接保存
04     Else      ' 如果是新文件
05         mnuSaveAs_Click      ' 调用另存为过程
06     End If
07 End Sub

```

(5) 单击【文件】>【另存为】菜单命令，在弹出的代码窗口中添加以下代码（代码 24-7.txt）。

```

01 Private Sub mnuSaveAs_Click()      ' 另存为按钮单击事件
02     On Error GoTo Err      ' 如果出错转到这里
03     CommonDialog1.Filter = "Bitmap" & "(*.bmp)|*.bmp|Jpeg Files (*.jpg)|*.jpg"
04     CommonDialog1.FilterIndex = 1
05     CommonDialog1.ShowSave      ' 显示另存为对话框
06     SavePicture picDraw.Image, CommonDialog1.FileName
07     strFilename = CommonDialog1.FileName
08     frmDraw.Caption = strTitle & " " & "<" & CommonDialog1.FileTitle & ">"      ' 设置窗体标题
09 Exit Sub
10 Err:End Sub

```

(6) 单击【文件】>【退出】菜单命令，在弹出的代码窗口中添加以下代码（代码 24-8.txt）。

```

01 Private Sub mnuExit_Click()      ' 退出按钮单击事件
02     Dim intExitChoice As Integer
03     If blnModified Then      ' 如果有编辑改动，执行下面过程
04         intExitChoice = MsgBox(" 是否保存后退出 ?", vbYesNoCancel, " 退出画图板 ")
        ' 弹出对话框，询问是否保存
05         If intExitChoice = vbYes Then      ' 按下“是”按钮
06             mnuSave_Click      ' 调用保存
07             End
08         ElseIf intExitChoice = vbNo Then      ' 按下“否”按钮
09             End      ' 直接结束
10         Else
11             Exit Sub
12         End If
13     End If
14 End      ' 若没有编辑改动则直接退出
15 End Sub

```

(7) 单击【工具】>【画点】菜单命令，在弹出的代码窗口中添加以下代码。

```

01 Private Sub mnuTool_Click(Index As Integer)
02     cmdTool_Click Index

```

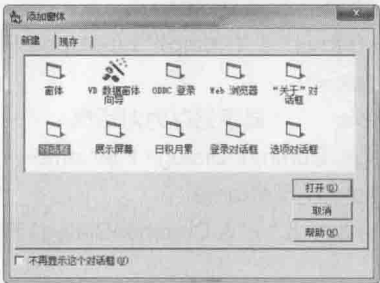
03 End Sub

(8) 单击【帮助】>【关于】菜单命令，在弹出的代码窗口中添加以下代码。

```
01 Private Sub mnuAbout_Click()  
02     MsgBox "VB 画图板 Ver 1.0 " & Chr(13) & Chr(13) & "VB 程序", vbInformation, "VB 画图板 "  
03 End Sub
```

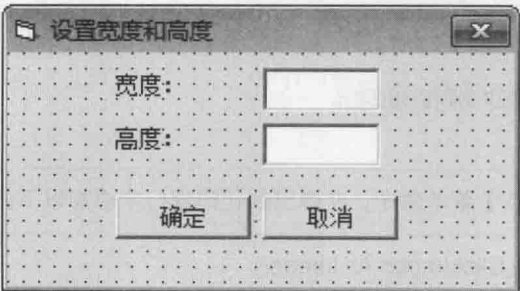
第 8 步：添加图像拉伸对话框及图像菜单命令代码

(1) 选择【工程】>【添加窗体】菜单命令，在弹出的【添加窗体】对话框的【新建】选项卡中选中“对话框”图标，单击【打开】按钮。



(2) 将添加的对话框窗体名称设置为“frmScale”，Caption 属性设置为“设置宽度和高度”，然后在窗体中添加两个标签控件和两个文本框控件，并按下表设置其命令按钮属性。

控件	名称	Caption	Text
Label1	LabelWidth	宽度：	
Label2	LabelHeight	高度：	
Text1	txtWidth		
Text2	txtHeight		
Command1	cmdOK	确定	
Command2	cmdCancel	取消	



(3) 双击【确定】按钮，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub cmdOK_Click()
02     mImage.iWidth = Val(txtWidth.Text)      'txtWidth 内容传给 mImage 的 iWidth 属性
03     mImage.iHeight = Val(txtHeight.Text)    'txtHeight 内容传给 iHeight 属性
04     Call ScaleImage(frmDraw.picDraw, frmDraw.picTool, mImage.iWidth, mImage.iHeight)
        '调用缩放图像过程
05     Unload frmScale      '卸载窗体
06 End Sub
    
```

---

(4) 双击【取消】按钮，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub cmdCancel_Click()      '取消按钮单击事件
02     Unload frmScale
03 End Sub
    
```

---

(5) 单击【图像】>【水平翻转】菜单命令，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub mnuHVer_Click()      '水平翻转菜单单击事件
02     Call FlipImage(picDraw, 0)    '调用翻转图像过程
03 End Sub
    
```

---

(6) 单击【图像】>【垂直翻转】菜单命令，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub mnuVer_Click()      '垂直翻转菜单单击事件
02     Call FlipImage(picDraw, 1)    '调用翻转图像过程
03 End Sub
    
```

---

(7) 单击【图像】>【图像拉伸】菜单命令，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub mnuScale_Click()      '图像拉伸菜单单击事件
02     frmScale.Show vbModal, frmDraw    '显示设置宽度和高度对话框
03 End Sub
    
```

---

(8) 单击【图像】>【清除图像】菜单命令，在弹出的代码窗口中添加以下代码。

---

```

01 Private Sub mnuCls_Click()      '清除图像菜单单击事件
02     Set picDraw.Picture = Nothing    '清除图像
03 End Sub
    
```

---

### 第 9 步：添加其他代码

在代码窗口中添加其他相关代码（代码 24-9.txt）。

---

```

01 Option Explicit
02 Private blnSquareFill As Boolean    '是否填充布尔值
03 Private blnCircleFill As Boolean    '是否填充布尔值
    
```

---



```

04 Private mudtTool As UDT_Tool
05 Private mudtPoint As UDT_Point
06 Private Sub Form_Unload(Cancel As Integer)
07     If blnModified Then '退出时, 如果有编辑改动则执行下面过程
08         Select Case MsgBox("是否保存后退出?", vbInformation + vbYesNoCancel, "退出画图板")
09             Case vbYes '按下“是”按钮
10                 If strFilename <> strNewFile Then '如果文件为当前文件, 直接保存
11                     SavePicture picDraw.Image, strFilename
12                 Else
13                     mnuSaveAs_Click '如果文件不是当前文件, 执行“另存为”命令
14                 End If
15             Case vbNo '按下“否”按钮, 不保存
16                 End
17             Case vbCancel '按下“取消”按钮, 取消退出
18                 Cancel = True
19             End Select
20         End If
21     End Sub
22 Private Sub picDraw_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
23     '按下鼠标, 执行画的过程
24     lngOutColor& = lngLColor&
25     With mudtPoint
26         .msglX = X
27         .msglY = Y
28     End With
29     With mudtTool '开始画
30         If .Line Then
31             InitiateLine mudtPoint, Me
32         ElseIf .Point Then
33             blnModified = True
34             picDraw.PSet (mudtPoint.msglX, mudtPoint.msglY), lngOutColor&
35         ElseIf .Circle Then
36             InitiateCircle mudtPoint, blnCircleFill, Me
37         ElseIf .Square Then
38             InitiateSquare mudtPoint, blnSquareFill, Me
39         ElseIf .Eraser Then
40             picDraw.PSet (X, Y), picDraw.BackColor
41         ElseIf .FreeLine Then
42             InitiateFreeLine mudtPoint, Me
43         End If
44     End With
45 End Sub

```

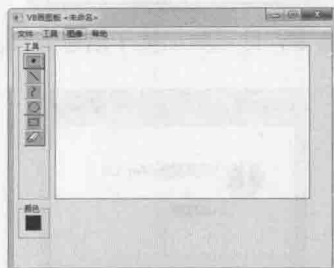
## 24.3 运行系统



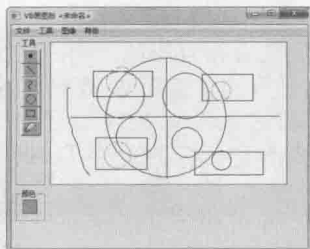
本节视频教学录像：2 分钟

所有工作都做好后，就可以开始测试自己一手设计的 VB 画图程序了。

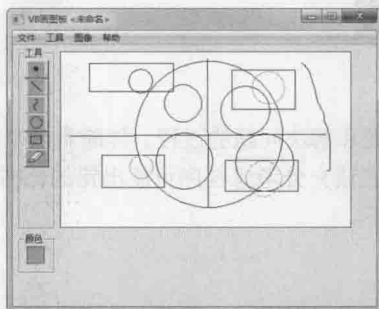
(1) 按【F5】快捷键运行程序。



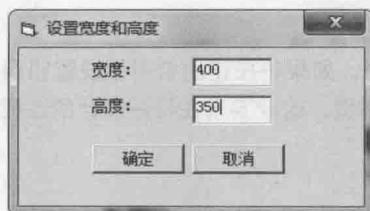
(2) 单击左侧的工具按钮，就可以在绘图区域绘制各种各样的图形。

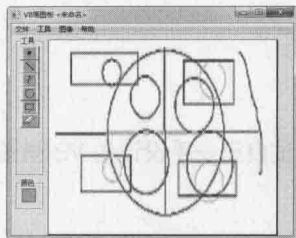


(3) 选择【图像】>【水平翻转】或【图像】>【垂直翻转】菜单命令，可以翻转图像。



(4) 选择【图像】>【图像拉伸】菜单命令，在弹出的【设置宽度和高度】对话框中输入【宽度】和【高度】值，然后单击【确定】按钮，即可拉伸图像。





(5) 选择【帮助】>【关于】菜单命令，即可弹出关于信息窗口。



#### 提示

定义坐标系和 ScaleMode 属性会相互影响。用户定义坐标系将使 ScaleMode 为 0。把 ScaleMode 属性设置为大于 0 的 z 值可恢复默认坐标系，同时改变 ScaleHeight 和 ScaleWidth 的度量单位，并将 ScaleLeft 和 ScaleTop 设置为 0，CurrentX 和 CurrentY 的值做相应改变以反映当前点的新坐标。不带参数的 Scale 方法恢复标准坐标系时，ScaleMode 值被自动修改为 1。

## 24.4 高手点拨



本节视频教学录像：2 分钟

软件开发的过程，就是发现问题和解决问题的过程。伴随着越来越多问题的解决，程序会变得越来越健壮。一个优秀的软件设计人员应该充分考虑程序可能出错的地方，并逐一排查。在设计 VB 画图板的过程中，容易出现常见问题如下。

### 1. 代码书写不规范

程序的运行对代码的书写规范要求很严格，一个小小的错误都可能导致程序出错，有时就算程序能运行，但结果却不是预期的那样。所以在书写每一个步骤的代码时，都应仔细规范，严格区分大小写、中英文标点符号等的使用。

### 2. 控件设置不当

在设计 VB 画图板的工具按钮时，如果将按钮的索引值设置错误，就会导致执行错误的命令。这类错误程序不会报告出错，所以不易察觉，这就要求在程序设计的过程中一定要细心谨慎，应认真地完成每一个步骤。

# 第25章



本章视频教学录像：10 分钟

## 多媒体应用开发——开发自己的播放器

多媒体技术是指把图片、文字、声音、动画、影像等多种媒体的信息通过计算机进行数字化加工处理，集成为一个具有交互性系统的一种技术，其特点是具有集成性、交互性、数字化、实时性。本章将带您开发一个属于自己的音乐播放器。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握 Windows Media Player 控件的使用方法
- ☐ 了解播放器制作的原理
- ☐ 能够制作音乐播放器

## 25.1 系统分析

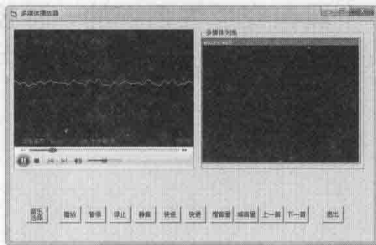


本节视频教学录像: 3 分钟

多媒体技术是当今信息技术领域中发展最快, 也是最为活跃的技术。目前, 已呈现出多样化的发展, 在实际中几乎随处可以见到它的身影。本章使用 Visual Basic 6.0 下的 Windows Media Player 控件制作一个多媒体播放器, 它不仅拥有简洁的播放界面和完善的播放功能, 同时在使用占有的内存也不多。

在第 19 章, 我们学习过使用 MCI 控件多媒体控制, 本章学习的 Windows Media Player 控件和 MCI 相比的优势在于 Windows Media Player 控件的功能要更强大一些, 不仅可以播放 AVI、MIDI 和 WAV 格式的文件, 还可以播放 MCI 控件无法播放的 MPEG 和 MOV 等多媒体格式。另外, 用 Windows Media Player 控件播放动画文件时, 还可以显示当前模仿时间和播放帧。

本章所要实现的多媒体播放器是一个可以播放目前主流媒体格式的多媒体播放器, 其具有媒体文件的选择、媒体的播放、音量大小的调节、播放进度的选择、暂停和停止等基本功能, 同时还可以实现自动循环播放、静音等实用功能。



## 25.2 系统设计

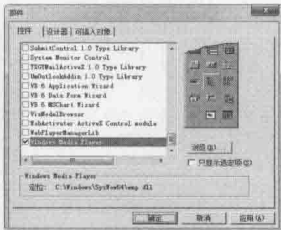
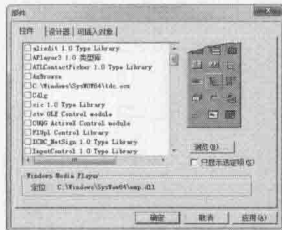


本节视频教学录像: 3 分钟

对播放器系统有了简单的了解之后, 本节具体讲述如何开发自己的播放器。

### 第 1 步: 添加部件

- (1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单击【打开】按钮。
- (2) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令, 弹出【部件】对话框。
- (3) 在【控件】选项卡中选择【Windows Media Player】复选框, 然后单击【确定】按钮。
- (4) 这样就将 Windows Media Player 控件添加到了工具箱中。



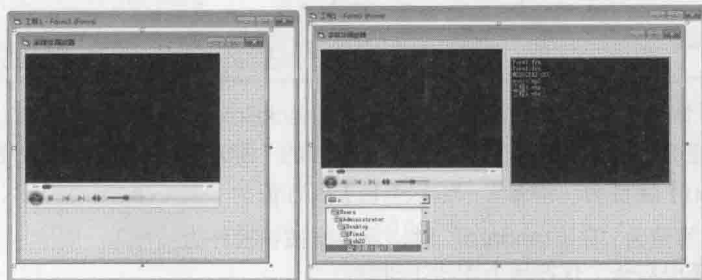
## 第 2 步：界面设计

(1) 将 Form1 窗体的 Caption 属性设置为“多媒体播放器”。

(2) 在 Form1 窗体中添加 1 个播放器 (WindowsMediaPlayer) 控件，保持默认属性。

(3) 在 Form1 窗体中添加 1 个标签 (Label) 控件，并将其【Caption】属性设置为空。

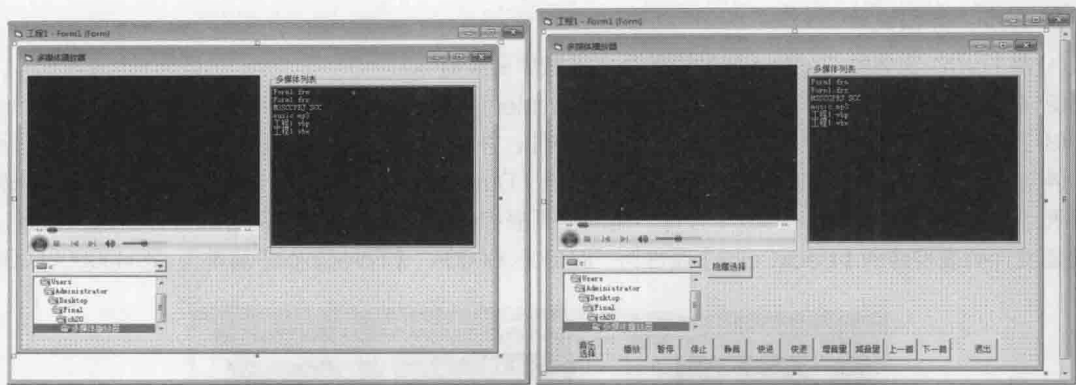
(4) 在 Form1 窗体中添加 1 个驱动器列表 (DriveListBox) 控件、1 个目录列表 (DirListBox) 控件和 1 个文件列表 (FileListBox) 控件，驱动器列表控件和目录列表控件的各属性保持默认设置，然后设置文件列表控件的【BackColor】属性为“&H00000000&”，【ForeColor】属性为“&H0000FF00&”。





(5) 在 Form1 窗体中添加 1 个框架 (Frame) 控件，用于显示播放列表，设置其【Caption】属性为“多媒体列表”，将其置于文件列表控件的底层。

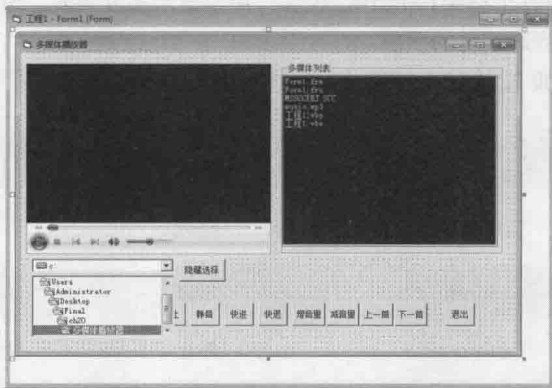
(6) 在 Form1 窗体中添加 12 个命令按钮 (CommandButton) 控件，按下表分别设置其属性。

控件名称	名称	Caption	控件作用
Command1	Command1	播放	用于播放多媒体文件
Command2	Command2	暂停	用于暂停多媒体文件的播放
Command3	Command3	停止	用于停止多媒体文件的播放
Command4	Command4	增音量	用于增加多媒体文件的音量
Command5	Command5	减音量	用于减小多媒体文件的音量
Command6	Command6	隐藏选择	用于隐藏多媒体文件的选择路径
Command7	Command7	音乐选择	用于显示多媒体文件的选择路径
Command8	Command8	退出	用于退出系统
Command9	Command9	上一首	用于播放当前多媒体文件的上一首
Command10	Command10	下一首	用于播放当前多媒体文件的下一首
Command11	Command11	快进	用于多媒体文件的向前快播
Command12	Command12	快退	用于多媒体文件的向后回播



(7) 在 Form1 窗体中添加 1 个复选框按钮 (CheckBox) 控件, 设置其【Style】属性值为 “1-Graphical”, 这里 Style 是设置控件的外观, 是标准的 (标准 Windows 风格) 还是图形的 (带有自定义图片), 【Caption】属性为 “静音”。当【Style】属性值为 “1-Graphical” 时, 复选框按钮控件为  , 当【Style】属性值为 “0-Standard” 时, 复选框按钮控件为  。

(8) 在 Form1 窗体中对各个控件按下图所示排列。



提示

在后面所编写的代码中, 会对 Form1 窗体中的部分控件进行相关的控制。

第 3 步: 编写程序代码

(1) 在 Form1 窗体上用右键单击, 在弹出的快捷菜单中选择【查看代码】选项, 进入代码窗口, 输入以下代码。

```
Public volum As Integer      ' 定义声音全局变量, 用于记录播放过程中的声音值
Public bool As Boolean       ' 定义全局变量
```

(2) 在 Form1 窗体的空白处双击, 在打开的代码窗口中输入以下代码 (代码 25-1.txt)。

```
01 Private Sub Form_Load()
02   If File1.ListCount > 0 Then      ' 列表中存在音乐文件, 各按钮控件可以操作
```



```

03 Command1.Enabled = False '【播放】按钮不可用
04 Command2.Enabled = True '【暂停】按钮可用
05 Command3.Enabled = True '【停止】按钮可用
06 Command4.Enabled = True '【增音量】按钮可用
07 Command5.Enabled = True '【减音量】按钮可用
08 Command9.Enabled = True '【上一首】按钮可用
09 Command10.Enabled = True '【下一首】按钮可用
10 Command11.Enabled = True '【快进】按钮可用
11 Command12.Enabled = True '【快退】按钮可用
12 Check1.Enabled = True '【静音】按钮可用
13 Command6.Visible = False '【隐藏选择】按钮不可见
14 Drive1.Visible = False '【驱动器列表】控件不可见
15 Dir1.Visible = False '【目录列表】控件不可见
16 File1.Pattern = "*.mp3;*.MP3;*.wma" '仅显示指定类型的音乐
17 File1.ListIndex = 0 '如果多媒体列表中有音乐，则第 1 首音乐处于选中状态
18 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName '播放第 1 首音乐
19 volum = WindowsMediaPlayer1.settings.volume '声音值
20 WindowsMediaPlayer1.Controls.play
21 Else '如果启动窗体后，没有多媒体列表
22 Command1.Enabled = False '【播放】按钮不可用
23 Command2.Enabled = False '【暂停】按钮不可用
24 Command3.Enabled = False '【停止】按钮不可用
25 Command4.Enabled = False '【增音量】按钮不可用
26 Command5.Enabled = False '【减音量】按钮不可用
27 Command6.Visible = False '【隐藏选择】按钮不可见
28 Command9.Enabled = False '【上一首】按钮不可用
29 Command10.Enabled = False '【下一首】按钮不可用
30 Command11.Enabled = False '【快进】按钮不可用
31 Command12.Enabled = False '【快退】按钮不可用
32 Check1.Enabled = False '【静音】按钮不可用
33 Drive1.Visible = False '【驱动器列表】控件不可见
34 Dir1.Visible = False '【目录列表】控件不可见
35 File1.Pattern = "*.mp3;*.MP3;*.wma" '仅显示指定类型的音乐
36 End If
37 End Sub

```



**提示**

例子中的 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName 是一个相对路径，表示当前目录下的文件，即你的工程所保存的位置。你也可以使用绝对路径，如：WindowsMediaPlayer1.URL = "C:\music.wma"。

(3) 在 Form1 窗体中双击驱动器列表控件，在打开的代码窗口中输入以下代码。

```

01 Private Sub Drive1_Change()
02   Dir1.Path = Drive1.Drive ' 将驱动器列表中选中的当前驱动器赋给目录列表的路径
03   File1.Pattern = "*.mp3;*.MP3;*.wma" ' 仅显示指定类型的音乐
04 End Sub

```

(4) 在 Form1 窗体中双击目录列表框控件, 在打开的代码窗口中输入以下代码 (代码 25-2.txt)。

```

01 Private Sub Dir1_Change()
02   File1.Path = Dir1.Path ' 将文件列表框的路径设置为目录列表框所选中的路径
03   If File1.ListIndex < 0 Then ' 判断当前播放列表是否小于零
04     Exit Sub
05   Else
06     Command2.Enabled = True ' 【暂停】按钮可用
07     Command3.Enabled = True ' 【停止】按钮可用
08     Command4.Enabled = True ' 【增音量】按钮可用
09     Command5.Enabled = True ' 【减音量】按钮可用
10     Command9.Enabled = True ' 【上一首】按钮可用
11     Command10.Enabled = True ' 【下一首】按钮可用
12     Command11.Enabled = True ' 【快进】按钮可用
13     Command12.Enabled = True ' 【快退】按钮可用
14     Check1.Enabled = True ' 【静音】按钮可用
15   End If
16 End Sub

```

(5) 在 Form1 窗体中双击文件列表框控件, 在打开的代码窗口中选择【DbClick】事件, 并输入以下代码 (代码 25-3.txt)。

```

01 Private Sub File1_DbClick()
02   WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName ' 播放路径
03   volumn = WindowsMediaPlayer1.settings.volume ' 声音值
04   WindowsMediaPlayer1.Controls.play ' 播放
05   Command2.Enabled = True ' 【暂停】按钮可用
06   Command3.Enabled = True ' 【停止】按钮可用
07   Command4.Enabled = True ' 【增音量】按钮可用
08   Command5.Enabled = True ' 【减音量】按钮可用
09   Command9.Enabled = True ' 【上一首】按钮可用
10   Command10.Enabled = True ' 【下一首】按钮可用
11   Command11.Enabled = True ' 【快进】按钮可用
12   Command12.Enabled = True ' 【快退】按钮可用
13   Check1.Enabled = True ' 【静音】按钮可用
14 End Sub

```

(6) 在 Form1 窗体中双击 WindowsMediaPlayer 控件, 在打开的代码窗口中选择【StatusChange】事件, 并输入以下代码。

---

```

01 Private Sub WindowsMediaPlayer1_StatusChange() ' 循环播放实现
02   On Error Resume Next ' 发现错误暂不处理
03   If bool = False Then
04     WindowsMediaPlayer1.Controls.play ' 继续播放
05   End If
06 End Sub

```

---

#### 第 4 步：编写各命令按钮控件代码

(1) 在 Form1 窗体中双击【音乐选择】按钮控件，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command7_Click()
02   Drive1.Visible = True ' 【驱动器列表】控件可见
03   Dir1.Visible = True ' 【目录列表】控件可见
04   Command6.Visible = True ' 【隐藏选择】按钮可见
05   Command7.Enabled = False ' 【音乐选择】按钮不可见
06 End Sub

```

---

(2) 在 Form1 窗体中双击【播放】按钮控件，在打开的代码窗口中输入以下代码。

---

```

01 Private Sub Command1_Click()
02   WindowsMediaPlayer1.Controls.play ' 允许播放
03   Command2.Enabled = True ' 【暂停】按钮可用
04   Command3.Enabled = True ' 【停止】按钮可用
05 End Sub

```

---

(3) 在 Form1 窗体中双击【暂停】按钮控件，在打开的代码窗口中输入以下代码（代码 25-4.txt）。

---

```

01 Private Sub Command2_Click()
02   bool = True
03   WindowsMediaPlayer1.Controls.pause ' 暂停
04   Command2.Enabled = False ' 【暂停】按钮不可用
05   Command3.Enabled = True ' 【停止】按钮可用
06   Command1.Enabled = True ' 【播放】按钮可用
07 End Sub

```

---

(4) 在 Form1 窗体中双击【停止】按钮控件，在打开的代码窗口中输入以下代码（代码 25-5.txt）。

---

```

01 Private Sub Command3_Click()
02   bool = True
03   WindowsMediaPlayer1.Controls.stop ' 停止
04   Command3.Enabled = False ' 【停止】按钮不可用
05   Command1.Enabled = True ' 【播放】按钮可用
06   Command2.Enabled = True ' 【暂停】按钮可用
07 End Sub

```

---

(5) 在 Form1 窗体中双击【静音】复选框控件，在打开的代码窗口中输入以下代码（代码 25-6.txt）。

```
01 Private Sub Check1_Click()
02     If Check1.Value = 0 Then '如果没有单击该按钮
03         WindowsMediaPlayer1.settings.mute = False      '不暂停
04     Else
05         WindowsMediaPlayer1.settings.mute = True        '暂停
06     End If
07 End Sub
```

(6) 在 Form1 窗体中分别双击【快进】和【快退】按钮控件，在打开的代码窗口中输入以下代码（代码 25-7.txt）。

```
01 Private Sub Command11_Click() '快进
02     Command1.Enabled = True    '【播放】按钮可用
03     WindowsMediaPlayer1.Controls.fastForward '快进
04 End Sub
05 Private Sub Command12_Click() '快退
06     Command1.Enabled = True    '【播放】按钮可用
07     WindowsMediaPlayer1.Controls.fastReverse '快退
08 End Sub
```

(7) 在 Form1 窗体中分别双击【增音量】和【减音量】按钮控件，在打开的代码窗口中输入以下代码（代码 25-8.txt）。

```
01 Private Sub Command4_Click() '增大音量
02     WindowsMediaPlayer1.settings.volume = WindowsMediaPlayer1.settings.volume + 5
    '音量值增 5 个单位
03     volum = WindowsMediaPlayer1.settings.volume '获取新的音量
04 End Sub
05 Private Sub Command5_Click() '减小音量
06     WindowsMediaPlayer1.settings.volume = WindowsMediaPlayer1.settings.volume - 5
    '音量值减 5 个单位
07     volum = WindowsMediaPlayer1.settings.volume '获取新的音量
08 End Sub
```

(8) 在 Form1 窗体中分别双击【上一首】和【下一首】按钮控件，在打开的代码窗口中输入以下代码（代码 25-9.txt）。

```
01 Private Sub Command9_Click() '上一首
02     If File1.ListIndex = 0 Then '如果当前多媒体列表为第 1 行
03         File1.ListIndex = File1.ListCount - 1 '列表变为最后一行显示
```

```

04 WindowsMediaPlayer1.Controls.previous '执行上一首操作
05 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName '播放路径
06 WindowsMediaPlayer1.Controls.play '播放
07 Else
08 File1.ListIndex = File1.ListIndex - 1 '选择上一首
09 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName '播放路径
10 volum = WindowsMediaPlayer1.settings.volume '声音值
11 WindowsMediaPlayer1.Controls.play '播放
12 End If
13 End Sub
14 Private Sub Command10_Click() '下一首
15 If File1.ListIndex = File1.ListCount - 1 Then '如果为最后一行
16 File1.ListIndex = 0 '多媒体列表中第 1 行呈亮显
17 WindowsMediaPlayer1.Controls.Next '进行下一首操作
18 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName '播放路径
19 WindowsMediaPlayer1.Controls.play '播放音乐
20 Else
21 File1.ListIndex = File1.ListIndex + 1 '当前多媒体列表向下进 1
22 WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName '播放路径
23 volum = WindowsMediaPlayer1.settings.volume '声音值
24 WindowsMediaPlayer1.Controls.play '播放
25 End If
26 End Sub

```

(9) 在 Form1 窗体中双击【隐藏选择】按钮控件，在打开的代码窗口中输入以下代码。(代码 25-10.txt)

```

01 Private Sub Command6_Click()
02 Drive1.Visible = False '【驱动器列表】控件不可见
03 Dir1.Visible = False '【目录列表】控件不可见
04 Command6.Visible = False '【隐藏选择】按钮不可见
05 Command7.Enabled = True '【音乐选择】按钮可见
06 End Sub

```

(10) 在 Form1 窗体中双击【退出】按钮控件，在打开的代码窗口中输入以下代码。

```

01 Private Sub Command8_Click() '退出程序
02 Unload Me '卸载窗体
03 End Sub

```

## 25.3 运行系统

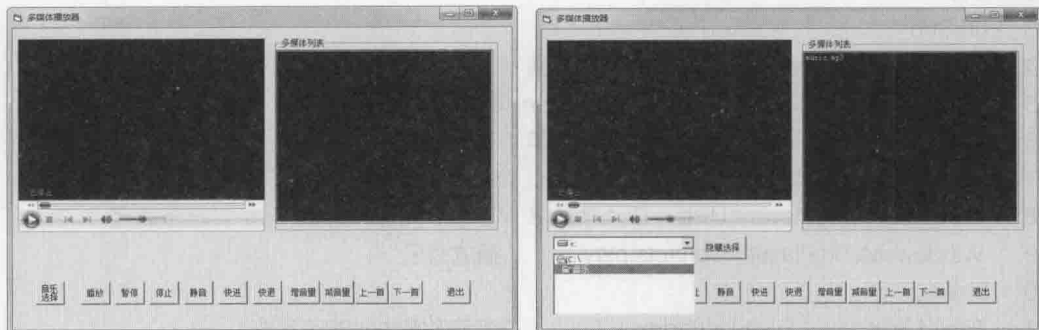


本节视频教学录像: 1 分钟

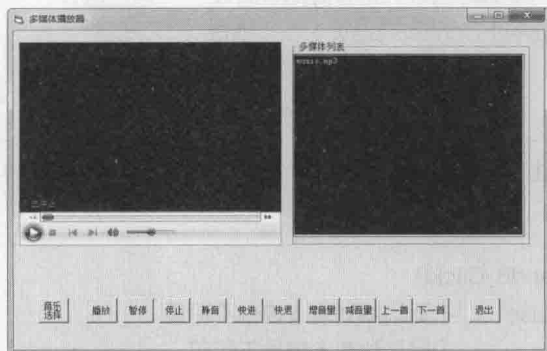
对系统设计好以后, 就可以使用自己编写的播放器听音乐了。

(1) 按【F5】快捷键, 启动程序。

(2) 单击【音乐选择】按钮, 在弹出的文件路径中选择音乐所在的文件。

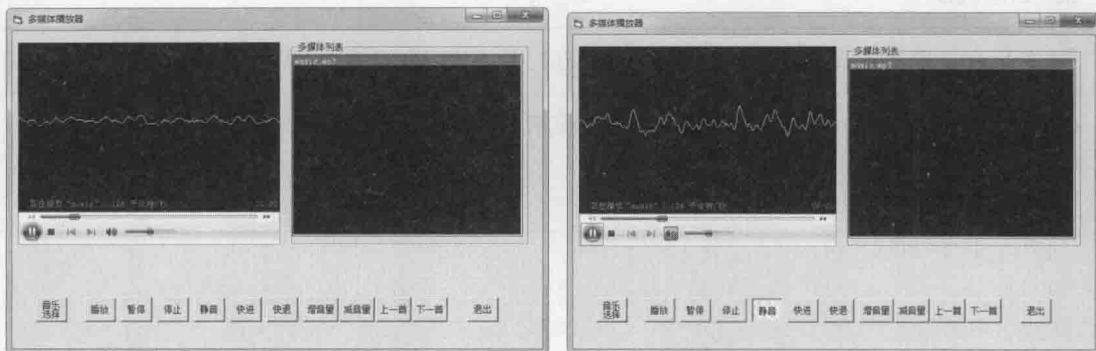


(3) 选择好音乐文件夹后, 可以单击【隐藏选择】按钮, 将弹出的文件路径隐藏起来。



(4) 任意双击【多媒体列表】中的文件, 即可播放音乐。

(5) 根据需要可以单击下方的命令按钮, 进行音乐播放的控制。



**提示**

如果当前文件路径中存在音乐文件，音乐文件则会在【多媒体列表】中显示，并且首次运行程序时，如果【多媒体列表】显示存在音乐文件，【多媒体列表】中的第 1 首音乐就会自动播放，同时，下方不可使用的按钮会变为可以使用。

## 25.4 开发过程常见问题及解决方法



本节视频教学录像：2 分钟

程序会在不断的修改中完善，由最初的简单功能逐渐变得面面俱到。然而由于个人思维方式的不同，对程序所设计的要求也就有所差别。下面说明一下在程序编写与测试的过程中所遇到的问题及解决的方法。

### 1. 进行【上一首】与【下一首】播放时，【多媒体列表】不能同步显示

根据 WindowsMediaPlayer 控件的 Controls.previous 属性和 Controls.Next 属性，可以很容易实现上一首与下一首歌曲的播放。这里以实现【上一首】为例，代码如下。

```
WindowsMediaPlayer1.Controls.previous      '进行上一首操作
WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName      '播放路径
WindowsMediaPlayer1.Controls.play          '播放
```

然而，在程序测试时，却发现播放中的音乐实现了上一首与下一首的转换播放，而【多媒体列表】中所显示的音乐文件名却没有改变。若要实现【多媒体列表】中文件名与实际播放的音乐文件同步，只需要将上面的代码进行如下的修改即可。

```
File1.ListIndex = File1.ListCount - 1      '列表变为最后一行显示
WindowsMediaPlayer1.Controls.previous      '进行上一首操作
WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName      '播放路径
WindowsMediaPlayer1.Controls.play          '播放
```

这段代码主要是对【上一首】按钮而言的，其编码思想为：让【多媒体列表】中的当前选择项先进行减 1 操作，即向前移动，显示上一首音乐的歌曲名，然后进行音乐的向前操作，同时获取新的音乐播放名，并对音乐进行播放。

### 2. 循环显示并播放【多媒体列表】中的音乐文件

播放的当前文件与【多媒体列表】中的文件名可以同步执行后却发现，当向上或向下执行文件时，到【多媒体列表】中的第 1 个音乐文件时，不能再进行【上一首】按钮的操作，或者是到【多媒体列表】中的最后一个音乐文件时，不能再进行【下一首】按钮的操作。为了解决这个问题，可以采取音乐文件循环播放的模式。也就是说，当【多媒体列表】中的音乐文件到第 1 个后，如果再单击【上一首】按钮时，就让其返回最后一个。同理，当【多媒体列表】中的音乐文件到最后一个后，如果再单击【下一首】按钮时，就让其返回第 1 个。这里以【上一首】按钮为例，其代码如下。

```
Private Sub Command9_Click() ' 上一首
```



```

If File1.ListIndex = 0 Then      ' 如果当前列表为第 1 行
    File1.ListIndex = File1.ListCount - 1 ' 列表变为最后一行显示
    WindowsMediaPlayer1.Controls.previous ' 进行上一首操作
    WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName ' 播放路径
    WindowsMediaPlayer1.Controls.play ' 播放
Else
    File1.ListIndex = File1.ListIndex - 1 ' 选择上一首
    WindowsMediaPlayer1.URL = Dir1.Path & "\" & File1.FileName ' 播放路径
    volume = WindowsMediaPlayer1.settings.volume ' 声音值
    WindowsMediaPlayer1.Controls.play ' 播放
End If
End Sub

```

这段代码的编码思想为：先判断当前播放的是不是第 1 首音乐，如果是就获取当前【多媒体列表】中文件的个数，并进行减 1 操作，使其最后一个音乐文件处于显示状态，然后读取最后一行音乐文件的文件名并播放；如果不是第 1 行的音乐文件，那么进行正常情况下的【上一首】按钮的操作即可。

## 25.5 高手点拨



本节视频教学录像：1 分钟

通过本章的学习，我们了解了 Windows Media Player 控件中的一些基本功能的使用方法，对于一些不是必需的功能，在这里给以说明，有兴趣的同学可以在自己的播放器上添加上这些功能，完成有个性的多媒体播放器。

fullScreen : Boolean 型，是否全屏显示。

wmp.settings.autoStart : Boolean 型，是否自动播放。

wmp.settings.mute : Boolean 型，是否静音。

wmp.settings.playCount : integer 型，播放次数。

wmp.currentMedia.duration : double 型，当前媒体总长度。

wmp.currentMedia.durationString : string 型，当前媒体总长度，字符串格式，如“03:25”。

wmp.currentMedia.getItemInfo : string 型，获取当前媒体信息。“Title”= 媒体标题，“Author”= 艺术家，“Copyright”= 版权信息，“Description”= 媒体内容描述，“Duration”= 持续时间（秒），“FileSize”= 文件大小，“FileType”= 文件类型，“sourceURL”= 原始地址。

wmp.currentMedia.setItemInfo : string 型，通过属性名设置媒体信息。

wmp.currentMedia.name : string 型，同 currentMedia.getItemInfo("Title")。

wmp.currentPlaylist.count : integer 型，当前播放列表所包含媒体数。

# 第26章



本章视频教学录像：9 分钟


## 文件系统应用开发——文件分割与合并程序

在特殊条件下，根据实际需求，需要对文件进行分割与合并处理。例如，通过邮箱发送一些非常重要的大文件，为了能够顺利地发出，需要对文件进行分割后发出，待对方收到被分割的文件后，再通过文件的合并，将那些被分割零碎的文件重新恢复至原始文件的状态，这样不仅成功地发送出了文件，也使得文件的安全性得到了保障。

### 本章要点（已掌握的在方框中打钩）

- ☐ 了解文件分割与合并的原理
- ☐ 熟悉文件分割与合并的编程过程
- ☒ 实现文件的分割与合并

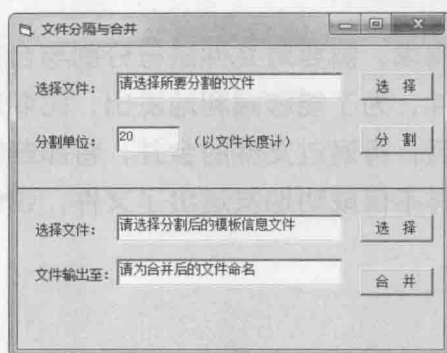
## 26.1 系统分析

 本节视频教学录像: 2 分钟

在实际工作中,经常会通过邮箱发送一些非常重要的文件,可是由于邮箱中都限制了附件的大小,使得发送这些重要的大文件非常不方便。这时,就可以考虑对这些文件进行分割,然后将它们发送出去。收到被分割的文件后,再通过文件的合并,将被分割零碎的文件重新恢复至原始文件的状态。

文件分割,就是将一个大文件先拆分为若干个小文件,然后对这些小文件分别进行处理。文件之所以能够被分割,是与文件在计算机中的存储方式密切相关的。任何一种类型的文件都是以二进制的形式存储于介质(包括硬盘、软盘等外部存储器)之中的,文件在分割时,就是根据用户的设定,以二进制的格式进行读写、分割;如果需要将文件进行组合,再以二进制的格式把分割后的文件重新组合起来。这样就实现了对文件的分割和合并。

在对文件分割与合并的原理有了初步的了解之后,下面就来实现文件的分割与合并。



该程序主要实现以下功能。

(1) 对所选择的任何一个文件根据分割单位进行分割。在分割文件的过程中,文件分割信息会存储于 .tpl 文件中。


(2) 根据 .tpl 文件所存储的分割文件信息,可以对分割后的文件重新合并。



### 提示

.tpl 文件是 Smarty 里的模板文件,Smarty 是一个使用 PHP 写出来的 PHP 模板引擎,是目前业界最著名的 PHP 模板引擎之一。它分离了逻辑代码和外在的内容,提供了一种易于管理和使用的方法,用来将原本与 HTML 代码混杂在一起 PHP 代码逻辑分离。简单地讲,目的就是要使 PHP 程序员同美工分离,使程序员改变程序的逻辑内容不会影响到美工的页面设计,美工重新修改页面不会影响到程序的逻辑,这在多人合作的项目中显得尤为重要。

## 26.2 系统设计

 本节视频教学录像: 2 分钟

本节我们将具体讲述如何实现文件的分割与合并。

第 1 步: 添加部件

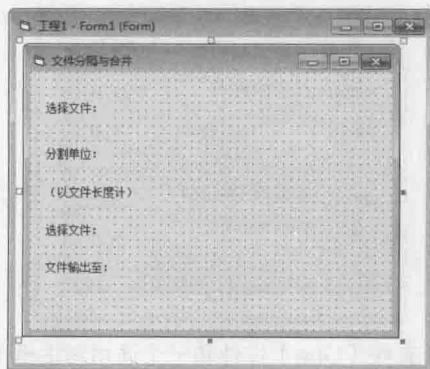
- (1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标，然后单击【打开】按钮。
- (2) 在 Visual Basic 6.0 界面中选择【工程】>【部件】菜单命令，弹出【部件】对话框。
- (3) 在【控件】选项卡中选择【Microsoft Common Dialog Control 6.0】复选框，然后单击【确定】按钮。
- (4) 这样就将 CommonDialog 控件添加到了工具箱中。



## 第 2 步：界面设计

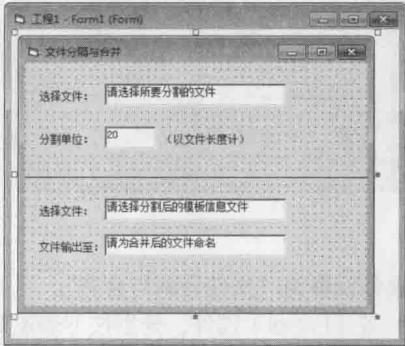
- (1) 将 Form1 窗体的 Caption 属性设置为“文件分割与合并”。
- (2) 在 Form1 窗体中添加 5 个标签 (Label) 控件，分别按下表设置其属性。

控件名称	名称	Caption	控件作用
Label1	Label1	选择文件：	提示记录中字段内容的意义
Label2	Label2	分割单位：	提示记录中字段内容的意义
Label3	Label3	(以文件长度计)	提示记录中字段内容的意义
Label4	Label4	选择文件：	提示记录中字段内容的意义
Label5	Label5	文件输出至：	提示记录中字段内容的意义



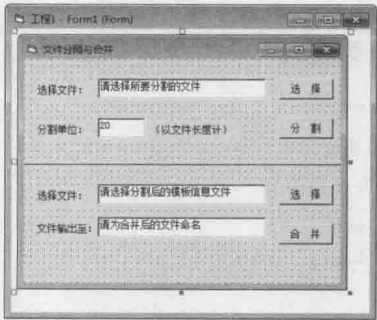
- (3) 在 Form1 窗体中添加 4 个文本框 (TextBox) 控件，分别按下表设置其属性。

控件名称	名称	Text	控件作用
TextBox	Text1	请选择所要分割的文件	显示所要分割文件的文件名
TextBox	Text2	20	显示所要分割文件的分割单位
TextBox	Text3	请选择分割后的模板信息文件	显示分割文件后的模板文件名
TextBox	Text4	请为合并后的文件命名	显示合并文件的名称

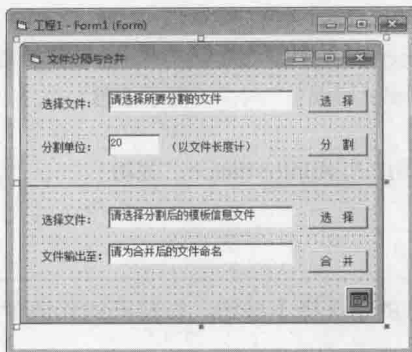


(4) 在 Form1 窗体中添加 4 个命令按钮（CommandButton）控件，分别按下表设置其属性。

控件名称	名称	Caption	控件作用
Command1	Command1	选 择	选择所要分割文件的文件
Command2	Command2	分 割	分割文件
Command3	Command3	选 择	选择分割文件后的模板文件
Command4	Command4	合 并	合并文件



(5) 在 Form1 窗体中添加 1 个直线（Line）控件和一个通用对话框（CommandDialog）控件，各控件按下图所示排列。



### 第 3 步：编写代码

(1) 在 Form1 窗体上用右键单击，在弹出的快捷菜单中选择【查看代码】菜单项，进入代码窗口，输入以下代码（代码 26-1.txt）。

```
01 Option Explicit
02 Private Type FileSection ' 自定义 FileSection 类型
03     Bytes() As Byte ' 函数返回一个字节型数组
04     FileLen As Long ' 定义文件长度变量
05 End Type
06 Private Type SectionedFile ' 自定义 SectionedFile 类型
07     Files() As FileSection ' 函数返回一个 FileSection 类型的数组
08     NumberOfFiles As Long ' 定义文件长度变量
09 End Type
10 Private Type FileInfo ' 自定义 FileInfo 类型
11     OrigProjSize As Long ' 定义文件分割大小变量
12     OrigFileName As String ' 定义文件分割名称变量
13     FileCount As Integer ' 定义文件分割总数变量
14     FileStartNum As Long ' 定义文件分割数量变量
15 End Type
```

(2) 在 Form1 窗体上双击上方的【选择】按钮，在打开的代码窗口中输入以下代码（代码 26-2.txt）。

```
01 Private Sub Command1_Click()
02     dlgOpen.Filter = "所有文件 (*.*)*.*" ' 选择文件类型
03     dlgOpen.ShowOpen ' 打开文件
04     If dlgOpen.FileName <> "" Then
05         Text1.Text = dlgOpen.FileName ' 记录文件名
06     End If
07 End Sub
```

(3) 在 Form1 窗体上双击【分割】按钮，在打开的代码窗口中输入以下代码（代码 26-3.txt）。

```
01 Private Sub Command2_Click()
```

```

02 Dim msg As String
03 If Not Split(Text1.Text, msg, CLng(Text2.Text)) Then '调用分割函数进行分割
04     MsgBox msg, vbCritical, " 错误 " '提示信息
05 Else
06     MsgBox " 文件分割成功! ", vbInformation, " 成功 " '提示信息
07 End If
08 End Sub

```

(4) 在 Form1 窗体上双击下方的【选择】按钮，在打开的代码窗口中输入以下代码（代码 26-4.txt）。

```

01 Private Sub Command3_Click()
02     dlgOpen.Filter = " 模板文件 (*.tpl)|*.tpl| 所有文件 (*.*)|*.* " '定义打开的文件类型
03     dlgOpen.ShowOpen '打开文件
04     If dlgOpen.FileName <> "" Then
05         Text3.Text = dlgOpen.FileName '获取文件名
06     End If
07 End Sub

```

(5) 在 Form1 窗体上双击【合并】按钮，在打开的代码窗口中输入以下代码（代码 26-5.txt）。

```

01 Private Sub Command4_Click()
02     If Not merge(Text3.Text, Text4.Text) Then '调用合并函数
03         MsgBox " 文件组合失败! ", vbCritical, " 错误 " '提示信息
04     Else
05         MsgBox " 文件组合成功! ", vbInformation, " 成功 " '提示信息
06     End If
07 End Sub

```

(6) 在 Form1 的代码窗口中，编写名为“Split”的文件分割函数（代码 26-6.txt）。

```

01 Public Function Split(strSplit As String, strErrMsg As String, Optional everPerUnit As Long =
1439865) As Boolean
02     On Error GoTo errHandler '发现错误去处理
03     Dim strSaveName As String, nFileNumber As Integer
04     Split = True '假设可以成功分割
05     Dim IFileLen As Long '定义变量，获取分割文件的长度
06     IFileLen = FileLen(strSplit) '获取分割文件的长度
07     If IFileLen <= everPerUnit + 1 Then '如果所要分割的文件总长度小于等于分割单位
08         Split = False '分割失败
09         strErrMsg = " 文件太小，很难分割! " '提示信息
10         Exit Function '退出函数体
11     End If
12     nFileNumber = FreeFile '使用 FreeFile 函数取得可用的文件号

```



```

13  Open strSplit For Binary As nFileNumber ' 以二进制的方式打开文件
14  Dim IFileNum As Long ' 定义被分割之后的子文件个数
15  If CInt(IFileLen / everPerUnit) >= IFileLen / everPerUnit Then ' 分割划分
16      IFileNum = CInt(IFileLen / everPerUnit) ' 获取分割总数
17  Else
18      IFileNum = CInt(IFileLen / everPerUnit) + 1 ' 获取分割总数
19  End If
20  Dim CurrentFile As SectionedFile
21  ReDim CurrentFile.Files(1 To IFileNum) ' 定义分割后文件数组
22  Dim i As Long
23  For i = 1 To IFileNum - 1
24      ReDim CurrentFile.Files(i).Bytes(1 To everPerUnit) ' 定义文件分割大小
25      CurrentFile.Files(i).FileLen = UBound(CurrentFile.Files(i).Bytes) ' 获取文件分割长度
26  Next
27  For i = 1 To IFileNum
28      Get #nFileNumber, , CurrentFile.Files(i).Bytes ' 读取文件内容
29  Next
30  ReDim CurrentFile.Files(IFileNum).Bytes(1 To IFileLen - ((IFileNum - 1) * everPerUnit))
' 重新定义最后一个子文件的大小
31  CurrentFile.NumberOfFiles = IFileNum ' 读取最后一个子文件号
32  Get #nFileNumber, , CurrentFile.Files(IFileNum).Bytes ' 读取最后一个子文件的内容
33  CurrentFile.Files(IFileNum).FileLen = UBound(CurrentFile.Files(IFileNum).Bytes)
    ' 分割文件
34  Close #nFileNumber ' 关闭文件
35  For i = 1 To CurrentFile.NumberOfFiles ' 所取得的内容, 分别存储到各个子文件中
36      strSaveName = "分割" & Format(i - 1, "00#") & "-" & strSplit ' 分割后的文件命名方式
37      nFileNumber = FreeFile ' 设置分割文件编号
38      Open strSaveName For Binary As nFileNumber ' 建立分割文件名
39      Put #nFileNumber, 1, CurrentFile.Files(i) ' 建立分割后的文件
40      Close #nFileNumber ' 关闭文件操作
41  Next
42  Dim FileInfoFile As FileInfo ' 将文件分割信息存储到自定义的结构类型 FileInfoFile 中
43  FileInfoFile.FileCount = IFileNum
44  FileInfoFile.OrigFileName = strSplit
45  FileInfoFile.OrigProjSize = FileLen(strSplit)
46  FileInfoFile.FileStartNum = 0
47  strSaveName = strSplit & ".tpl" ' 将文件分割信息存入到模板文件中
48  nFileNumber = FreeFile
49  Open strSaveName For Binary As #nFileNumber ' 以二进制形式打开文件
50  Put #nFileNumber, , FileInfoFile ' 创建模板文件
51  Close #nFileNumber ' 关闭文件操作
52  Exit Function ' 退出函数体
53  errorHandler: ' 进行容错处理

```

```

54  strErrMsg = Err.Description ' 容错描述
55  Split = False
56  End Function

```

(7) 在 Form1 代码窗口中, 编写名为 “merge” 的文件分割函数 (代码 26-7.txt)。

```

01 Public Function merge(strFile As String, strOutFile As String) As Boolean
02     Dim FileInfo As FileInfo, File As SectionedFile, nFileNumber As Integer
03     merge = True ' 假设文件可以组合
04     nFileNumber = FreeFile ' 打开模板文件, 获取文件在分割时的记录
05     Open strFile For Binary As #nFileNumber ' 打开文件
06     Get #nFileNumber, , FileInfo ' 获取文件分割信息
07     Close #nFileNumber ' 关闭文件
08     ReDim File.Files(1 To FileInfo.FileCount) ' 重新定义文件分割数组
09     Dim strOpeNname As String
10     Dim i As Long
11     For i = 1 To FileInfo.FileCount ' 从每一个子文件中读取分割的数据内容
12         strOpeNname = " 分 割 " & Format((FileInfo.FileStartNum - 1 + i), "00#") & "-" & FileInfo.
OrigFileName ' 获取分割文件时的格式
13         nFileNumber = FreeFile ' 获取文件分割号
14         Open strOpeNname For Binary As #nFileNumber ' 以二进制形式打开分割后的文件内容
15         Get #nFileNumber, 1, File.Files(i) ' 获得分割后的文件内容
16         Close #nFileNumber ' 关闭文件分割
17     Next
18     nFileNumber = FreeFile ' 创建一个新文件, 并将合并后的文件数据放入其中
19     Open strOutFile For Binary As #nFileNumber ' 打开文件
20     For i = 1 To FileInfo.FileCount ' 开始创建文件
21         Put #nFileNumber, , File.Files(i).Bytes ' 读取子文件数据
22     Next
23     Close #nFileNumber ' 关闭文件
24 End Function

```

## 26.3 运行系统



本节视频教学录像: 2 分钟

当所有的代码都编写好后, 就可以运行系统, 实现文件的分割与合并了。

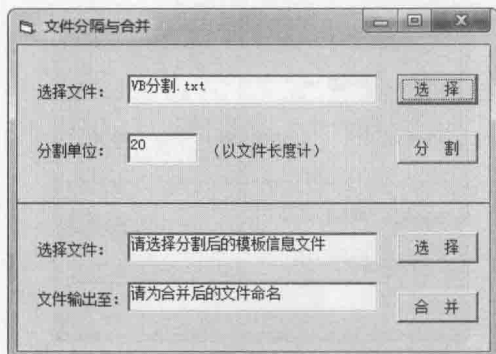
第 1 步: 分割与合并文本文件

(1) 按【F5】快捷键, 启动程序。

(2) 单击窗体上方的【选择】按钮, 在打开的【打开】对话框中, 选择所要分割的文本文件。



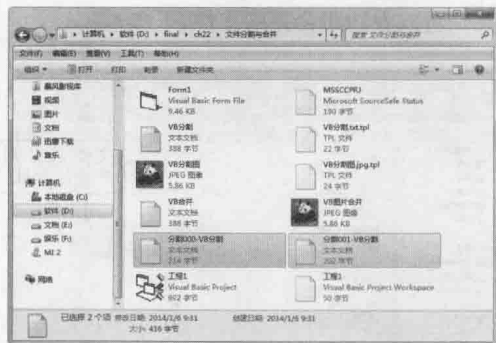
(3) 单击【打开】按钮，在上方的【选择文件】文本框中会显示所打开的文本文件名。



(4) 设置【分割单位】为“200”，单击【分割】按钮，如果分割成功，则会弹出【成功】对话框。



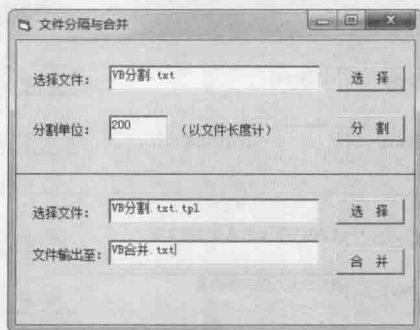
(5) 打开所分割文件的文件夹，可以看到所分割的文件。



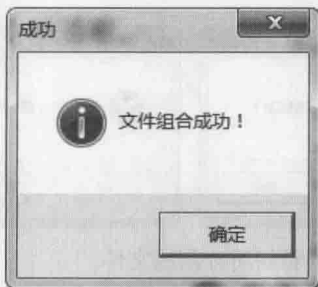
(6) 单击窗体下方的【选择】按钮，在打开的【打开】对话框中，选择分割文件时所存储的模板文件。



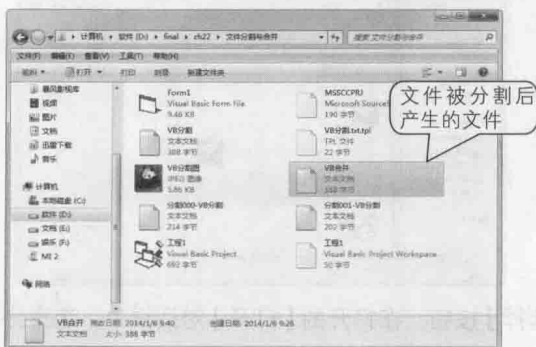
(7) 在【文件输出至】文本框中，输入所要合并的文本文件“VB 合并.txt”。



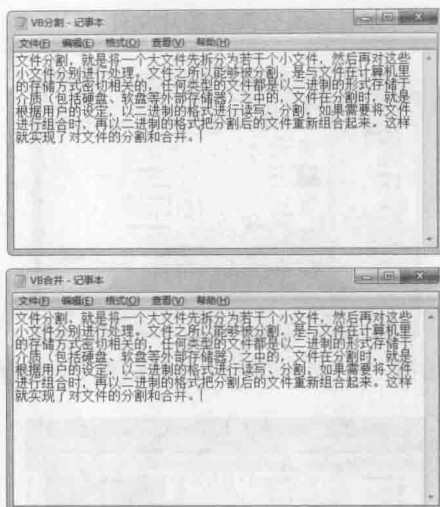
(8) 单击【合并】按钮，弹出【成功】对话框。



(9) 打开所分割文件的文件夹，可以看到所合并的文件。



(10) 打开原始文件，与分割后又合并的文件进行对比。

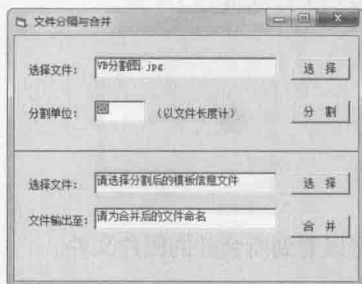


## 第 2 步：分割与合并图片文件

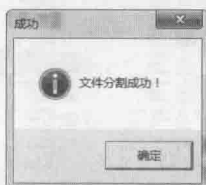
(1) 单击窗体上方的【选择】按钮，在打开的【打开】对话框中，选择所要分割的图片文件。



(2) 单击【打开】按钮，在上方的【选择文件】文本框中会显示所打开的文本文件名。

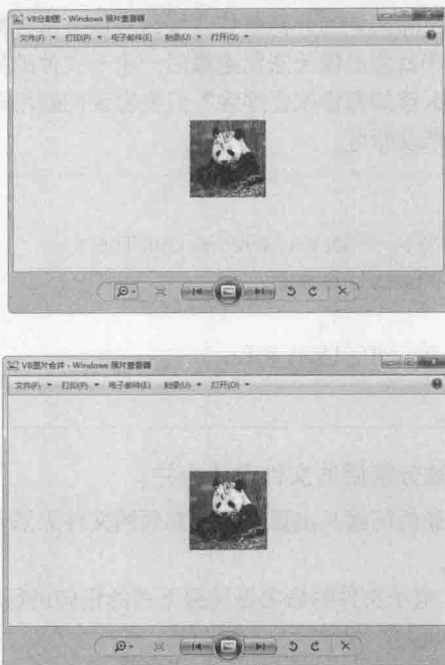


(3) 设置【分割单位】为“5000”，单击【分割】按钮，如果分割成功，则会弹出【成功】对话框。





(9) 打开原始文件，与分割后又合并的图片文件进行对比。



**注意**

【分割单位】应根据文件的实际需要设置。文件【分割单位】设置得越小，所分割的文件就会越多。如果【分割单位】设置得过大，就会弹出【错误】对话框，提示文件太小，无法按照【分割单位】进行分割。  
除部分文本文件外，被分割的文件在没有被合并前都无法独立运行。

## 26.4 开发过程常见问题及解决方法



本节视频教学录像：2 分钟

程序的开发并不是一帆风顺的，问题总会时不时地出现。无论问题是大小，总会有一个好的办法解决。

### 1. 文件分割个数的制定

在进行文件分割之时，子文件的个数是根据文件的长度和每个分割单位的比较得出的。在最初编写代码时是这样写的：

```
Dim IFileNum As Long
If CInt(IFileLen / everPerUnit) >= IFileLen / everPerUnit Then
IFileNum = CInt(IFileLen / everPerUnit)
End If
```



这样,初次看程序时,按照逻辑推理感觉也没有什么问题,可是在分割文件时,问题就出来了。如果被分割的文件长度,刚好可以整除文件的分割单位,那么可以正常分割。可是,如果被分割的文件长度不能整除文件的分割单位,程序就会出现无法创建最后一个子文件的错误,也就是说,在分割文件的过程中,会将部分数据丢失。那么该如何修改程序呢?只要发现问题所存在的根本原因,问题就很容易解决,将上面的代码进行如下的修改即可。

```
Dim IFileNum As Long
If CInt(IFileLen / everPerUnit) >= IFileLen / everPerUnit Then
    IFileNum = CInt(IFileLen / everPerUnit)
Else
    IFileNum = CInt(IFileLen / everPerUnit) + 1
End If
```

## 2. 子文件命名不统一,导致分割后的文件无法合并

文件可以正常分割了,但是新的问题又出现了:分割后的文件无法合并。究其原因,是因为在编写代码时,对子文件的命名不统一。

在分割文件的 Split 函数中,对子文件的命名是按照下面的语句进行的。

```
strSaveName = "分割" & Format(i - 1, "00#") & "-" & strSplit
```

而在合并文件的 merge 函数中,对子文件的命名是按照下面的语句进行的。

```
strOpeNname = "分割" & "-" & Format((FileInfo.FileStartNum - 1 + i), "00#") & FileInfo.OrigFileName
```

因此当程序合并文件时,就会出现无法找到正确的文件名而导致文件无法合并。为此将合并文件的 merge 函数中的代码进行如下的修改,就可以正常地合并文件了。

```
strOpeNname = "分割" & Format((FileInfo.FileStartNum - 1 + i), "00#") & "-" & FileInfo.OrigFileName
```

## 26.5 高手点拨



本节视频教学录像: 1 分钟

总之,利用 VB 编程,实现文件分割与组合简单易行,软件功能强大。尤其是使用双通道技术可以快速分割海量文件并在目的地迅速组合。该方法实现容易,界面操作简单,代码短小。当然,也可使用其他方法实现文件的分割与组合。例如在分割文件时,另外形成一个扩展名为 .bat 的批处理文件,这样在目的地,双击该文件可自动组合文件。

# 第27章



本章视频教学录像：8 分钟


## 游戏开发——VB 连连看

连连看玩法简单，新鲜有趣，是比较流行的游戏，你有没有想过自己开发一个连连看游戏。本章讲解如何利用 Visual Basic 6.0 开发一款简单的连连看小游戏，内容包含图像框控件的使用、数组的使用、程序结构的控制和资源文件的使用等。

### 本章要点（已掌握的在方框中打钩）

- ☐ 图像框控件
- ☐ 数组的使用
- ☐ 选择结构
- ☐ 循环结构
- ☐ 资源文件的使用
- ☐ 定时器控件的使用

## 27.1 系统分析

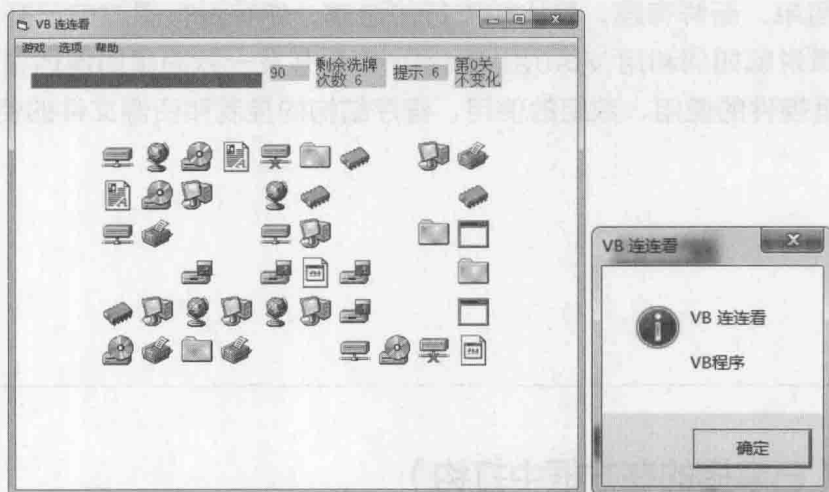
 本节视频教学录像: 2 分钟

连连看游戏相信大家玩过。在二维平面上排列着几排小图案, 用鼠标单击相同的两张图案, 如果图案之间的连线不超过两个拐点, 并且不穿过其他图案, 则单击的这两张图案就会消失。平面上的全部图案消失之时, 就是游戏胜利之时。


本章将制作出一个简单的连连看小游戏。下面简要地分析一下游戏的设计方法。

- (1) 主窗体容纳图像框、形状和计时器等控件。
- (2) 图像框控件用于显示游戏中的小图案。
- (3) 形状控件用于对游戏时间的消逝进行模拟。
- (4) 选择【提示】菜单命令, 则会提示当前可进行的操作, 并用线条绘制出来。
- (5) 选择【暂停】菜单命令, 可以暂停游戏, 再次单击则可恢复游戏。

程序的运行效果如图所示。




## 27.2 系统设计与开发

 本节视频教学录像: 3 分钟

在对本章所要设计的程序有了一定的了解后, 本节具体讲述如何对系统进行详细的设计和代码编写。

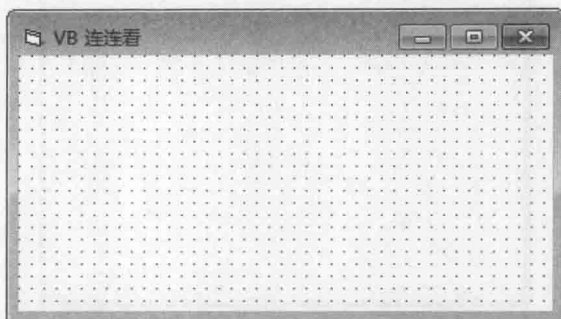
整个设计过程分为主窗体设计、编辑菜单、添加模块和资源、添加模块代码等部分。主窗体用于显示游戏操作界面和游戏相关信息, 菜单用于控制游戏运行, 模块和资源以及代码部分用于实现具体的游戏功能。

### 第 1 步: 主窗体设计

- (1) 启动 Visual Basic 6.0, 在弹出的【新建工程】对话框中选择【标准 EXE】图标 , 然后单

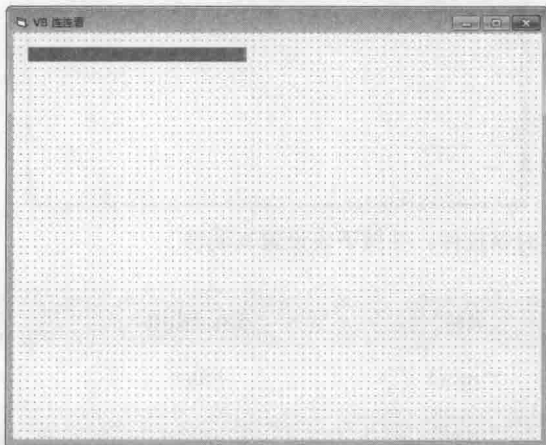
击【打开】按钮。

(2) 将 Form1 的名称改为“FormMain”，将 Caption 属性设置为“VB 连连看”。



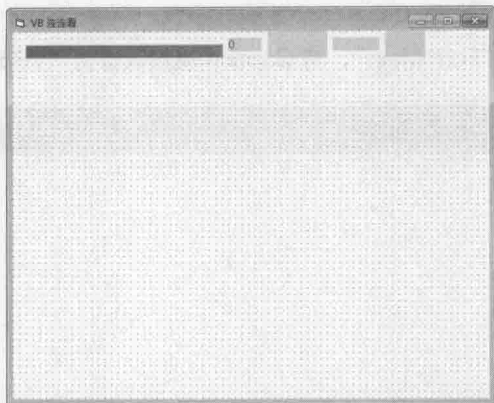
(3) 在窗体中添加两个形状控件，按下表设置其属性。

控件名称	名称	BackColor	BorderColor	BackStyle
Shape1	Shapeback	&H80000005&	&H00C0C0C0&	0 - Transparent
Shape2	Shapefront	&H000000FF&	&H000000C0&	1 - Opaque

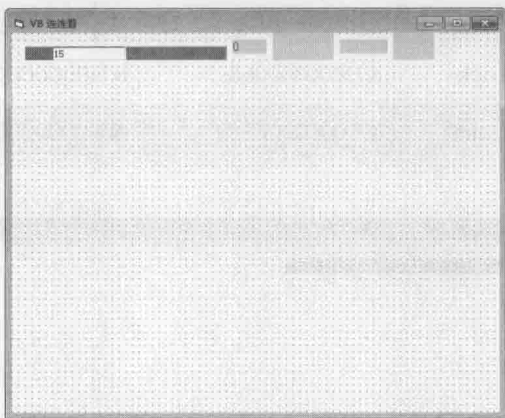


(4) 在窗体中添加 4 个标签控件，按下表设置其属性。

控件名称	名称	Caption	BackColor	控件作用
Label1	Labelscore	0	&H00E0E0E0&	显示游戏分数
Label2	Labelreset		&H00E0E0E0&	显示剩余洗牌次数
Label3	Labeltip		&H00E0E0E0&	显示剩余提示次数
Label4	Labelstageinfo		&H00E0E0E0&	显示当前关数信息

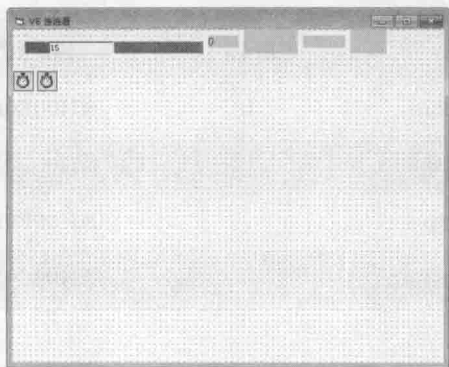


(5) 在窗体中添加 1 个文本框控件，将其 Text 属性设置为“15”，Visible 属性设置为“False”，将其叠放到形状控件 Shapeback 的下面，然后按下图所示排列各个控件。

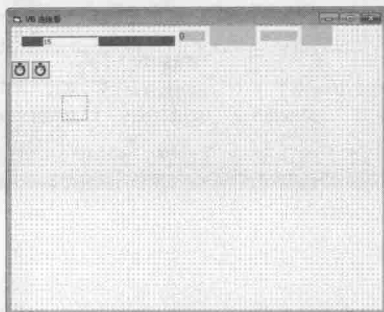


(6) 在窗体中添加两个计时器控件，并按下表设置其属性。

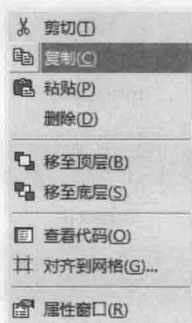
控件名称	名称	Enabled	Interval
Timer1	Timer1	False	100
Timer2	Timer2	False	1200



(7) 在窗体中添加 1 个图像控件，将其名称设置为“Image1”，Visible 属性设置为“False”。



(8) 右击刚添加的图像控件，在弹出的快捷菜单中选择【复制】菜单项。右击窗体空白处，在弹出的快捷菜单中选择【粘贴】菜单项，弹出对话框，提示“创建一个控件数组吗？”，单击【是】按钮，然后删除创建的 Image1(1) 控件。

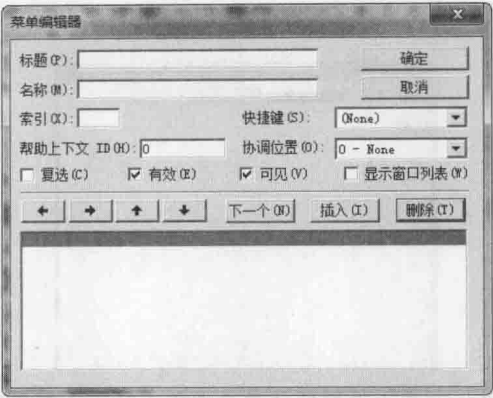


(9) 在窗体中添加 1 个 Label 控件，将其名称设置为“Labelrest”，Caption 属性设置为“暂停”，Visible 属性设置为“False”。



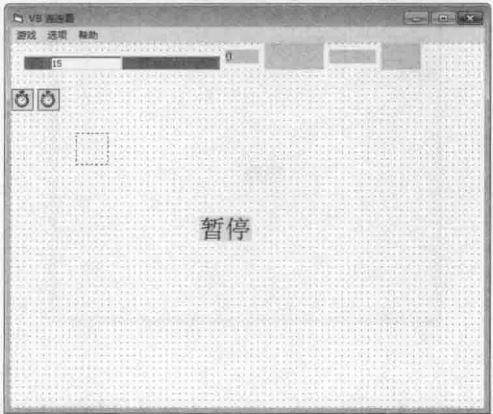
## 第 2 步：编辑菜单

(1) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。



(2) 按下表为窗体添加菜单，标题前带有“...”的为上一级的子菜单，添加时只需输入后面文字即可。

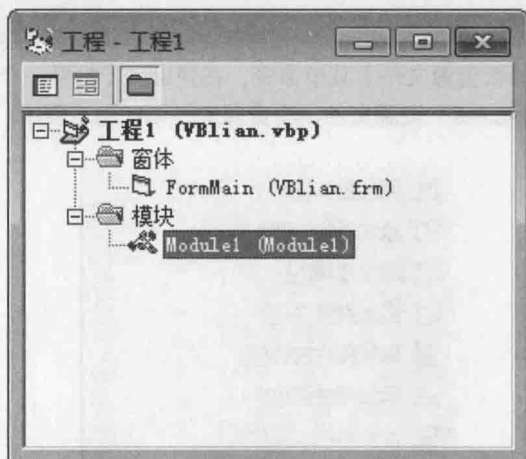
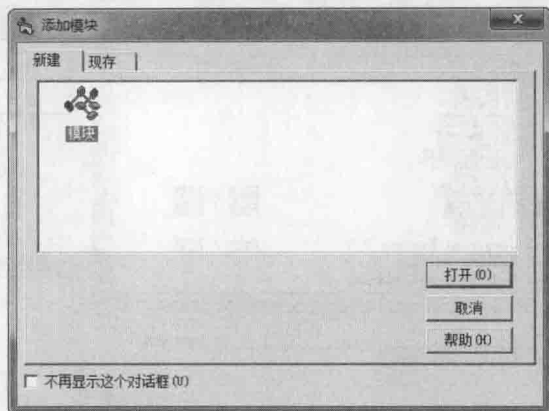
标题	名称	菜单作用
游戏	mngame	“游戏”菜单
... 开始游戏	mnstart	开始游戏
... 退出游戏	mnexit	退出游戏
选项	mnset	“选项”菜单
... 提示	mntip	游戏提示
... 重新洗牌	mnrearrange	重新洗牌
... 暂停游戏	mnrest	暂停游戏
帮助	mnhelp	“帮助”菜单
... 关于	mnabout	显示关于对话框



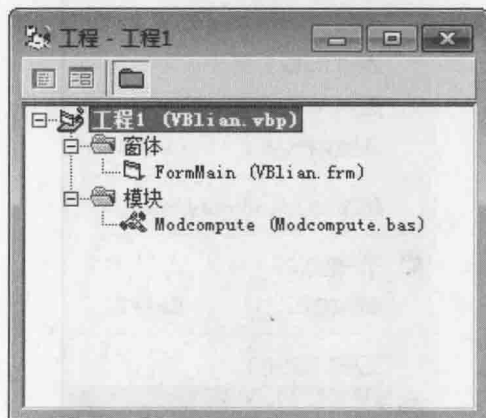


### 第 3 步：添加模块和资源

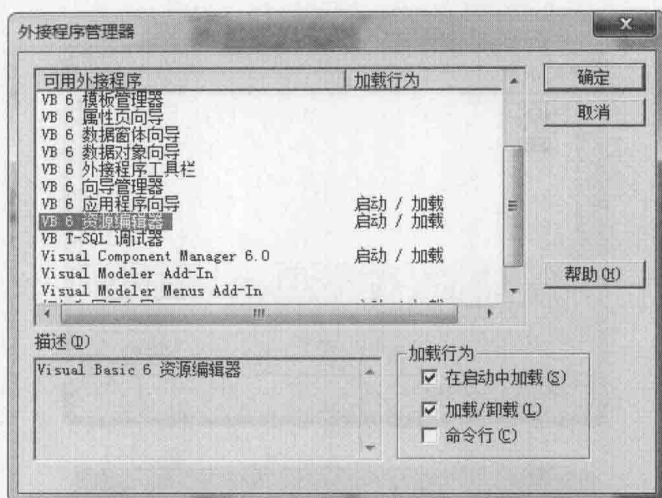
(1) 选择【工程】>【添加模块】菜单命令，在弹出的【添加模块】对话框中选择【新建】选项卡，选中“模块”图标，然后单击【打开】按钮。



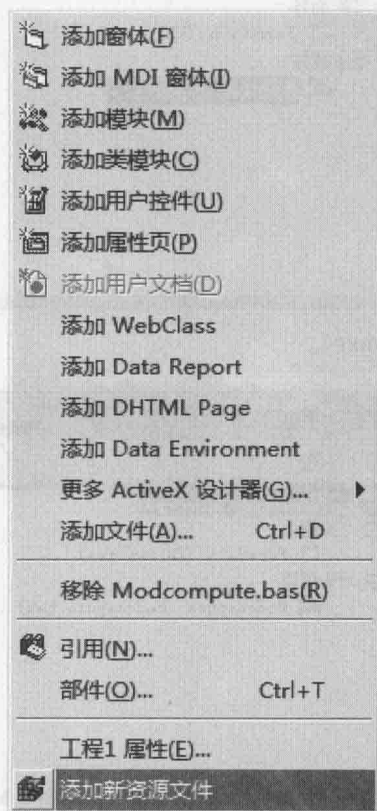
(2) 将其名称设置为“Modcompute”。

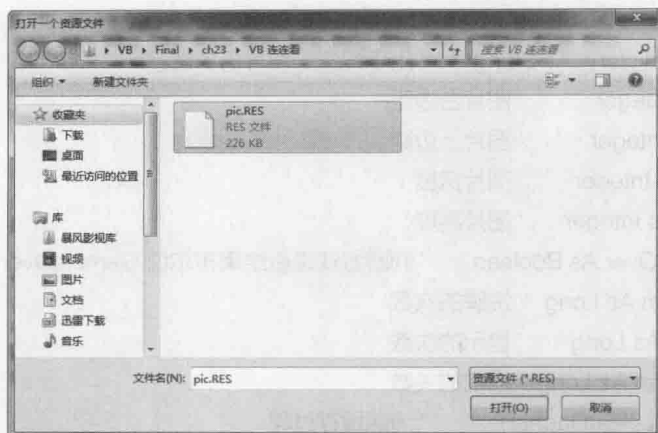


(3) 选择【外接程序】>【外接程序管理器】菜单命令，在弹出的【外接程序管理器】对话框的【可用外接程序】列表框中选中“VB 6 资源编辑器”选项，选中【在启动中加载】和【加载/卸载】两个复选框，然后单击【确定】按钮。



(4) 选择【工程】>【添加新资源文件】菜单命令，在弹出的【打开一个资源文件】对话框中选择随书光盘中的“Sample\ch27\pic.res”资源文件，将资源文件添加到工程中。





#### 提示

“外接程序管理器”可以用于加载和移除集成开发环境 (IDE) 中的外接程序以及指定它们的加载行为的工具。“外接程序管理器”还列出了在 Visual Studio 中注册的所有外接程序。使用“外接程序管理器”可以立即加载或卸载外接程序。

#### 第 4 步：添加模块代码

双击 Modcompute 模块，在代码窗口中添加实现游戏算法的代码（代码 27-1.txt）。

```
01 Global Const ZERO = 0 ' 定义全局常量 ZERO 值为 0
02 Global Const ASCENDING_ORDER = 0 ' 定义全局常量 ASCENDING_ORDER 值为 0
03 Global Const DESCENDING_ORDER = 1 ' 定义全局常量 DESCENDING_ORDER 值为 1
04 Global gliterations' 定义全局变量迭代次数 gliterations
05 Public Xmax As Integer ' 一维元素最大值
06 Public Ymax As Integer ' 二维元素最大值
07 Public Sarr() As String ' 当前图片的编号
08 Dim zc() As String ' 暂存结果
09 Public hsArr() As String ' 图片编号的二维数组表示方式
10 Public rArr() As String ' 产生随机数组
11 Public cArr() As String ' 临时数组存放
12 Public Rcount As Long
13 Public Ccount As Long
14 Public clickNumber As Integer ' 当前的单击数
15 Public clickTemp As Integer
16 Public FirstClick As Integer ' 第 1 次单击的内容
17 Public count As Long
18 Public CurrentLocation As Integer ' 当前检测的位置
19 Dim CurrentPic As Integer ' 即实际图片内容
20 Dim PicCache As String ' 当前图片缓存
```

```

21 Dim HistoryCache As String      ' 图片的历史缓存
22 Dim PicCacheHistory As String   ' 用于搜索的图片历史缓存
23 Dim Lefts As Integer            ' 图片左边距
24 Dim Tops As Integer             ' 图片上边距
25 Dim Widths As Integer           ' 图片宽度
26 Dim Heights As Integer          ' 图片高度
27 Public GamelsOver As Boolean     ' 判断游戏是否结束布尔值 GamelsOver
28 Public resetnum As Long          ' 洗牌的次数
29 Public tipnum As Long            ' 提示的次数
30 Public stagenum As Long          ' 当前的关数
31 Public Function clearCache()     ' 清除缓存过程
32 CurrentLocation = 0              ' 设置当前位置为 0
33 CurrentPic = 0                  ' 设置当前图片编号为 0
34 PicCache = ""                   ' 设置图片缓存为 0
35 HistoryCache = ""               ' 设置历史缓存为 0
36 PicCacheHistory = ""            ' 设置图片缓存历史为 0
37 End Function
38 Public Function arrInit(intxmax As Integer, intymax As Integer)
    ' 初始化图片序列
39 Xmax = intxmax                  ' 一维元素最大值
40 Ymax = intymax                  ' 二维元素最大值
41 ReDim Sarr(Xmax * Ymax)         ' 存放当前图片编号
42 ReDim zc((Xmax - 2) * (Ymax - 2))
43 ReDim hsArr(Xmax, Ymax)         ' 生成图片编号的二维数组
44 Rcount = 0
45 Ccount = 0
46 For i = 1 To FormMain.Image1.count - 1      ' 不显示多余的控件出来
47     Unload FormMain.Image1(i)
48 Next i
49 End Function

```

其他代码见随书光盘。

#### 第5步：添加窗体及菜单命令代码

(1) 双击 FormMain 窗体空白处，在弹出的代码窗口中添加窗体的加载事件 Form\_Load 代码。

```

01 Private Sub Form_Load()
02 Call showtoolTipText      ' 调用 showtoolTipText 过程，在窗体显示游戏信息
03 End Sub

```

(2) 单击【游戏】>【开始游戏】菜单命令，在弹出的代码窗口中添加【开始游戏】菜单的单击事件 mnstart\_Click 代码。

```
01 Private Sub mnstart_Click() '开始
02 Call arrlInit(8, 12) '初始化连连看游戏图片
03 Call loadpicture '加载图片
04 Call MadePicture(CLng(15))
05 Call Gamerestart '调用 Gamerestart 过程
06 Call gameLoadData '调用 gameLoadData 过程
07 End Sub
```

(3) 单击【游戏】>【退出游戏】菜单命令，在弹出的代码窗口中添加【退出游戏】菜单的单击事件 mnexit\_Click 代码。

```
01 Private Sub mnexit_Click()
02 Unload Me '卸载游戏窗体
03 End Sub
```

(4) 单击【选项】>【提示】菜单命令，在弹出的代码窗口中添加【提示】菜单的单击事件 mntip\_Click 代码。

```
01 Private Sub mntip_Click()
02 If tipnum > 0 Then
03 Call NoFindNext
04 tipnum = tipnum - 1 '提示个数变量 tipnum 减 1
05 End If
06 End Sub
```

(5) 单击【选项】>【重新洗牌】菜单命令，在弹出的代码窗口中添加【重新洗牌】菜单的单击事件 mnrearrange\_Click 代码。

```
01 Private Sub mnrearrange_Click()
02 If resetnum > 0 Then
03 Call a '打散
04 resetnum = resetnum - 1 '重新洗牌次数变量 resetnum 减 1
05 End If
06 End Sub
```

(6) 单击【选项】>【暂停游戏】菜单命令，在弹出的代码窗口中添加【暂停游戏】菜单的单击事件 mnrest\_Click 代码。

```
01 Private Sub mnrest_Click()
```

```

02 mnrest.Checked = Not mnrest.Checked
03 Call gamePAUSE
04 End Sub

```

(7) 单击【帮助】>【关于】菜单命令，在弹出的代码窗口中添加【关于】菜单的单击事件 mnabout\_Click 代码。

```

01 Private Sub mnabout_Click()
02     MsgBox "VB 连连看" & Chr(13) & Chr(13) & "VB 程序", vbInformation, "VB 连连看"
03 End Sub

```

### 第6步：添加其他代码

在代码窗口中添加其他相关代码（代码 27-2.txt）。

```

01 Public formFirstClick As Integer, formLastClick As Integer
02 Function gameLoadData() '数据初始化
03     resetnum = 6        '重新洗牌次数变量 resetnum 初始化赋值 6
04     tipnum = 6          '提示个数变量 tipnum 初始化赋值 6
05     stagenum = 1        '当前关数变量 stagenum 初始化赋值 1
06     Text1 = 3000
07     Timer2.Enabled = True
08 End Function
09 Function showtoolTipText() '显示信息
10     Labelreset.Caption = "剩余洗牌次数" & resetnum '洗牌
11     Labeltip.Caption = "提示" & tipnum '提示
12     Labelstageinfo.Caption = Levelstr(stagenum) '关数
13 End Function
14 Function Levelstr(num As Long)
15     Select Case num
16     Case 1
17         Levelstr = "第 0 关 不变化"
18     Case 2
19         Levelstr = "第 1 关 向下"
20     Case 3
21         Levelstr = "第 2 关 向左"
22     Case 4
23         Levelstr = "第 3 关 上下分离"
24     Case 5
25         Levelstr = "第 4 关 左右分离"

```

```

26 Case 6
27     Levelstr = "第 5 关 上下集中"
28 Case 7
29     Levelstr = "第 6 关 左右集中"
30 Case 8
31     Levelstr = "第 7 关 上左下右"
32 Case 9
33     Levelstr = "第 8 关 左下右上"
34 Case 10
35     Levelstr = "第 9 关 向外扩散"
36 Case 11
37     Levelstr = "第 10 关 向内集中"
38 Case 12
39     Levelstr = "特级篇"
40 End Select
41 End Function
    
```

其他代码见随书光盘。

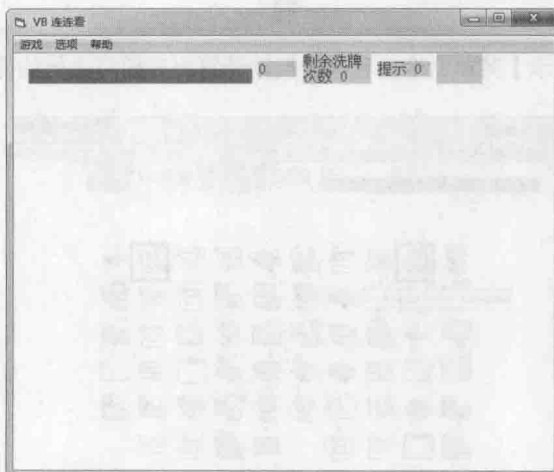
## 27.3 运行系统



本节视频教学录像：2 分钟

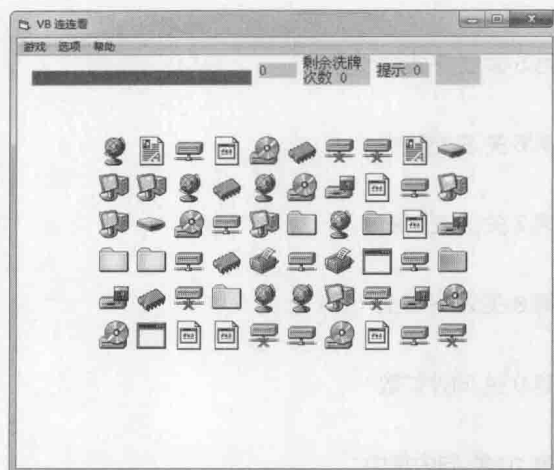
所有工作都做好后，就可以开始体验自己一手打造的连连看游戏了。

(1) 按【F5】快捷键运行程序。

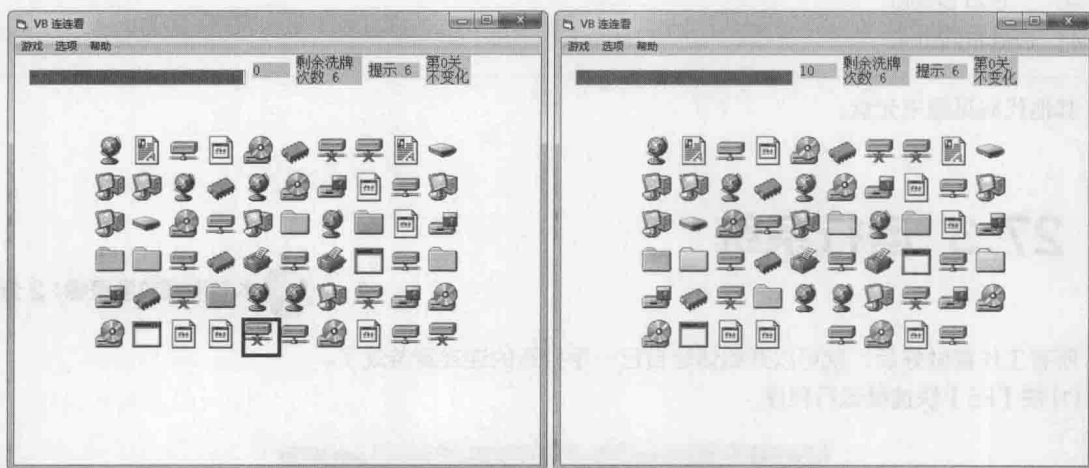


(2) 选择【游戏】>【开始游戏】菜单命令，即可开始游戏。

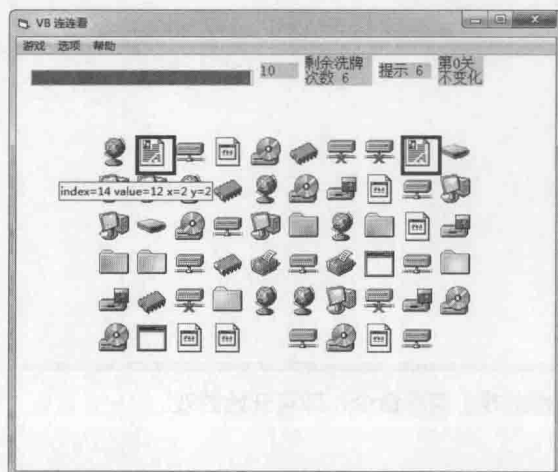




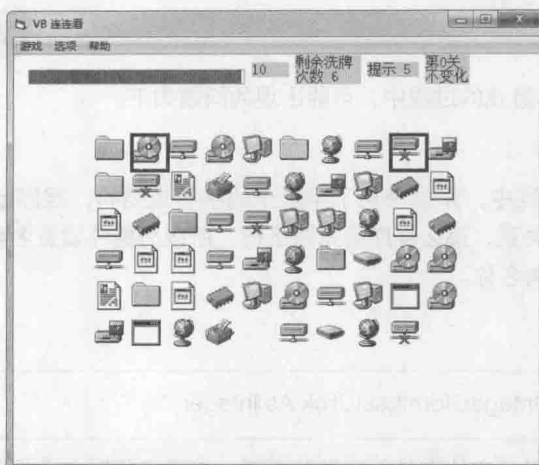
(3) 单击某一个图案，图案的周围会出现 1 个蓝色的框线，然后单击其他相同的图案，如果两个图案间的拐点不超过两个，并且不穿过其他图案，那么单击的这两个图案就会消失。



(4) 选择【选项】>【提示】菜单命令，程序会自动计算出当前可进行的操作，并用蓝色框线画出。



(5) 选择【选项】>【重新洗牌】菜单命令，则会将当前剩下的图案重新排列。



(6) 选择【选项】>【暂停游戏】菜单命令，游戏界面上的图案全部消失，并且显示“暂停”提示。再次选择【选项】>【暂停游戏】菜单命令，则可恢复游戏。



(7) 选择【帮助】>【关于】菜单命令，则会弹出对话框提示相关信息。



## 27.4 高手点拨



本节视频教学录像: 1 分钟

在设计“VB 连连看”小游戏的过程中,可能出现的问题如下。

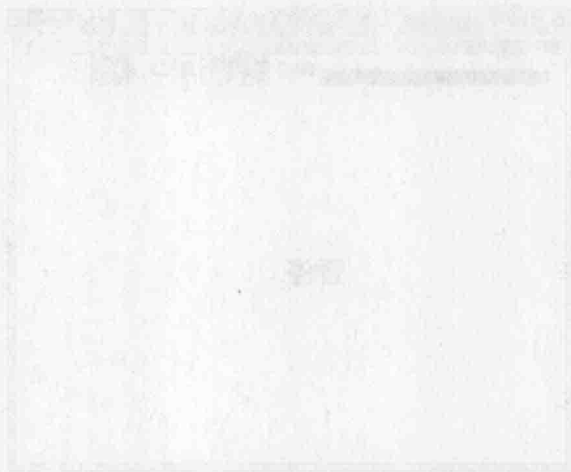
### 1. 控件名称设置不当

在为控件设置名称的过程中,如果使用了不规范的字符或结构,程序就会报告编译错误。在引用时,如果对控件的名称书写失误,那么程序就无法运行。所以为控件设置名称时应遵循一定的规则,应选取好理解、易看懂的词作为名称。

### 2. 函数声明不能随意

```
Public formFirstClick As Integer, formLastClick As Integer
```

在声明这个函数时,如果把它放在其他函数的后面,编译运行时就会报错。所以声明函数时应按照一定的规则进行,不能随意书写。



# 第5篇

## 项目实战

本篇通过个人账目管理系统和超市进销存管理系统的开发实战，带领读者全程感受 Visual Basic 项目工程的开发过程，从而积累一些宝贵的开发经验。

第 28 章 数据库应用开发——个人账目管理系统

第 29 章 打造你的小型超市——超市进销存管理系统

# 第28章



本章视频教学录像：17分钟

## 数据库应用开发——个人账目管理系统

本章详解讲解如何在 Visual Basic 6.0 中开发个人账目管理系统，这个系统能完成基本的日常收入管理、日常支出管理、借入借出款项管理和月度统计等功能。内容包含个人账目管理系统的系统分析、数据库设计、数据库连接语句，以及如何使用数据库控件与数据库连接。

### 本章要点（已掌握的在方框中打钩）

- ☐ 熟悉个人账目管理系统的系统分析
- ☐ 熟悉个人账目管理系统的数据库设计
- ☐ 掌握 MSHFlexGrid 控件与数据库的连接
- ☐ 熟悉数据库查询语句
- ☐ 熟悉数据库插入语句
- ☐ 熟悉修改和删除数据库数据

## 28.1 系统分析



本节视频教学录像: 3 分钟

本章以个人账目管理系统为例介绍数据库应用程序的开发过程, 包括个人账目管理系统的分析、数据库的分析和设计、系统各个模块的创建、系统与数据库的连接等内容。

### 28.1.1 系统需求分析

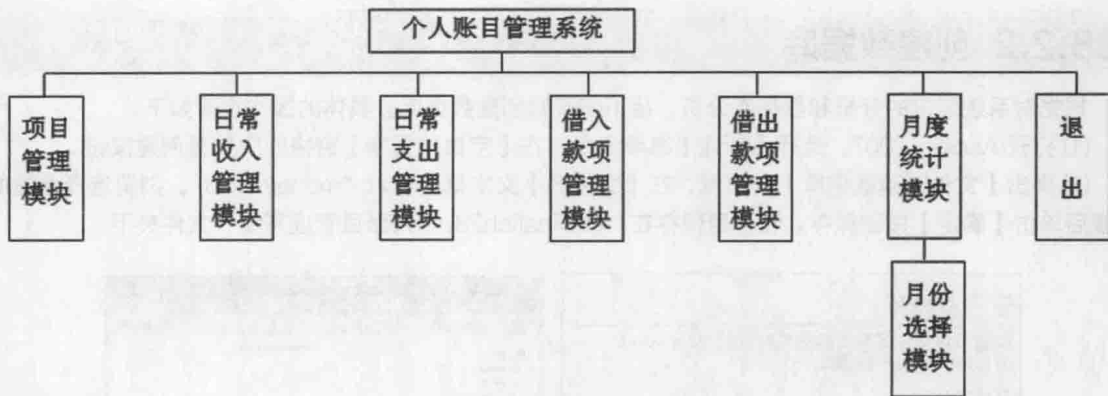
目前, 个人对账目的管理很大程度上仍局限于手工记录的管理方式, 大部分的人采用传统的纸和笔将每一笔收入及支出记录下来。这种方式不仅需要使用大量的时间, 而且由于记录的结构不合理, 还可能导致记录结果与实际不符的情况, 无法达到预期的需求。

为了解决这一问题, 更好地对个人账目进行管理, 应开发一个能实现以下功能的管理系统。

- (1) 可以对收入和支出项目进行管理。
- (2) 可以对日常收入记录进行管理。
- (3) 可以对日常支出记录进行管理。
- (4) 可以对借入款项记录进行管理。
- (5) 可以对借出款项记录进行管理。
- (6) 可以对选定的某一月份的收入和支出情况进行统计。

### 28.1.2 系统功能模块设计

根据对系统需求的分析, 个人账目管理系统主要划分为项目管理模块、日常收入管理模块、日常支出管理模块、借入款项管理模块、借出款项管理模块和月度统计模块等 6 个主要模块。系统中的主要模块如图所示。



系统各个模块的功能如下。

(1) 项目管理模块: 对收入、支出项目进行管理。可以添加新的收入、支出项目, 修改和删除已存在的收入、支出项目。

(2) 日常收入管理模块: 对日常收入记录进行管理。可以添加新的日常收入的日期、方式、金额、来源及备注等信息, 也可以对已保存的日常收入记录进行修改和删除。

- (3) 日常支出管理模块：用于对日常支出记录进行管理。
- (4) 借入款项管理模块：添加借入款项记录，修改和删除已保存的借入款项记录。
- (5) 借出款项管理模块：对借出款项记录进行管理。
- (6) 月度统计模块：对已选择的某一月份的的收入和支出情况进行统计，并显示结果。

## 28.2 数据库分析和设计

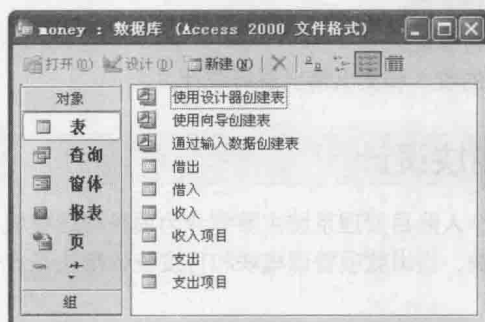


本节视频教学录像：3 分钟

有别于传统的记录方式，数据库存储高效、准确、快捷。本系统采用的数据库为 Access 数据库。

### 28.2.1 数据库分析

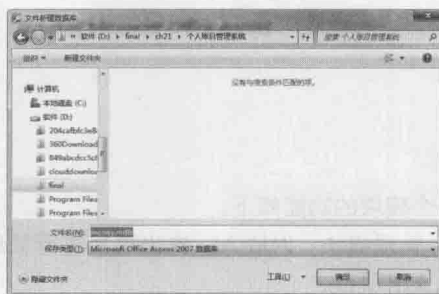
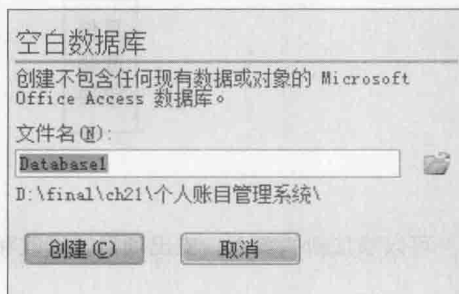
个人账目管理系统数据库主要用到的表包含收入项目表、支出项目表、收入表、支出表、借入表和借出表等，如图所示。



### 28.2.2 创建数据库

根据对系统需求的分析和数据库分析，接下来开始创建数据库。具体的操作步骤如下。

- (1) 打开 Access 2007，选择【新建】菜单命令，在【空白数据库】窗格的中单击创建按钮。
- (2) 弹出【文件新建数据库】对话框，在【文件名】文本框中输入“money.mdb”，浏览合适的位置后单击【确定】按钮保存。在这里保存在“D:\Final\ch28\个人账目管理系统”文件夹下。





## 28.2.3 创建表

### 1. 收入项目表

收入项目表表名为“收入项目”，用来记录个人账目系统的收入项目。该表包含 1 个字段，结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	项目	否	文本	50	是

### 2. 支出项目表

支出项目表表名为“支出项目”，用来记录个人账目系统的支出项目。该表包含 1 个字段，结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	项目	否	文本	50	是

### 3. 收入表

收入表表名为“收入”，用来记录个人账目系统的收入记录。结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	日期	否	文本	50	是
2	方式	否	文本	50	是
3	金额	否	文本	50	是
4	项目	否	文本	50	是
5	来源	否	文本	50	是
6	备注	否	备注	N/A	否
7	序号	是	自动编号	长整型	N/A

### 4. 支出表

支出表表名为“支出”，用来记录个人账目系统的支出记录。结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	日期	否	文本	50	是
2	方式	否	文本	50	是
3	金额	否	文本	50	是
4	项目	否	文本	50	是
5	去向	否	文本	50	是
6	备注	否	备注	N/A	否
7	序号	是	自动编号	长整型	N/A

5. 借入表

借入表表名为“借入”，用来记录个人账目系统的借入款项记录。结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	借款人	否	文本	50	是
2	金额	否	文本	50	是
3	出借人	否	文本	50	是
4	日期	否	日期 / 时间	N/A	是
5	出借原因	否	文本	50	否
6	已还	否	文本	50	是

6. 借出表

借出表表名为“借出”，用来记录个人账目系统的借出款项记录。结构如表所示。

序号	字段名	是否主键	数据类型	字段大小	必填字段
1	借款人	否	文本	50	是
2	金额	否	文本	50	是
3	出借人	否	文本	50	是
4	日期	否	日期 / 时间	N/A	是
5	借款原因	否	文本	50	否
6	已还	否	文本	50	是



提示

在关系数据表中，行称为元组，对应存储文件中的记录，描述了关系中一个具体的个体；列称为属性，对应存储文件中的字段，描述了关系的一个特征；存取一个学生信息的数据单位是记录。

## 28.3 系统界面设计



本节视频教学录像：3 分钟

数据库创建完成，接下来进行个人账目管理系统的工程设计。

### 28.3.1 创建工程和数据库连接模块

首先为个人账目管理系统创建工程。在 Visual Basic 6.0 中创建一个新的标准 EXE 工程，工程保存为“个人账目管理系统”。

由于本系统要经常读写数据库，所以将相关操作封装为函数的形式可以大大简化代码的编写量，并且使代码简洁易懂。

为程序添加一个模块，将名称设置为“Module\_data”，添加其代码如下（代码 28-1.txt）。

```
01 Public Str_path As String '用于存储路径的字符串变量
02 Public Cdate1 As String '传日期（收入情况列表）
03 Public Cdate2 As String '传日期
04 Public aa As Boolean '判断是否选择了日期
05 Public Function Connectstring() As String
06 Dim Str_path As String
07 Str_path = CurDir() & "\money.mdb" '获取存储数据库路径
08 Connectstring = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & Str_path & ";Persist
Security Info=False"
09 End Function
10 Public Function ExeCutesql(ByVal Sql As String, Msgstring As String) As ADODB.Recordset
'共享数据库连接函数
11 Dim Cnn As ADODB.Connection '定义连接对象
12 Dim Rst As ADODB.Recordset '定义记录集
13 Dim Stokens() As String '数组
14 On Error GoTo executesql_error
15 Stokens = Split(Sql) '将 SQL 语句按关键字保存在数组中
16 Set Cnn = New ADODB.Connection '实例化 Cnn 对象
17 Cnn.Open Connectstring
18 If InStr("INSERT,DELETE,UPDATE", UCase$(Stokens(0))) Then
```

```

19      Cnn.Execute Sql
20      Msgstring = Tokens(0) & " 查询成功 "
21      Else
22          Set Rst = New ADODB.Recordset
23          Rst.Open Trim$(Sql), Cnn, adOpenKeyset, adLockOptimistic
        ' 提取符合要求的记录集
24          Set Executesql = Rst
25          Msgstring = " 查询到 " & Rst.RecordCount & " 条记录 "
26      End If
27  executesql_exit:
28      Set Rst = Nothing      ' 释放记录集
29      Set Cnn = Nothing      ' 释放连接语句
30      Exit Function
31  executesql_error:
32      Msgstring = " 查询错误: " & Err.Description
33      Resume executesql_exit
34  End Function

```

## 28.3.2 添加控件

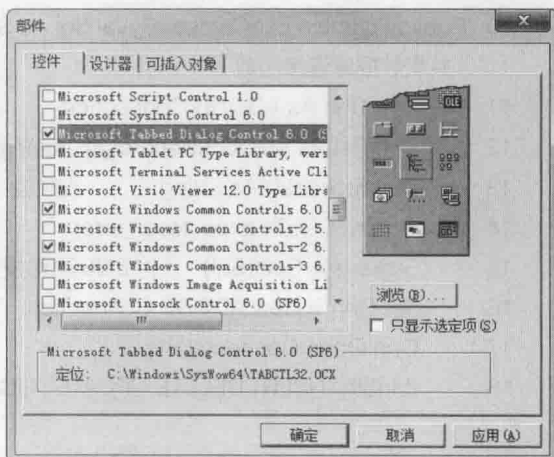
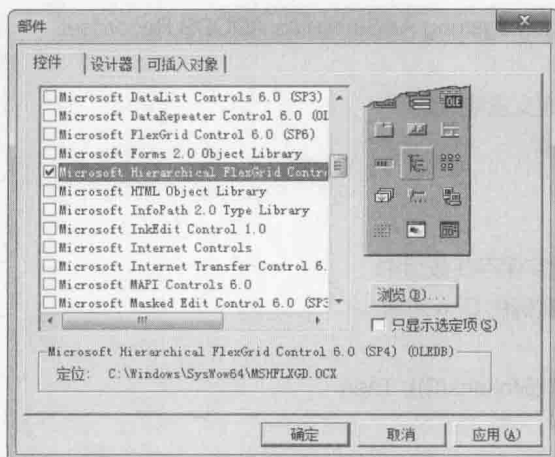
在开始创建各个模块界面之前, 首先要将系统需要使用到的控件添加到工具箱中。需要在工具箱中添加以下 4 个控件。

### 1. Microsoft Hierarchical FlexGrid Control 6.0

该控件在系统中用于显示数据库中的记录。

### 2. Microsoft Tabbed Dialog Control 6.0

该控件用于在【项目管理】和【月度统计】窗体中使用 SSTab 控件。

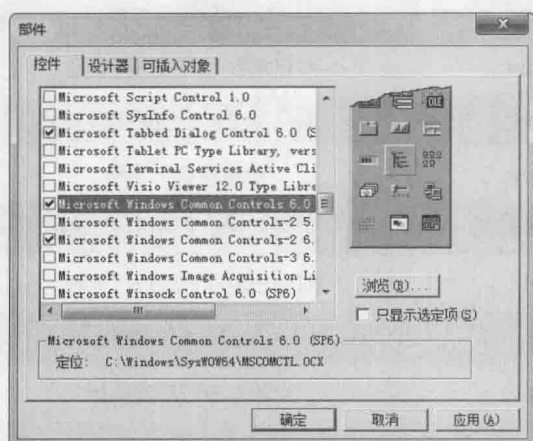


### 3. Microsoft Windows Common Controls 6.0

该控件用于在系统中创建及使用 Toolbar 控件和 ImageList 控件。

### 4. Microsoft Windows Common Controls-2 6.0

该控件用于在系统中创建及使用 DTPicker 控件。

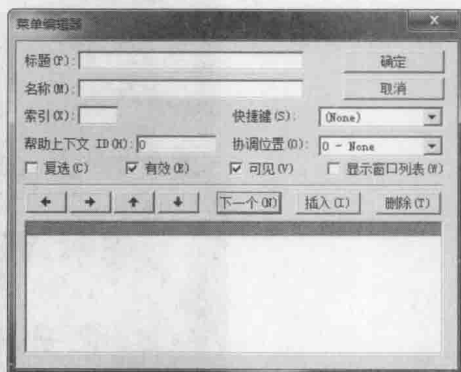


## 28.3.3 系统主界面设计

系统主界面是进行账目管理操作的主要窗口，通过该窗口可以使用菜单或命令按钮调用不同的窗口完成相应的功能。

(1) 选中添加工程时自动创建的窗体 Form1，修改“名称”属性为“frm\_main”，将该窗体的“Caption”属性设置为“个人账目管理系统”，为“Picture”属性选择一个合适的图像文件。

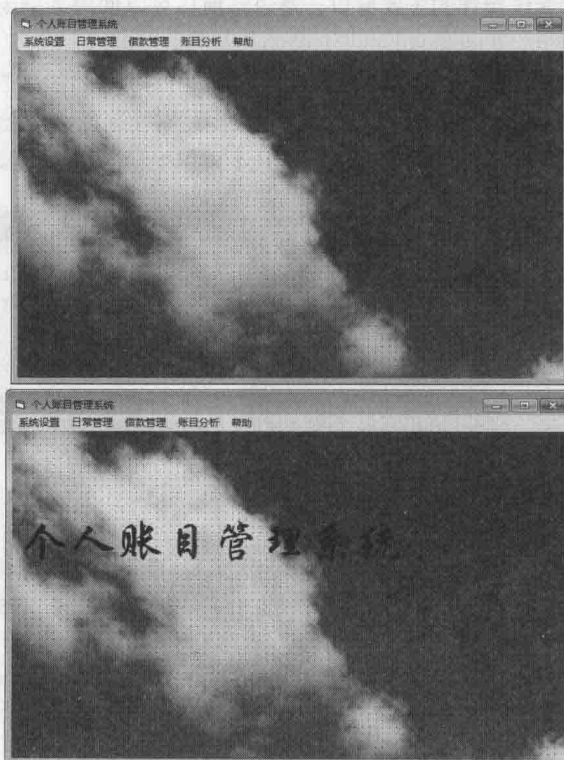
(2) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。



(3) 按下表为窗体添加菜单，标题前带有“...”的为上一级的子菜单，添加时只需输入后面的文字即可。

(4) 在 frm\_main 窗体中添加一个 Label 控件，设置 Caption 属性为“个人账目管理系统”，并设置 Font 属性的字体为“华文行楷”，大小为“初号”。

标题	名称	菜单作用
系统设置	mnu_set	“系统设置”菜单
… 项目管理	mnu_item	对收入、支出项目进行管理
… 退出管理	mnu_exit	退出系统
日常管理	mnu_daily	“日常管理”菜单
… 日常收入	mnu_income	对日常收入记录进行管理
… 日常支出	mnu_spend	对日常支出记录进行管理
借款管理	mnu_money	“借款管理”菜单
… 借入款项	mnu_borrow	对借入款项记录进行管理
… 借出款项	mnu_lend	对借出款项记录进行管理
账目分析	mnu_ana	“账目分析”菜单
… 月度统计	mnu_monthly	对账目进行月度统计
帮助	mnu_help	“帮助”菜单
… 关于	mnu_about	显示关于信息



接下来为主窗体添加 Toolbar 控件和 ImageList 控件，然后为 ImageList 控件添加图片，图片文件见本书随书光盘的“Sample\ch28”。设置 Toolbar 控件的【图像列表】属性为“ImageList1”，然后按下表设置其属性。

索引	标题	工具提示文本	图像
1	项目管理	项目管理	1
2	日常收入	日常收入	2
3	日常支出	日常支出	3
4	借入款项	借入款项	4
5	借出款项	借出款项	5
6	月度统计	月度统计	6
7	关于	关于	7
8	退出	退出	8



主窗体用来根据选择显示不同的窗体，并不用于实现具体的功能，工具栏命令按钮的单击事件则会调用相应的菜单单击事件。

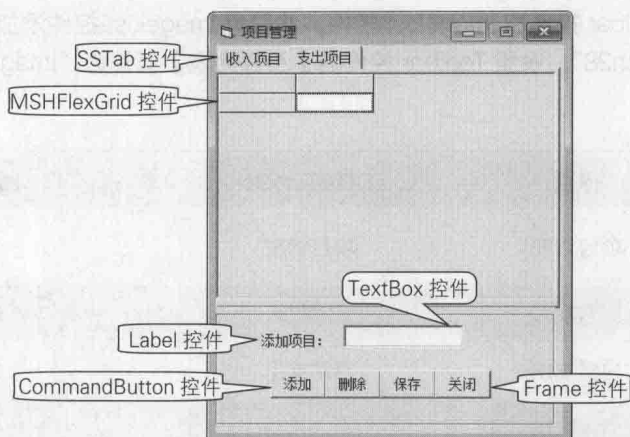
### 28.3.4 系统功能实现的各界面设计

限于篇幅，下面不一一介绍各页面的设计过程，只给出各页面设计的最终效果图。

#### 1.【项目管理】窗体的界面设计

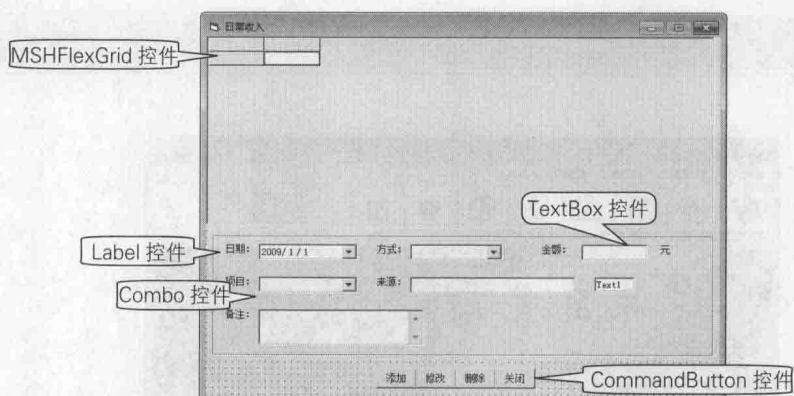
该窗体主要用于添加新的收入项目和支出项目，并对已存在的项目进行修改和删除。





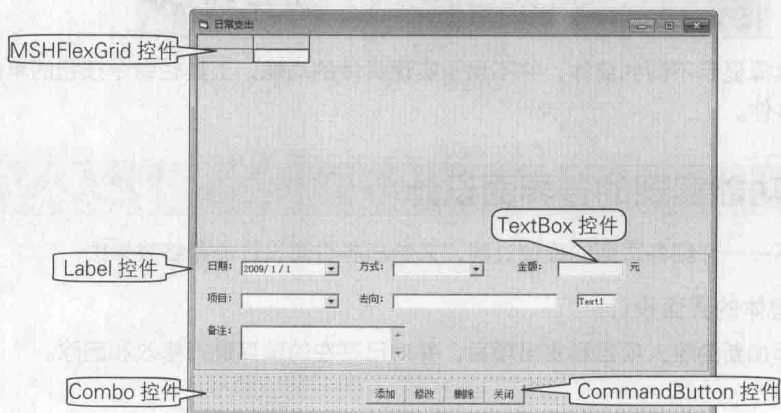
## 2. 【日常收入】窗体的界面设计

该窗体主要用于添加新的日常收入记录，记录收入的日期、方式、金额、项目、来源及备注等信息。



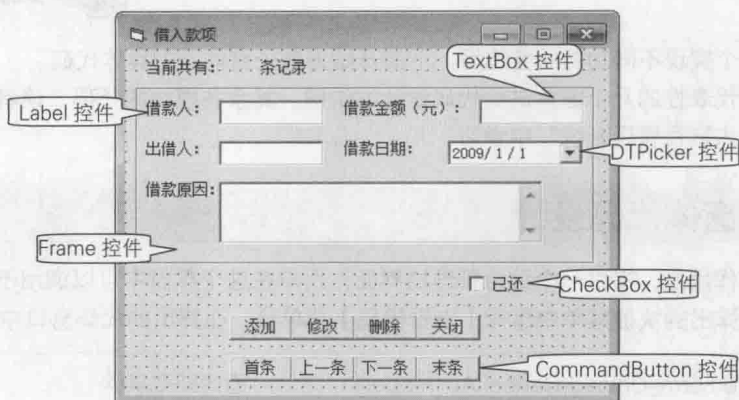
## 3. 【日常支出】窗体的界面设计

该窗体用于对日常支出记录进行管理。



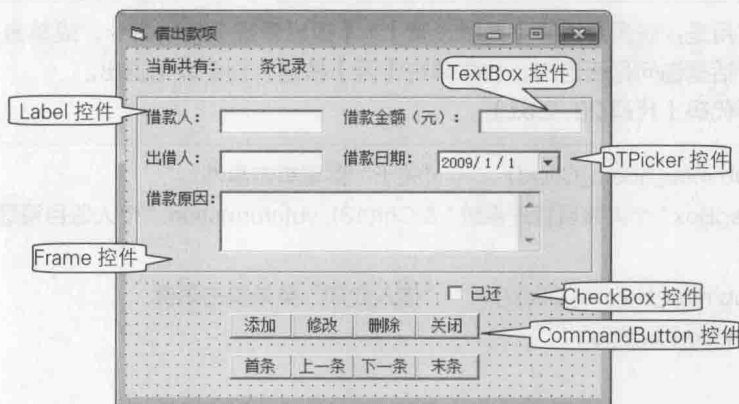
## 4. 【借入款项】窗体的界面设计

该窗体主要用于对借入款项记录进行管理，记录借入款项的借款人、出借人、借款金额、借款日期和是否已还等信息，并能浏览、修改和删除已保存的记录。



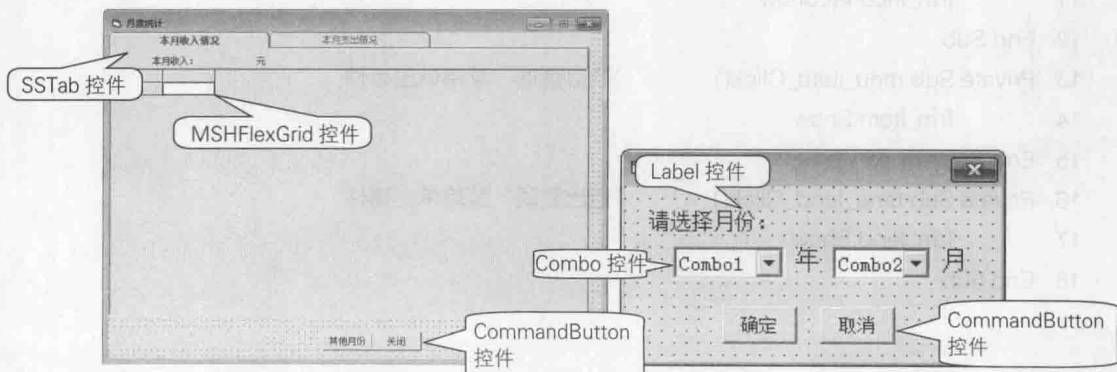
### 5.【借出款项】窗体的界面设计

该窗体用于管理借出款项记录。



### 6.【月度统计】窗体的界面设计

【月度统计】窗体用于对选定某一月份的收入、支出情况进行统计，并根据结果显示本月是盈余或透支。选择月份时将用到【选择月份】窗体。



## 28.4 系统代码设计



本节视频教学录像: 3 分钟

在创建完成各个实现不同功能的窗体后, 下面开始为各个窗体添加具体代码。

在这里只对有代表性的几个窗体进行代码设计的讲解。其余各模块的代码, 读者可以在随书光盘中的“Final\ch28\个人账目管理系统”中学习。

### 28.4.1 主窗体代码设计

主窗体模块的作用是, 提供一个综合管理的界面, 用户在这个界面中可以调用不同的功能窗体。右击窗体空白处, 在弹出的快捷菜单中选择【查看代码】菜单项, 在弹出的代码窗口中添加以下代码。

```
01 Private Sub Form_Unload(Cancel As Integer)      ' 窗体卸载事件
02     If MsgBox(" 是否退出? ", vbYesNo) = vbNo Then Cancel = 1
           ' 如果选择“否”则取消
03 End Sub
```

这段代码的作用是: 当用户选择【系统设置】>【退出管理】菜单命令, 或单击主窗体上的【退出】按钮时, 将弹出对话框询问是否退出, 如果单击【否】按钮, 则会取消退出。

接着添加以下代码 (代码 28-2.txt)。

```
01 Private Sub mnu_about_Click()      ' “关于” 菜单单击事件
02     MsgBox " 个人账目管理系统 " & Chr(13), vbInformation, " 个人账目管理系统 "
03 End Sub
04 Private Sub mnu_borrow_Click()     ' “借入款项” 菜单单击事件
05     frm_borrow.Show
06 End Sub
07 Private Sub mnu_exit_Click()       ' “退出管理” 菜单单击事件
08     Unload Me
09 End Sub
10 Private Sub mnu_income_Click()     ' “日常收入” 菜单单击事件
11     frm_income.Show
12 End Sub
13 Private Sub mnu_item_Click()       ' “项目管理” 菜单单击事件
14     frm_item.Show
15 End Sub
16 Private Sub mnu_lend_Click()       ' “借出款项” 菜单单击事件
17     frm_lend.Show
18 End Sub
```

```

19 Private Sub mnu_monthly_Click() '“月度统计”菜单单击事件
20     frm_monthly.Show
21 End Sub
22 Private Sub mnu_spend_Click() '“日常支出”菜单单击事件
23     frm_spend.Show
24 End Sub

```

这段代码的作用是，选择不同的菜单命令时，则会调用显示相应的窗口。

## 28.4.2 【日常收入】窗体代码设计

右击【日常收入】窗体空白处，在弹出的快捷菜单中选择【查看代码】菜单项，在弹出的代码窗口中添加以下代码（代码 28-3.txt）。

```

01 Dim Mydb As New ADODB.Recordset '定义 Mydb、Mydb1 等为 ADODB.Recordset
    对象，用于容纳数据表记录
02 Dim Mydb1 As New ADODB.Recordset
03 Dim Mydb2 As New ADODB.Recordset
04 Dim Count1 As New ADODB.Recordset
05 Dim Str_text As String '定义 Str_text 为字符串变量，用于修改数据表记录
06 Private Sub Form_Load() '窗体加载事件
07     Dim A As Integer 'A 用于保存“收入项目”表中记录数
08     Set Mydb1 = ExeCutesql("select * from 收入项目 ", Str_text)
09     A = Mydb1.RecordCount
10     Set Combo2.DataSource = Mydb1 '设置 Combo2 的数据源为 Mydb1
11     For I = 1 To A '为 Combo2 中添加“收入项目”表中的字段
12         Combo2.AddItem Mydb1.Fields(0)
13         Mydb1.MoveNext
14     If Mydb1.EOF Then Exit For
15 Next I
16 Call Db
17 DTPicker1.Value = Date
18 Combo1.AddItem "现金" '为 Combo1 添加“现金”字段
19 Combo1.AddItem "实物" '为 Combo1 添加“实物”字段
20 Combo1.AddItem "支票" '为 Combo1 添加“支票”字段
21 Combo1.AddItem "转账" '为 Combo1 添加“转账”字段
22 End Sub
23 Public Function Db()
24     Set Mydb = ExeCutesql("select * from 收入 order by 序号 ", Str_text)

```

```

25     Set MSHFlexGrid1.DataSource = Mydb
26 End Function

```

当用户在录入有效的日期、方式、金额、项目、来源及备注信息后,单击【添加】按钮,即可把收入记录添加到数据库中。相关代码如下(代码28-4.txt)。

```

01 Private Sub cmd_add_Click()      '“添加”按钮单击事件
02     On Error Resume Next
03     Dim A, B      'A 用于判断是否添加记录, B 用于保存总记录数值
04     B = 1 'B 初始设置为 1
05     Set Count1 = ExeCutesql("select * from 收入 ", Str_text)      '查“收入”表,结果传给 Count1
06     Count1.MoveLast      'Count1 移至记录结尾
07     B = Count1.Fields(6) + 1      '总记录数加 1, 结果传给 B
08     A = MsgBox("是否添加当前记录?", vbYesNo + 32, "添加记录")
09     If A = vbYes Then
10         If txt_intake.Text = "" Then
11             MsgBox "请填写来源!", vbOKOnly + 32, "注意"
12             txt_intake.SetFocus
13         Else
14             ExeCutesql "INSERT INTO 收入 VALUES('" & Format(DTPicker1.Value, "yyyy-mm-dd") & "','" &
Combo1.Text & "','" & txt_money.Text & "','" & Combo2.Text & "','" & txt_intake.Text & "','" & txt_mome.Text & "','"
& B & ")", Str_text '写入数据库
15             MsgBox "数据已经保存!", vbOKOnly + 64, "添加记录"
16             Call Db
17         End If
18     End If
19 End Sub

```

单击【删除】按钮可以把某个信息记录删除,其相关代码如下(代码28-5.txt)。

```

01 Private Sub cmd_del_Click()      '“删除”按钮单击事件
02     On Error Resume Next
03     Dim A
04     A = MsgBox("是否删除当前记录?", vbYesNo + 32 + 256, "删除记录")
05     If A = vbYes Then
06         ExeCutesql "DELETE from 收入 where 序号 =" & txt_note.Text & "", Str_text
'删除“收入”表中序号值为 txt_note 文本框内容的记录
07         Call Db
08         Set Mydb = ExeCutesql("select * from 收入 ", Str_text)

```

```

09 Set MSHFlexGrid1.DataSource = Mydb '设置 MSHFlexGrid1 的数据源为 Mydb
10 End If
11 End Sub

```

### 28.4.3 【日常支出】窗体代码设计

【日常支出】窗体的设计思路与【日常收入】窗体相同，所以代码的结构相似，不同的地方在于对数据库的操作语句。

在【日常支出】窗体中，【金额】文本框在获得焦点又失去焦点时，会判断文本框中输入的内容是否为空，以及是否为数字。如果为空，则提醒用户“请输入金额”；如果输入的内容不是数字，则提醒用户“金额只能输入数字”。实现这一功能的代码如下（代码 28-6.txt）。

```

01 Private Sub txt_money_LostFocus()
02     Dim A As Boolean '判断是否为数字
03     Dim C '暂存 txt_money 文本框内容
04     C = txt_money.Text
05     A = IsNumeric(C)
06     If C = "" Then '如果为空
07         MsgBox "请输入金额!", vbOKOnly + 32, "注意!"
08         txt_money.SetFocus 'txt_money 获得焦点
09     Else
10         If A = False Then '不是数字
11             MsgBox "金额只能输入数字!", vbOKOnly + 32, "注意!"
12             txt_money.SetFocus
13         End If
14     End If
15 End Sub

```

### 28.4.4 【借入款项】窗体代码设计

在窗体加载时，执行数据库查询语句，查询“借入”表，并将结果传回，然后设置 Check1 的默认值为“0”，Label2 的 Caption 值为数据记录个数。相关代码如下（代码 28-7.txt）。

```

01 Dim Mydb As New ADODB.Recordset
    '定义 Mydb 为 ADODB.Recordset 对象，用于容纳数据表记录
02 Dim Mydb1 As New ADODB.Recordset
    '定义 Mydb1 为 ADODB.Recordset 对象，用于容纳数据表记录
03 Dim Str_text As String '定义 Str_text 为字符串变量，用于修改数据表记录

```

```

04 Private Sub Form_Load() ' 窗体加载事件
05     On Error Resume Next
06     Call Db
07     Check1.Value = 0      ' 复选按钮 Check1 值设置为 0
08     Label2.Caption = Mydb.RecordCount
09     DTPicker1.Value = Date
10 End Sub
11 Private Function Db()
12     Set Mydb = ExeCutesql("select * from 借入", Str_text)
    ' 查询“借入”表，结果传给 Mydb
13 End Function

```

单击【添加】按钮，可将与借入款项有关的信息写入数据库。相关代码如下（代码 28-8.txt）。

```

01 Private Sub cmd_add_Click()      ' “添加”按钮单击事件
02     On Error Resume Next
03     Select Case MsgBox("是否添加当前记录?", vbYesNo + 32, "添加记录")
        ' 询问是否添加记录
04     Case vbYes      ' 选择“是”按钮
05         ExeCutesql "insert into 借入 values('" & txt_man_borrow.Text & "','" & txt_money.Text & "','" & txt_
man_lend.Text & "','" & Format(DTPicker1.Value, "yyyy-mm-dd") & "','" & txt_way.Text & "','" & Check1.Value
& "')" , Str_text      ' 写入数据库
06         MsgBox "数据已经保存!", vbOKOnly + 64, "添加记录"
        ' 报告添加记录成功
07         Call Db      ' 调用 Db 过程
08         Mydb.Requery ' 重新查询 Mydb 数据表
09         Label2.Caption = Mydb.RecordCount      ' 设定 Label2 的 Caption 值为 Mydb 的记录数
10     Case vbNo      ' 选择“否”按钮
11         frm_borrow.Show
12     End Select
13 End Sub

```

### 28.4.5 【借出款项】窗体代码设计

Banging 过程用于设置【借出款项】窗体中相关控件的数据源属性，以及其数据段。其代码如下（代码 28-9.txt）。

```

01 Private Function Banging()
02     On Error Resume Next

```



```

03 Set txt_man_borrow.DataSource = Mydb ' 设置相关控件的数据源为 Mydb
04 Set txt_man_lend.DataSource = Mydb
05 Set txt_money.DataSource = Mydb
06 Set DTPicker1.DataSource = Mydb
07 Set txt_way.DataSource = Mydb
08 Set Check1.DataSource = Mydb
09 txt_man_borrow.DataField = " 借款人 " ' 设置相关控件的数据段
10 txt_man_lend.DataField = " 出借人 "
11 txt_money.DataField = " 金额 "
12 DTPicker1.DataField = " 日期 "
13 txt_way.DataField = " 借款原因 "
14 Check1.DataField = " 已还 "
15 End Function

```

### 28.4.6 【月度统计】窗体代码设计

【月度统计】窗体加载时执行 Activate 事件，会对本月的收入、支出情况进行统计，并在窗体上显示出来。窗体的 Activate 事件代码主要如下（代码 28-10.txt）。

```

01 Private Sub Form_Activate() ' 窗体激活事件
02 On Error Resume Next
03 Dim A, B, C As Integer
    'A 为存储收入金额变量，B 为存储支出金额变量，C 为存储净收入金额，D 用于存储“盈余”或“透支”
    字符串
04 Dim D As String
05 Dim Year1, Month1, Riqi, Riqi1, Riqi3, Riqi4
06
07 If aa = True Then ' 选择了日期
08 Set Mydb = ExeCutesql("select * from 收入 where 日期 between " & Cdate1 & " and " &
Cdate2 & " ", "") ' 查询“收入”表中日期介于 Cdate1 与 Cdate2 之间的记录
09 Set MSHFlexGrid1.DataSource = Mydb ' 设置 MSHFlexGrid1 数据源为 Mydb
10 Set Mydb1 = ExeCutesql("select * from 支出 where 日期 between " & Cdate1 & " and " &
Cdate2 & " ", "") ' 查询“支出”表中日期介于 Cdate1 与 Cdate2 之间的记录
11 Set MSHFlexGrid2.DataSource = Mydb1 ' 设置 MSHFlexGrid2 数据源为 Mydb1
12
13 Set Money = ExeCutesql("select sum( 金额 ) from 收入 where 日期 between " & Cdate1 &
"and " & Cdate2 & " ", "")
    ' 对“收入”表中日期介于 Cdate1 与 Cdate2 之间的金额进行求和

```

```

14  A = Money.Fields(0) ' 求和结果传给 A
15  If IsNull(A) Then    ' 值为空
16      A = 0           ' A 值设置为 0
17  End If
18  Label2.Caption = A
19  Set Money1 = ExeCutesql("select sum( 金额 ) from 支出 where 日期 between '" & Cdate1 &
"and '" & Cdate2 & "'", "") ' 对“支出”表中日期介于 Cdate1 与 Cdate2 之间的金额求和
20  B = Money1.Fields(0) ' 求和结果传给 B
21  If IsNull(B) Then    ' 值为空
22      B = 0           ' B 值设置为 0
23  End If
24  Label6.Caption = B
25  C = A - B           ' 净收入值传给 C
26  If C > 0 Then        ' C 大于 0
27      D = " 盈余 "     ' 盈余
28  Else                 ' 不大于 0
29      D = " 透支 "     ' 透支
30  End If
31  Label4.Caption = Format(Cdate1, "yyyy 年 mm 月") & " " & "本月" & D & C & "元"
' 显示盈余或透支结果

```

单击【其他月份】按钮，将会加载【选择月份】窗体用于选择月份。【选择月份】窗体的【确定】按钮单击事件代码如下（代码 28-11.txt）。

```

01 Private Sub Command1_Click()
02     If Combo1.Text = "" Then ' Combo1 没有选择值
03         MsgBox "请选择年份！ ", vbOKOnly + 32, " 注意！ "
04     Else
05         If Combo2.Text = "" Then ' Combo2 没有选择值
06             MsgBox "请选择月份！ ", vbOKOnly + 32, " 注意！ "
07         Else
08             aa = True ' 选择了日期
09             Year1 = Combo1.Text ' Combo1 的值传给 Year1
10             Month = Combo2.Text ' Combo2 的值传给 Month
11             Riqi = Year1 & "-" & Month "Year1 - Month" 值传给字符串变量 Riqi
12             Riqi1 = Year1 & "-" & Month + 1 "Year1 - Month+1" 值传给字符串变量 Riqi1
13             Cdate1 = Format(Riqi, "yyyy-mm")
' Riqi 的值以 yyyy-mm 形式传给 Cdate1，作为月度总结开始月份
14             Cdate2 = Format(Riqi1, "yyyy-mm")

```

'Riqi1 的值以 yyyy-mm 形式传给 Cdate2, 作为月度总结结束月份

```
15 Unload Me
16 End If
17 End If
18 End Sub
```

## 28.5 运行系统



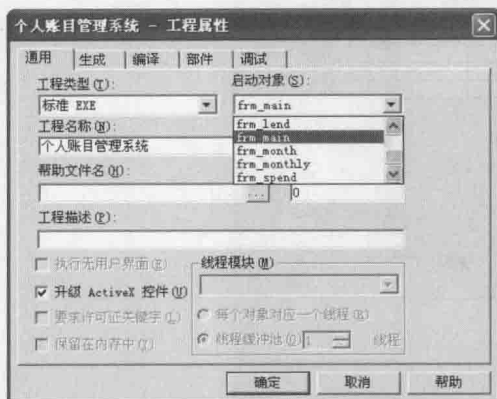
本节视频教学录像: 4 分钟

设计完成个人账目管理系统的界面以及代码后, 就可以开始使用个人账目管理系统对账目进行管理了。

### 28.5.1 系统主界面操作

系统的运行应该从主窗体开始, 这样才能调用其他窗体。选择【工程】>【属性】菜单命令, 在弹出的【个人账目管理系统 - 工程属性】对话框中, 选取【通用】选项卡下的【启动对象】下拉列表中的【frm\_main】窗体, 然后单击【确定】按钮。

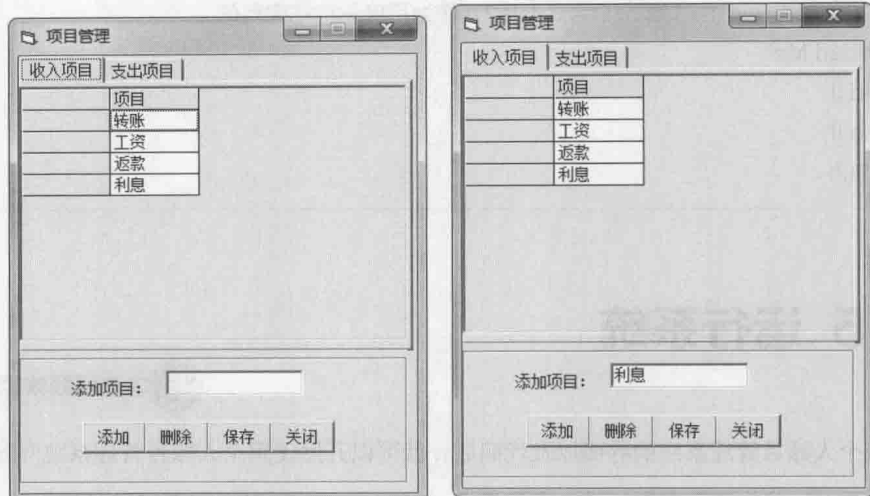
单击工程中的【启动】按钮或按【F5】快捷键运行系统, 显示系统主界面, 在主界面上选择不同的菜单命令或者单击不同的命令按钮, 即可显示不同的窗体。



### 28.5.2 项目管理操作

选择【系统设置】>【项目管理】菜单命令, 或者单击工具栏中的【项目管理】按钮, 即可进入【项目管理】界面。

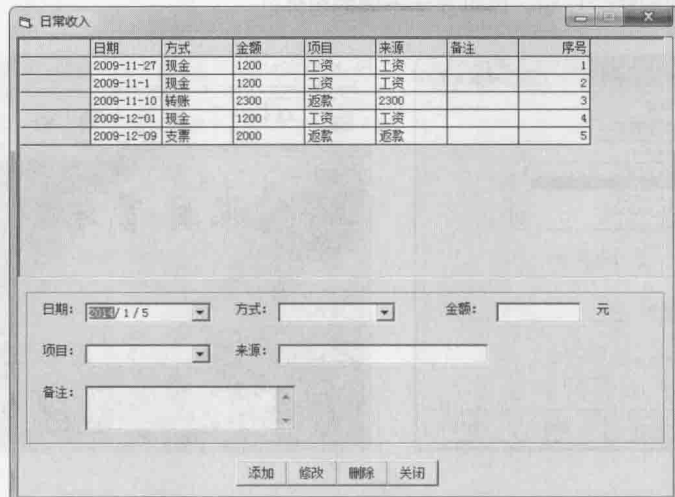
单击【添加】按钮, 在【添加项目】文本框中输入“转账”, 然后单击【保存】按钮, 即可往系统中添加一条收入项目。



选中一条记录后单击【删除】按钮，即可把已经存在的收入项目删除。  
对支出项目的管理与此类似。

### 28.5.3 日常收入、支出管理操作

单击工具栏中的【日常收入】按钮，即可进入【日常收入】管理的界面。



分别填写日常收入的相关信息后单击【添加】按钮，即可添加一条收入记录。选中一条记录，然后单击【删除】按钮，即可把记录删除。

对日常支出的管理与此类似。

### 28.5.4 借入款项、借出款项管理操作

单击工具栏中的【借入款项】按钮，即可进入【借入款项】管理的界面。

对借入款项的添加、修改和删除操作与对日常收入记录的管理类似。单击【首条】按钮，即可显示

借入款项的第 1 条记录信息。

借入款项

当前共有: 4 条记录

借款人: 借款人1      借款金额(元): 12345

出借人: 出借人1      借款日期: 2009/11/26

借款原因: 借款原因1

☐ 已还

添加   修改   删除   关闭

首条   上一条   下一条   末条

单击【上一条】、【下一条】按钮，可以浏览记录，单击【末条】按钮则会显示最后一条记录信息。对借出款项的管理与此类似。

## 28.5.5 月度统计管理操作

单击工具栏中的【月度统计】按钮，即可进入【月度统计】管理的界面。

窗体加载时，会自动对当前月份进行统计，分别在【本月收入情况】和【本月支出情况】显示详细记录，并在窗体下方显示月度盈余或透支总结。

单击【其他月份】按钮，在弹出的【选择月份】窗口中选择相应的日期，然后单击【确定】按钮，则会在【月度统计】窗口中显示已选择月份的统计详情。

月度统计

本月收入情况

日期	方式	金额	类别	备注	序号
2009-11-01	工资	1000	工资		1
2009-11-05	工资	1000	工资		2

2009年11月, 本月透支: 2000元

其他月份   关闭

选择月份

请选择月份:

2009 年 11 月

确定   取消

月度统计

本月支出情况

日期	方式	金额	类别	备注	序号
2009-11-01	工资	1000	工资		1
2009-11-05	工资	1000	工资		2

2009年11月, 本月透支: 2000元

其他月份   关闭

## 28.6 高手点拨



本节视频教学录像: 1 分钟

在设计“个人账目管理系统”的过程中，可能会出现以下一些问题。

### 1. 数据库语句格式不对

系统对数据库的操作，都是通过数据库操作语句完成的。在【日常收入】模块中，【添加】按钮的核心代码如下。

```
ExeCutesql "INSERT INTO 收入 VALUES(' & Format(DTPicker1.Value, "yyyy-mm-dd") & ', ' & Combo1.Text
```

```
& "," & txt_money.Text & "," & Combo2.Text & "," & txt_intake.Text & "," & txt_mome.Text & "," & B & ") ", Str_text
```

这句代码的作用是, 将【日常收入】窗体相关控件的输入信息, 写入到数据库的【收入】表中, 数据的先后顺序必须与【收入】表中的字段相对应, 否则操作就会不成功。

## 2. 对相关控件数据源属性的设置不对

在【日常收入】窗体加载的过程中, 加载事件会向【方式】下拉列表添加【收入项目】中的相关信息, 其核心代码如下。

```
01 Dim A As Integer 'A 用于保存“收入项目”表中记录数
02 Set Mydb1 = ExeCutesql("select * from 收入项目 ", Str_text)
03 A = Mydb1.RecordCount
04 Set Combo2.DataSource = Mydb1 '设置 Combo2 的数据源为 Mydb1
05 For I = 1 To A '为 Combo2 中添加“收入项目”表中的字段
06     Combo2.AddItem Mydb1.Fields(0)
07     Mydb1.MoveNext
08     If Mydb1.EOF Then Exit For
09 Next I
```

第 06 行代码的作用是, 将 Mydb1 的第 0 列的数据依次添加到 Combo2 中。本系统中还有其他类似的代码, 在添加时一定要正确填写 Fields 后面的参数, 否则就会出现错误。

# 第29章



本章视频教学录像：8 分钟

## 打造你的小型超市——超市进销存管理系统

超市的信息化应首先从进销存管理开始，使用信息化的超市进销存管理系统可以高效地组织超市资源，精确掌握超市的发展趋向，提高管理水平。


本章讲解如何利用 Visual Basic 6.0 开发超市进销存管理系统，包括多窗体应用程序的创建、SSTab 控件和 Data 控件的应用、使用 Toolbar 控件创建工具栏和利用 DBCombo 控件输入数据等。

### 本章要点（已掌握的在方框中打钩）

- ☐ 掌握多窗体应用程序的创建
- ☐ 掌握 SSTab 控件的应用
- ☐ 掌握 Data 控件的应用
- ☐ 掌握 Toolbar 控件的应用
- ☐ 熟悉 DBCombo 控件的应用
- ☐ 熟悉 Timer 控件的应用




## 29.1 需求及功能分析

 本节视频教学录像: 1 分钟

一个超市在发展的过程中, 随着商品数量的逐渐增加, 与商品有关的信息量也会成倍增长, 每时每刻都要对商品的各種信息进行统计分析。这些工作如果全部通过人工来完成, 不仅效率低下, 管理成本高, 而且面对大量的信息, 也很容易出错。所以开发一个小型的超市管理系统很有必要。

一个小型超市管理系统应该具有员工管理、供应商管理、客户管理、货物分类管理、货物库存管理, 以及进货和出货记录管理等基本功能。

## 29.2 数据库设计

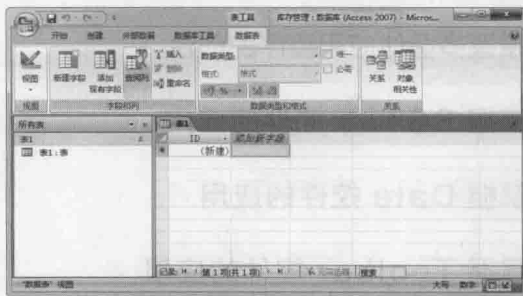
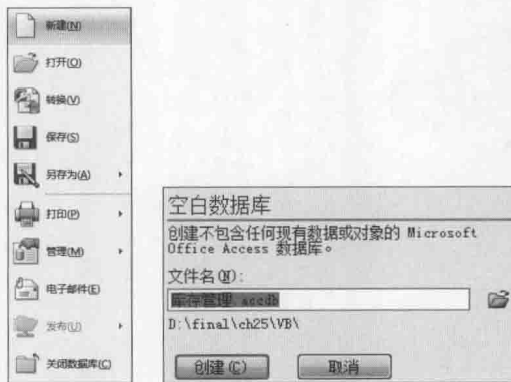
 本节视频教学录像: 2 分钟

Microsoft Access 数据库管理系统界面友好、易学易用、开发简单、接口灵活, 能完善地处理各种数据库对象, 具有强大的数据处理功能, 而且可以方便地生成各种数据对象, 非常适于小型超市管理系统数据库的设计。所以, 在本系统中选用 Microsoft Access 来作为后台数据库。

### 29.2.1 创建数据库

在 Access 2007 中创建数据库的具体步骤如下。

- (1) 打开 Access 2007, 选择【Office 按钮】>【新建】菜单命令。
- (2) 在【空白数据库】窗格中, 在【文件名】文本框中输入“库存管理.mdb”, 浏览合适的位置后单击【创建】按钮保存。在这里我们保存在“D:\Final\ch29\VB 小型超市管理”文件夹下。
- (3) 这样就创建了一个新的空数据库。



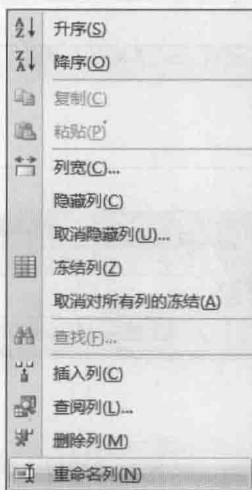
### 29.2.2 创建表

下面开始为数据库添加表, 具体的操作步骤如下。

- (1) 单击工具栏中的【新建】按钮, 在该页面中单击【表】按钮。

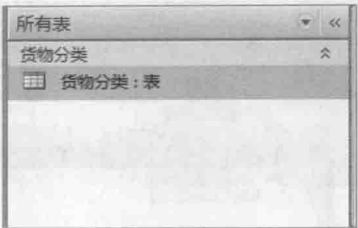


(2) 在弹出的表中，右击“字段 1”列表，在弹出的快捷菜单中选择【重命名列】菜单项，在编辑区输入“分类”。



(3) 单击【保存】菜单命令，在弹出的【另存为】对话框的【表名称】文本框中输入“货物分类”，然后单击【确定】按钮。这时 Access 2007 的主界面中就多了 1 个“货物分类”表。





(4) 按照上面的方法再创建 1 个表，并依次为其添加“名称”、“分类”、“单位”、“数量”和“备注”等列，然后保存表为“货物库存表”。

	名称	分类	单位	数量	备注
▶					

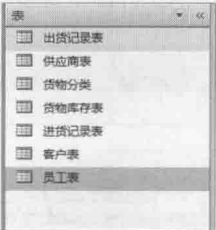
名称	分类	单位	数量	备注
* (新建)				



(5) 按照下面的表格创建其他表。

表	添加列项目	保存名称
1	名称、地址、电话、联系人、级别、备注	供应商表
2	名称、地址、电话、联系人、产品、单位、数量	客户表
3	姓名、职务、电话	员工表
4	名称、数量、经手人、用途、日期、时间	出货记录表
5	名称、供应商、数量、单价、经手人、日期、时间	进货记录表

创建完成的表如图所示。



## 29.3 系统界面设计




本节视频教学录像：2 分钟

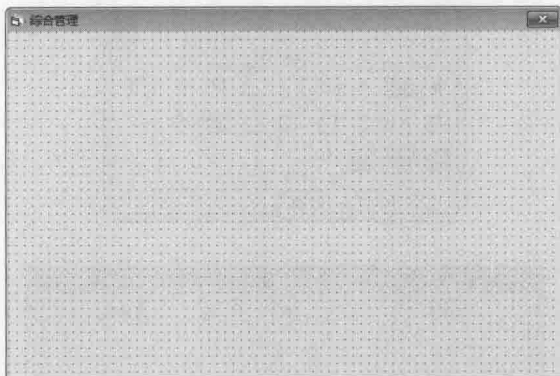
完成数据库的设计后，本节讲述系统界面的设计。对于部分界面将详细讲解，其他界面限于篇幅将给出最终效果图。

### 29.3.1 【综合管理】窗体设计

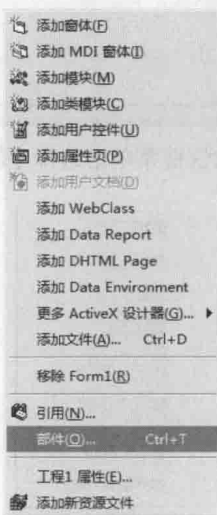
【综合管理】窗体设计的具体步骤如下。

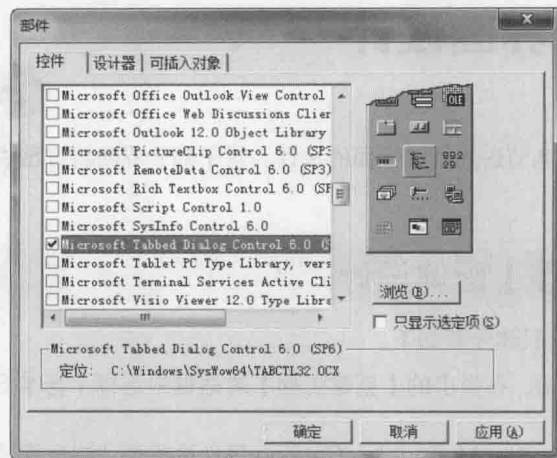
(1) 启动 Visual Basic 6.0，在弹出的【新建工程】对话框中选择【标准 EXE】图标 ，然后单击【打开】按钮。

(2) 将 Form1 的名称改为“frmManage”，Caption 属性设置为“综合管理”。将 BorderStyle 属性设置为“1 - Fixed Single”，并调整其大小。

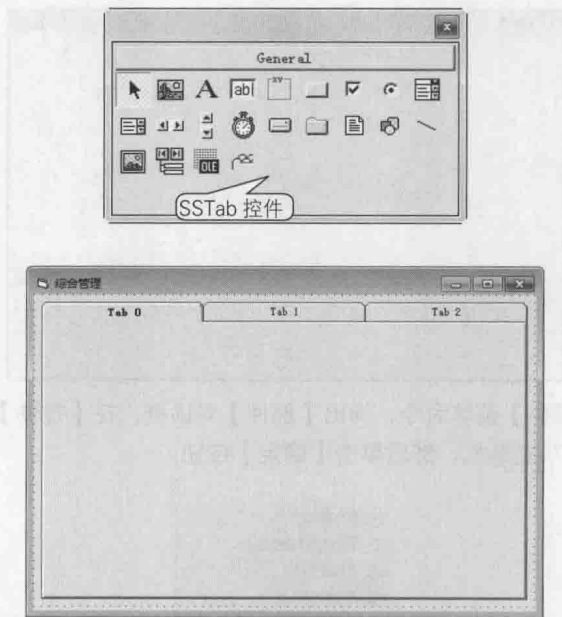


(3) 选择【工程】>【部件】菜单命令，弹出【部件】对话框，在【控件】选项卡中选中“Microsoft Tabbed Dialog Control 6.0”复选框，然后单击【确定】按钮。

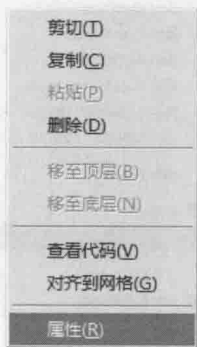




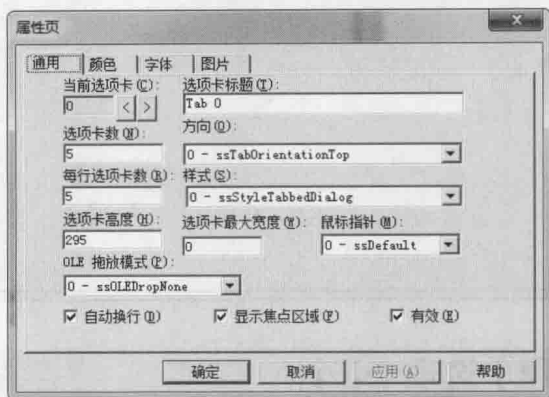
(4) 这时工具箱中就会出现 1 个 SSTab 控件。双击 SSTab 控件，把它添加到窗体中并调整其大小。



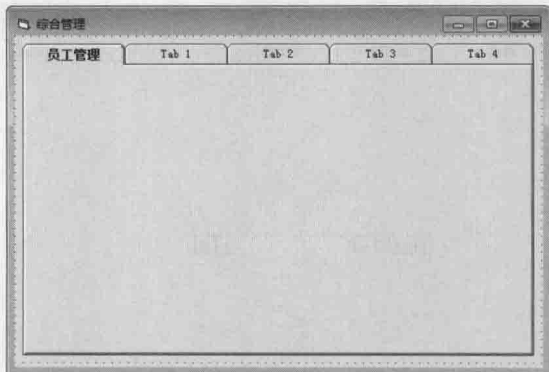
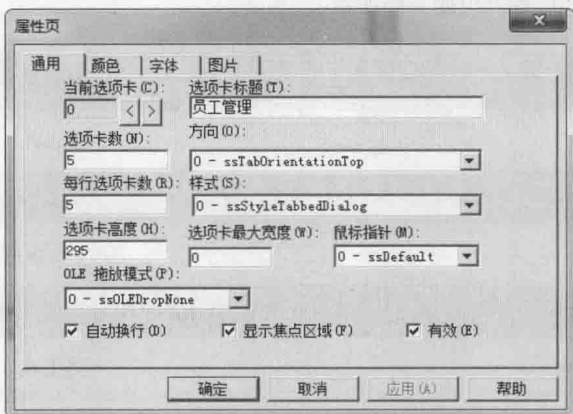
(5) 右击添加的 SSTab 控件，在弹出的快捷菜单中选择【属性】菜单项。



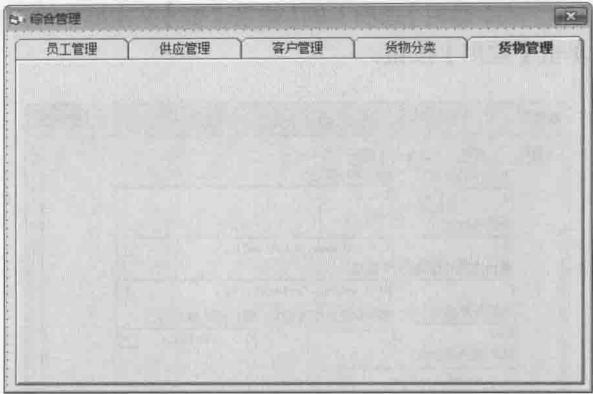
(6) 弹出【属性页】对话框，在【通用】选项卡的【选项卡数】文本框中输入“5”，在【每行选项卡数】文本框中输入“5”，然后单击【应用】按钮。



(7) 单击“当前选项卡”下面的按钮，调整当前选项卡为“0”，在【选项卡标题】文本框中输入“员工管理”，然后单击【应用】按钮，这时 SSTab 控件的第 1 个标签标题即会更改为“员工管理”。



(8) 按照上面的步骤，将其他标签依次改为“供应管理”、“客户管理”、“货物分类”和“货物管理”，然后单击【确定】按钮即可。



### 29.3.2 【员工管理】选项卡设计

【员工管理】选项卡设计的具体步骤如下。

(1) 单击选中【员工管理】选项卡，双击工具箱中的 Data 控件，在窗体中添加 1 个 Data 控件，按下表设置其属性，并调整其位置大小如下图所示。

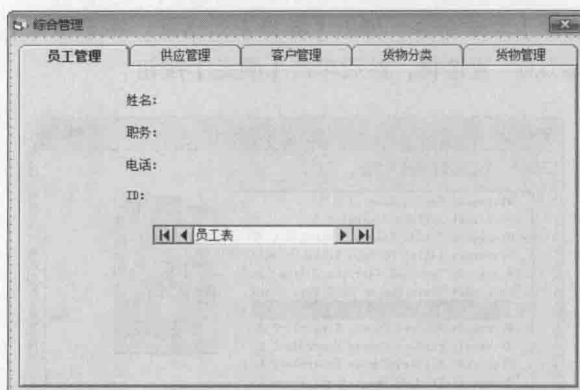
属性名称	属性值
名称	Data1
Caption	员工表
Connect	Access 2000;
DatabaseName	D:\Final\ch29\VB 小型超市管理\库存管理.mdb
RecordSource	员工表



(2) 在窗体中添加 4 个标签控件，按下表设置其属性，并调整其位置。

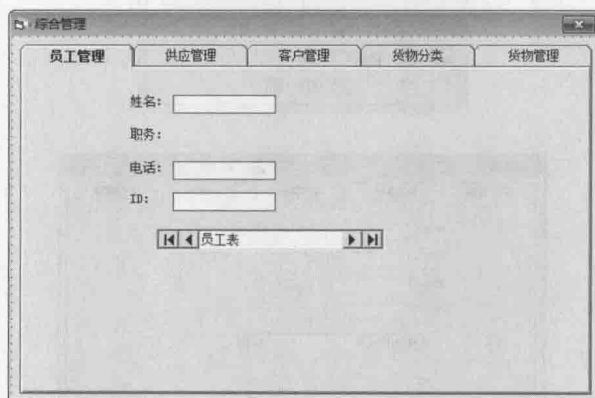


控件	名称	Caption
Label1	Label1	姓名：
Label2	Label2	职务：
Label3	Label3	电话：
Label4	Label4	ID：

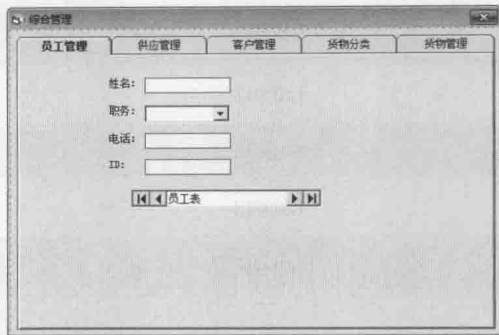


(3) 在窗体中添加 3 个文本框控件，按下表设置其属性，并调整其位置。

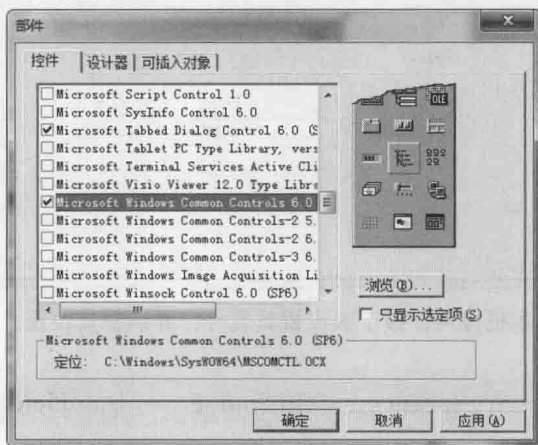
控件	名称	Appearance	DataSource	DataField	Text
Text1	Text1	0 - Flat	Data1	姓名	-
Text2	Text2	0 - Flat	Data1	电话	-
Text3	Text3	0 - Flat	Data1	ID	-



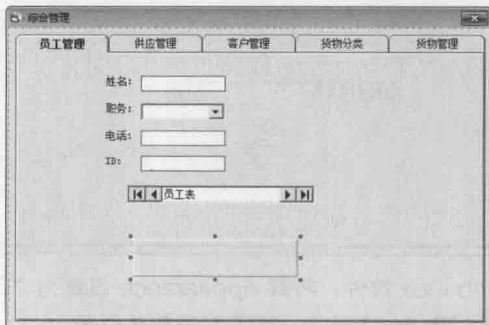
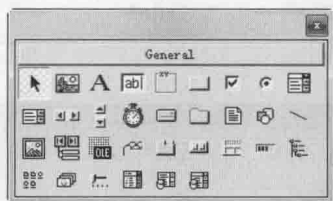
(4) 在窗体中添加 1 个 ComboBox 控件，将其 Appearance 设置为“0 - Flat”，DataSource 属性设置为“Data1”，DataField 属性设置为“职务”，按下图调整其位置。



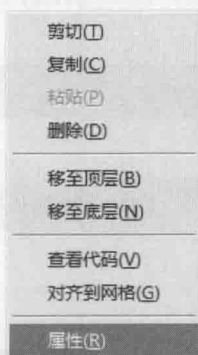
(5) 选择【工程】>【部件】菜单命令，弹出【部件】对话框，在【控件】选项卡中选中“Microsoft Windows Common Controls 6.0”复选框，然后单击【确定】按钮。



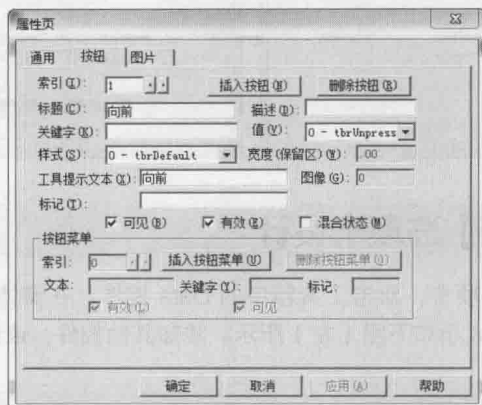
(6) 单击工具箱中的 Toolbar 控件，在窗体中适当的位置拖动添加 1 个 Toolbar 控件，并按照下图调整其大小。



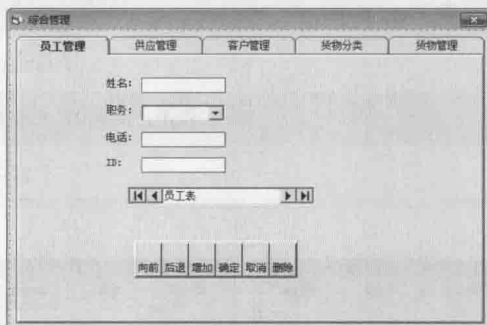
(7) 右击添加的 Toolbar 控件，在弹出的快捷菜单中选择【属性】菜单项。



(8) 弹出【属性页】对话框，选择【按钮】选项卡，单击【插入按钮】按钮，在【标题】和【工具提示文本】文本框中输入“向前”，然后单击【应用】按钮。



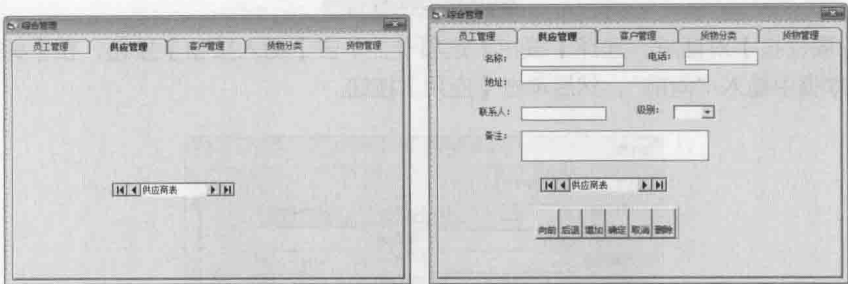
(9) 按照上面的步骤，再添加 5 个按钮，并设置其【标题】和【工具提示文本】依次为“后退”、“增加”、“确定”、“取消”和“删除”。添加完成的 Toolbar 控件如图所示。



### 29.3.3 【供应管理】选项卡设计

单击选择【供应管理】选项卡，双击工具箱中的 Data 控件，在窗体中添加 1 个 Data 控件，按下表设置其属性，并调整其位置大小如下图（左）所示。然后添加其他控件，最终效果如下图（右）所示。

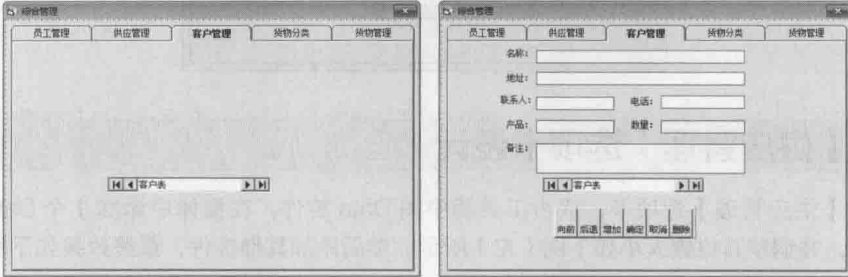
属性名称	属性值
名称	Data2
Caption	供应商表
Connect	Access 2000;
DatabaseName	D:\Final\ch29\VB 小型超市管理\库存管理.mdb
RecordSource	供应商表



### 29.3.4 【客户管理】选项卡设计

单击选择【客户管理】选项卡，双击工具箱中的 Data 控件，在窗体中添加 1 个 Data 控件，按下表设置其属性，并调整其位置大小如下图（左）所示。添加其他控件，最终效果如下图（右）所示。

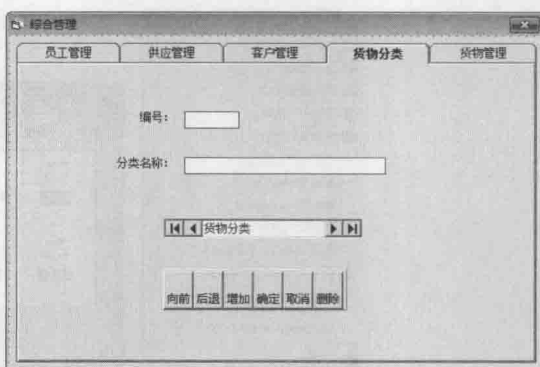
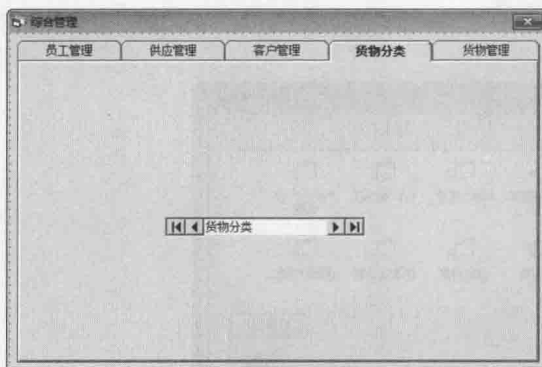
属性名称	属性值
名称	Data3
Caption	客户表
Connect	Access 2000;
DatabaseName	D:\Final\ch29\VB 小型超市管理\库存管理.mdb
RecordSource	客户表



### 29.3.5 【货物分类】选项卡设计

单击选择【货物分类】选项卡，双击工具箱中的 Data 控件，在窗体中添加 1 个 Data 控件，按下表设置其属性，并调整其位置大小如下图（左）所示。然后添加其他控件，最终效果如下图（右）所示。

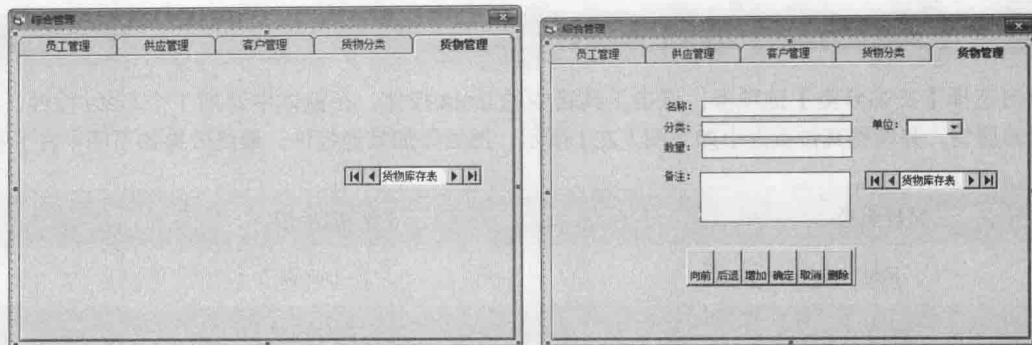
属性名称	属性值
名称	Data4
Caption	货物分类
Connect	Access 2000;
DatabaseName	D:\Final\ch29\VB 小型超市管理\库存管理.mdb
RecordSource	货物分类



### 29.3.6 【货物管理】选项卡设计

单击选择【货物管理】选项卡，双击工具箱中的 Data 控件，在窗体中添加 1 个 Data 控件，按下表设置其属性，并调整其位置大小如下图（左）所示。然后添加其他控件，最终效果如下图（右）所示。

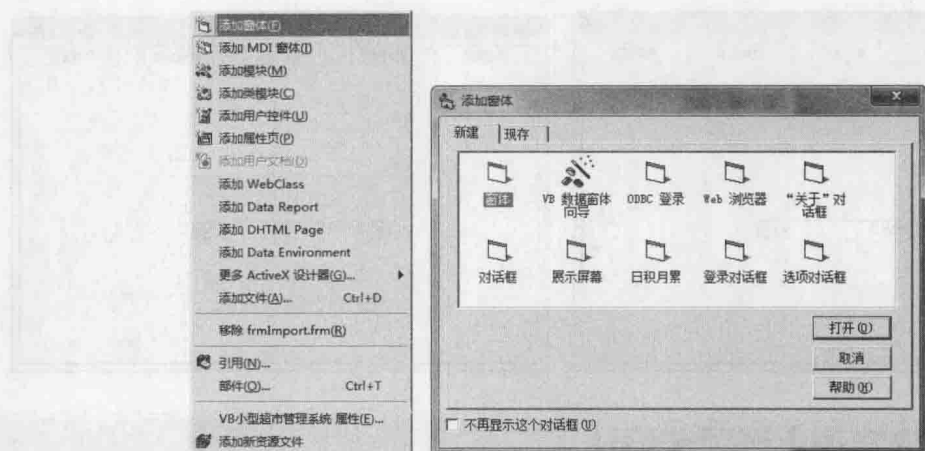
属性名称	属性值
名称	Data5
Caption	货物库存表
Connect	Access 2000;
DatabaseName	D:\Final\ch29\VB 小型超市管理\库存管理.mdb
RecordSource	货物库存表



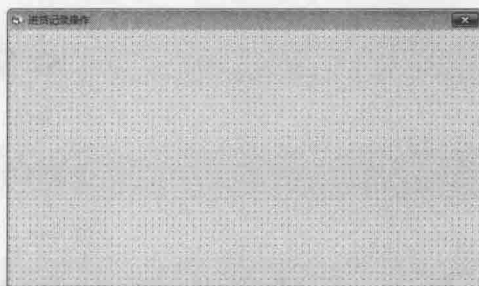
### 29.3.7 【进货记录操作】窗体设计

【进货记录操作】窗体设计的具体步骤如下。

(1) 选择【工程】>【添加窗体】菜单命令，弹出【添加窗体】对话框，在【新建】选项卡中选中“窗体”图标，然后单击【打开】按钮。



(2) 将 Form2 的名称改为“frmImport”，Caption 属性设置为“进货记录操作”。将 BorderStyle 属性设置为“1 - Fixed Single”，并调整其大小。



(3) 在窗体中添加 5 个 Data 控件，按下表设置其属性，将其 DatabaseName 属性均设置为“D:\Final\ch29\VB 小型超市管理\库存管理.mdb”，并调整其位置。

控件	名称	Caption	Connect	RecordSource
Data1	Data1	货物分类	Access 2000;	货物分类
Data2	Data2	货物库存表	Access 2000;	货物库存表
Data3	Data3	供应商表	Access 2000;	供应商表
Data4	Data4	员工表	Access 2000;	员工表
Data5	Data5	进货记录表	Access 2000;	进货记录表

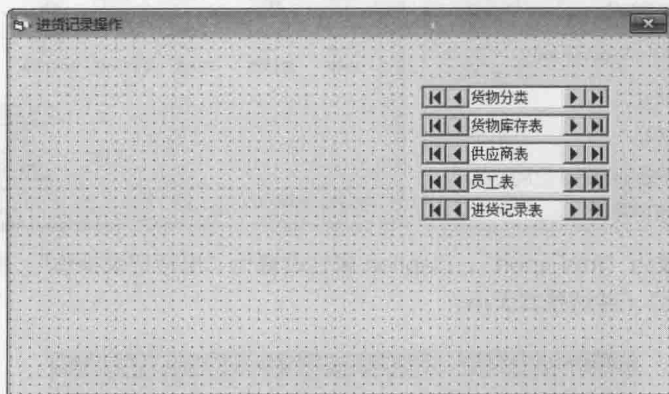


图 29-3-1 添加其他控件，最终效果如图所示。

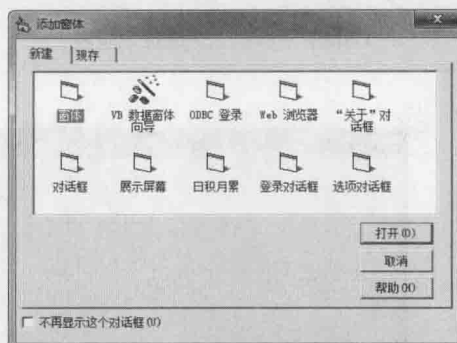


### 29.3.8 【出货记录操作】窗体设计

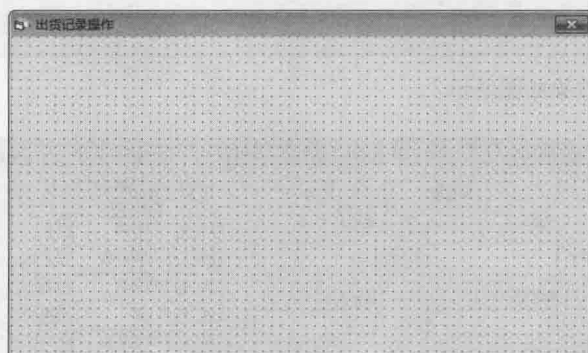
【出货记录操作】窗体设计的具体步骤如下。

(1) 选择【工程】>【添加窗体】菜单命令，弹出【添加窗体】对话框，在【新建】选项卡中选中“窗体”图标，然后单击【打开】按钮。



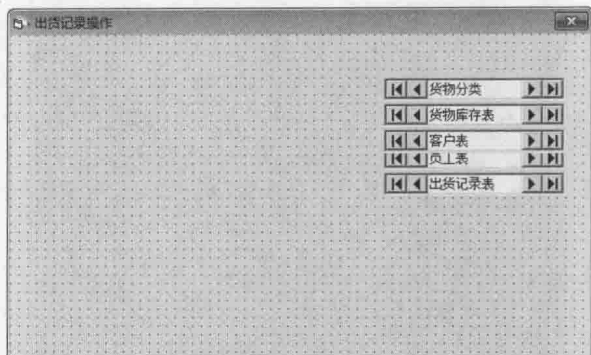


(2) 将窗体的名称改为“frmExport”，Caption 属性设置为“出货记录操作”。将 BorderStyle 属性设置为“1 - Fixed Single”，并调整其大小。



(3) 在窗体中添加 5 个 Data 控件，按下表设置其属性，将其 DatabaseName 属性均设置为“D:\Final\ch29\VB 小型超市管理\库存管理.mdb”，并调整其位置。

控件	名称	Caption	Connect	RecordSource
Data1	Data1	货物分类	Access 2000;	货物分类
Data2	Data2	货物库存表	Access 2000;	货物库存表
Data3	Data3	客户表	Access 2000;	客户表
Data4	Data4	员工表	Access 2000;	员工表
Data5	Data5	出货记录表	Access 2000;	出货记录表



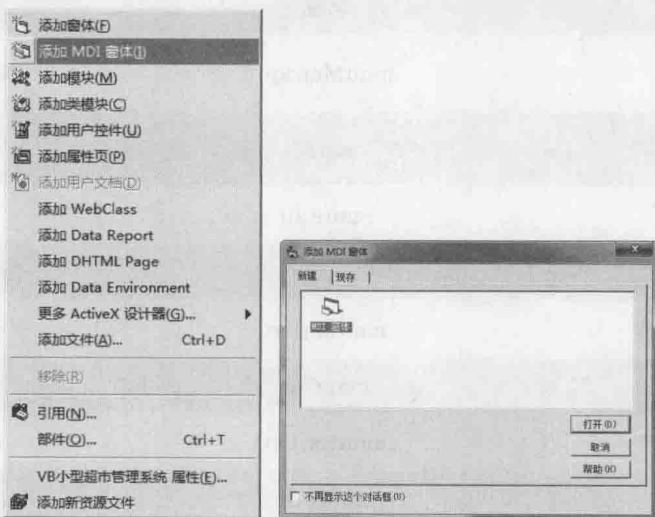
添加其他控件，最终效果如图所示。



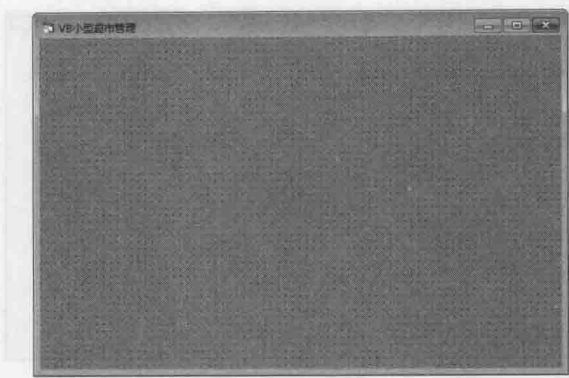
### 29.3.9 【VB 小型超市管理】主窗体设计

最后是【VB 小型超市管理】主窗体的设计，具体的操作步骤如下。

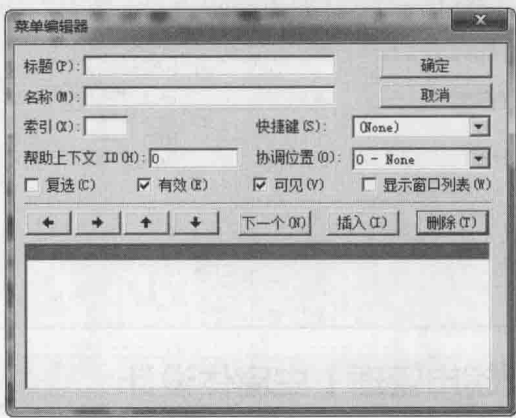
(1) 选择【工程】>【添加 MDI 窗体】菜单命令，弹出【添加 MDI 窗体】对话框，在【新建】选项卡中选中“MDI 窗体”图标，然后单击【打开】按钮。



(2) 将 MDIForm1 的名称改为“MDIFrmMain”，Caption 属性设置为“VB 小型超市管理”。

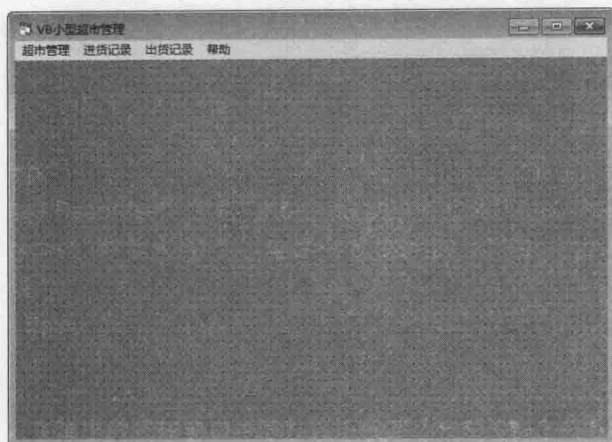


(3) 选择【工具】>【菜单编辑器】菜单命令，打开【菜单编辑器】对话框。



(4) 按下表为窗体添加菜单，标题前面带有“...”的为上一级的子菜单，添加时只需输入后面的文字即可。

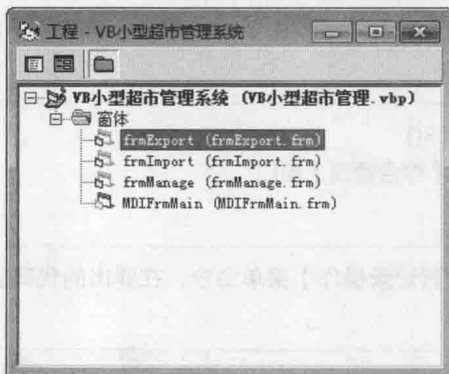
标题	名称	菜单作用
超市管理	mnuManage	“超市管理”菜单
... 综合管理	mnuAll	综合管理
... 退出管理	mnuExit	退出管理
进货记录	mnuIn	“进货记录”菜单
... 进货记录操作	mnuImport	进货记录操作
出货记录	mnuOut	“出货记录”菜单
... 出货记录操作	mnuExport	出货记录操作
帮助	mnuHelp	“帮助”菜单
... 关于	mnuAbout	显示“关于”对话框



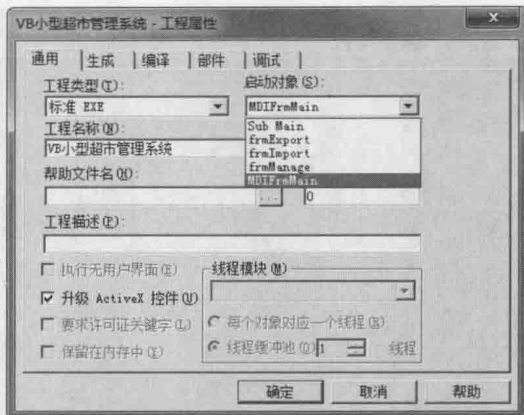
(5) 在【工程管理】窗口中选择 frmManage 窗体，将其 MDIChild 属性设置为“True”。



(6) 按照上面的步骤，将 frmImport 窗体和 frmExport 窗体的 MDIChild 属性均设置为“True”。



(7) 选择【工程】>【VB 小型超市管理系统 属性】菜单命令，弹出【VB 小型超市管理系统 - 工程属性】对话框，选择【启动对象】下拉列表中的“MDIFrmMain”选项即可。



## 29.4 系统代码编写



本节视频教学录像: 1 分钟

完成窗体的设计后, 本节为窗体添加代码。

### 29.4.1 添加【VB 小型超市管理】窗体代码

首先为【VB 小型超市管理】窗体添加代码。

(1) 在【工程管理】窗口中选择 MDIFrmMain 窗体, 然后选择【超市管理】>【综合管理】菜单命令, 在弹出的代码窗口中添加以下代码 (代码 29-1-1.txt)。

```
01 Private Sub mnuAll_Click()  
02     frmManage.Show      ' 显示【综合管理】窗口  
03 End Sub
```

(2) 选择【超市管理】>【退出管理】菜单命令, 在弹出的代码窗口中添加以下代码 (代码 29-1-2.txt)。

```
01 Private Sub mnuExit_Click()  
02     Unload Me      ' 卸载【综合管理】窗口  
03 End Sub
```

(3) 选择【进货记录】>【进货记录操作】菜单命令, 在弹出的代码窗口中添加以下代码 (代码 29-1-3.txt)。

```
01 Private Sub mnuImport_Click()  
02     frmImport.Show      ' 显示【进货记录操作】窗口  
03 End Sub
```

(4) 选择【出货记录】>【出货记录操作】菜单命令, 在弹出的代码窗口中添加以下代码 (代码 29-

1-4.txt)。

```
01 Private Sub mnuExport_Click()  
02     frmExport.Show      ' 显示【出货记录操作】窗口  
03 End Sub
```

(5) 选择【帮助】>【关于】菜单命令，在弹出的代码窗口中添加以下代码（代码 29-1-5.txt）。

```
01 Private Sub mnuAbout_Click()  
02     MsgBox "VB 小型超市管理 Ver 1.0 " & Chr(13) & Chr(13) & "VB 程序", vbInformation, "VB 小型  
超市管理 "      ' 显示【关于】对话框  
03 End Sub
```

## 29.4.2 添加【综合管理】窗体代码

接下来为【综合管理】窗体添加代码。

(1) 在【工程管理】窗口中选择 frmManage 窗体，然后双击窗体空白处，在弹出的代码窗口中添加以下代码（代码 29-2-1.txt）。

```
01 Private Sub Form_Load()  
02     SSTab1.Tab = 0  
03     Combo1.AddItem " 员工 "  
04     Combo1.AddItem " 经理 "  
05     Combo1.AddItem " 采购 "  
06     Combo1.AddItem " 库管 "  
07     Combo2.AddItem "1"  
08     Combo2.AddItem "2"  
09     Combo2.AddItem "3"  
10     Combo2.AddItem "4"  
11     Combo2.AddItem "5"  
12     Combo3.AddItem " 个 "  
13     Combo3.AddItem " 只 "  
14     Combo3.AddItem " 支 "  
15     Combo3.AddItem " 盒 "  
16     Combo3.AddItem " 本 "  
17     Combo3.AddItem " 条 "  
18     Combo3.AddItem " 瓶 "  
19     Combo3.AddItem " 张 "  
20     Combo3.AddItem " 台 "  
21     Combo3.AddItem " 块 "  
22     Combo3.AddItem " 捆 "  
23     Combo3.AddItem " 卷 "
```

```

24 Combo3.AddItem "打"
25 Combo3.AddItem "双"
26 Combo3.AddItem "对"
27 Combo3.AddItem "米"
28 Combo3.AddItem "厘米"
29 Combo3.AddItem "斤"
30 Combo3.AddItem "公斤"
31 Combo3.AddItem "克"
32 Combo3.AddItem "吨"
33 Data1.DatabaseName = App.Path & "\库存管理.mdb"
34 Data1.RecordSource = "员工表"
35 Data1.Refresh
36 Data1.Visible = False
37 Data2.DatabaseName = App.Path & "\库存管理.mdb"
38 Data2.RecordSource = "供应商表"
39 Data2.Refresh
40 Data2.Visible = False
41 Data3.DatabaseName = App.Path & "\库存管理.mdb"
42 Data3.RecordSource = "客户表"
43 Data3.Refresh
44 Data3.Visible = False
45 Data4.DatabaseName = App.Path & "\库存管理.mdb"
46 Data4.RecordSource = "货物分类"
47 Data4.Refresh
48 Data4.Visible = False
49 Data5.DatabaseName = App.Path & "\库存管理.mdb"
50 Data5.RecordSource = "货物库存表"
51 Data5.Refresh
52 Data5.Visible = False
53 End Sub

```

(2) 单击选择【员工管理】选项卡，双击 Toolbar1 控件，在弹出的代码窗口中添加以下代码 (代码 29-2-2.txt)。

```

01 Private Sub Toolbar1_ButtonClick(ByVal Button As MSComctlLib.Button)
02 Select Case Button.Index
03 Case 1 "前进" '按下【前进】按钮
04 Data1.Recordset.MovePrevious
05 If Data1.Recordset.BOF = True Then
06 Data1.Recordset.MoveFirst
07 End If
08 Case 2 "后退" '按下【后退】按钮

```



```

09 Data1.Recordset.MoveNext
10 If Data1.Recordset.EOF = True Then
11 Data1.Recordset.MoveLast
12 End If
13 Case 3 "增加" '按下【增加】按钮
14 On Error GoTo add_error
15 Data1.Recordset.AddNew
16 add_error:
17 If Err.Number = 3426 Then
18 MsgBox "已经使用过添加记录! 错误: "& Err & ", "& Err.Description, 0, "提示"
19 Unload Me
20 End If
21 Toolbar1.Buttons(1).Visible = False
22 Toolbar1.Buttons(2).Visible = False
23 Toolbar1.Buttons(3).Visible = False
24 Toolbar1.Buttons(4).Visible = True
25 Toolbar1.Buttons(5).Visible = True
26 Toolbar1.Buttons(6).Visible = False
27 Case 4 "确定" '按下【确定】按钮
28 On Error GoTo update_error
29 If Text3.Text = "" Then
30 MsgBox "必须填写! ", 0, "提示"
31 Exit Sub
32 End If
33 Data1.UpdateRecord
34 Data1.Recordset.Bookmark = Data1.Recordset.LastModified
35 update_error:
36 If Err.Number = 3020 Then
37 MsgBox "请先添加记录! 错误: "& Err & ", "& Err.Description, 0, "提示"
38 Unload Me
39 End If
40 Toolbar1.Buttons(1).Visible = True
41 Toolbar1.Buttons(2).Visible = True
42 Toolbar1.Buttons(3).Visible = True
43 Toolbar1.Buttons(4).Visible = True
44 Toolbar1.Buttons(5).Visible = True
45 Toolbar1.Buttons(6).Visible = True
46 Case 5 "取消" '按下【取消】按钮
47 On Error GoTo CanceErr
48 Data1.Recordset.Edit
49 Data1.Recordset.CancelUpdate

```

```

50 Toolbar1.Buttons(1).Visible = True
51 Toolbar1.Buttons(2).Visible = True
52 Toolbar1.Buttons(3).Visible = True
53 Toolbar1.Buttons(4).Visible = True
54 Toolbar1.Buttons(5).Visible = True
55 Toolbar1.Buttons(6).Visible = True
56 CancelErr:
57 If Err.Number = 3021 Then
58     MsgBox "没有记录! 错误: " & Err & ", " & Err.Description, 0, "提示"
59 End If
60 Case 6 "删除" '按下【删除】按钮
61 On Error GoTo del_error
62     a = MsgBox("真的删除吗?", vbExclamation + vbOKCancel + vbApplicationModal, "删除记录")
63 If a = 1 Then
64     Data1.Recordset.Delete
65     Data1.Recordset.MoveLast
66 End If
67 If a = 2 Then Exit Sub
68 del_error:
69 If Err.Number = 3426 Then
70     MsgBox "已经全部删除! 错误: " & Err & ", " & Err.Description, 0, "提示"
71 Unload Me
72 End If
73 End Select
74 End Sub

```

(3) 单击选择【供应管理】选项卡，双击 Toolbar2 控件，在弹出的代码窗口中添加以下代码（代码 29-2-3.txt）。

```

01 Private Sub Toolbar2_ButtonClick(ByVal Button As MSComctlLib.Button)
02     Select Case Button.Index
03     Case 1 '前进
04         Data2.Recordset.MovePrevious
05     If Data2.Recordset.BOF = True Then
06         Data2.Recordset.MoveFirst
07     End If
08     Case 2 '后退
09         Data2.Recordset.MoveNext
10     If Data2.Recordset.EOF = True Then
11         Data2.Recordset.MoveLast

```

```

12 End If
13 Case 3 ' 增加
14 On Error GoTo add_error
15 Data2.Recordset.AddNew
16 add_error:
17 If Err.Number = 3426 Then
18     MsgBox "已经使用过添加记录! 错误: " & Err & ", " & Err.Description, 0, "提示"
19     Unload Me
20 End If
21 Toolbar2.Buttons(1).Visible = False
22 Toolbar2.Buttons(2).Visible = False
23 Toolbar2.Buttons(3).Visible = False
24 Toolbar2.Buttons(4).Visible = True
25 Toolbar2.Buttons(5).Visible = True
26 Toolbar2.Buttons(6).Visible = False
27 Case 4 ' 确定
28 On Error GoTo update_error
29 If Text3.Text = "" Then
30     MsgBox "必须填写售电员! ", 0, "提示"
31 Exit Sub
32 End If
33 Data2.UpdateRecord
34 Data2.Recordset.Bookmark = Data2.Recordset.LastModified
35 update_error:
36 If Err.Number = 3020 Then
37     MsgBox "请先添加记录! 错误: " & Err & ", " & Err.Description, 0, "提示"
38     Unload Me
39 End If
40 Toolbar2.Buttons(1).Visible = True
41 Toolbar2.Buttons(2).Visible = True
42 Toolbar2.Buttons(3).Visible = True
43 Toolbar2.Buttons(4).Visible = True
44 Toolbar2.Buttons(5).Visible = True
45 Toolbar2.Buttons(6).Visible = True
46 Case 5 ' 取消
47 On Error GoTo CanceErr
48 Data2.Recordset.Edit
49 Data2.Recordset.CancelUpdate
50 Toolbar2.Buttons(1).Visible = True

```

```

51 Toolbar2.Buttons(2).Visible = True
52 Toolbar2.Buttons(3).Visible = True
53 Toolbar1.Buttons(4).Visible = True
54 Toolbar2.Buttons(5).Visible = True
55 Toolbar2.Buttons(6).Visible = True
56 CancelErr:
57 If Err.Number = 3021 Then
58     MsgBox "没有记录! 错误: " & Err & ", " & Err.Description, 0, "提示"
59 End If
60 Case 6 ' 删除
61 On Error GoTo del_error
62     a = MsgBox("真的删除吗? ", vbExclamation + vbOKCancel + vbApplicationModal, "删除记录")
63 If a = 1 Then
64     Data2.Recordset.Delete
65     Data2.Recordset.MoveLast
66 End If
67 If a = 2 Then Exit Sub
68 del_error:
69 If Err.Number = 3426 Then
70     MsgBox "已经全部删除! 错误: " & Err & ", " & Err.Description, 0, "提示"
71 Unload Me
72 End If
73 End Select
74 End Sub

```

(4) 单击选择【客户管理】选项卡，双击 Toolbar3 控件，在弹出的代码窗口中添加以下代码（代码 29-2-4.txt）。

```

01 Private Sub Toolbar3_ButtonClick(ByVal Button As MSComctlLib.Button)
02     Select Case Button.Index
03     Case 1 ' 前进
04         Data3.Recordset.MovePrevious
05     If Data3.Recordset.BOF = True Then
06         Data3.Recordset.MoveFirst
07     End If
08     Case 2 ' 后退
09         Data3.Recordset.MoveNext
10     If Data3.Recordset.EOF = True Then
11         Data3.Recordset.MoveLast
12     End If
13     Case 3 ' 增加

```

```

14 On Error GoTo add_error
15 Data3.Recordset.AddNew
16 add_error:
17 If Err.Number = 3426 Then
18     MsgBox "已经使用过添加记录! 错误: "& Err & ", "& Err.Description, 0, "提示"
19     Unload Me
20 End If
21 Toolbar3.Buttons(1).Visible = False
22 Toolbar3.Buttons(2).Visible = False
23 Toolbar3.Buttons(3).Visible = False
24 Toolbar3.Buttons(4).Visible = True
25 Toolbar3.Buttons(5).Visible = True
26 Toolbar3.Buttons(6).Visible = False
27 Case 4 ' 确定
28 On Error GoTo update_error
29 If Text3.Text = "" Then
30     MsgBox "必须填写售电员! ", 0, "提示"
31 Exit Sub
32 End If
33 Data3.UpdateRecord
34 Data3.Recordset.Bookmark = Data3.Recordset.LastModified
35 update_error:
36 If Err.Number = 3020 Then
37     MsgBox "请先添加记录! 错误: "& Err & ", "& Err.Description, 0, "提示"
38     Unload Me
39 End If
40 Toolbar3.Buttons(1).Visible = True
41 Toolbar3.Buttons(2).Visible = True
42 Toolbar3.Buttons(3).Visible = True
43 Toolbar3.Buttons(4).Visible = True
44 Toolbar3.Buttons(5).Visible = True
45 Toolbar3.Buttons(6).Visible = True
46 Case 5 ' 取消
47 On Error GoTo CanceErr
48 Data1.Recordset.Edit
49 Data1.Recordset.CancelUpdate
50 Toolbar3.Buttons(1).Visible = True
51 Toolbar3.Buttons(2).Visible = True
52 Toolbar3.Buttons(3).Visible = True
53 Toolbar3.Buttons(4).Visible = True

```

```

54 Toolbar3.Buttons(5).Visible = True
55 Toolbar3.Buttons(6).Visible = True
56 CanceErr:
57 If Err.Number = 3021 Then
58     MsgBox " 没有记录!  错误: " & Err & ", " & Err.Description, 0, " 提示"
59 End If
60 Case 6 ' 删除
61 On Error GoTo del_error
62     a = MsgBox("真的删除吗? ", vbExclamation + vbOKCancel + vbApplicationModal, "删除记录")
63 If a = 1 Then
64     Data3.Recordset.Delete
65     Data3.Recordset.MoveLast
66 End If
67 If a = 2 Then Exit Sub
68 del_error:
69 If Err.Number = 3426 Then
70     MsgBox " 已经全部删除!  错误: " & Err & ", " & Err.Description, 0, " 提示"
71 Unload Me
72 End If
73 End Select
74 End Sub

```

(5) 单击选择【货物分类】选项卡，双击 Toolbar4 控件，在弹出的代码窗口中添加以下代码（代码 29-2-5.txt）。

```

01 Private Sub Toolbar4_ButtonClick(ByVal Button As MSComctlLib.Button)
02 Select Case Button.Index
03 Case 1 ' 前进
04     Data4.Recordset.MovePrevious
05 If Data4.Recordset.BOF = True Then
06     Data4.Recordset.MoveFirst
07 End If
08 Case 2 ' 后退
09     Data4.Recordset.MoveNext
10 If Data4.Recordset.EOF = True Then
11     Data4.Recordset.MoveLast
12 End If
13 Case 3 ' 增加
14 On Error GoTo add_error
15     Data4.Recordset.AddNew
16 add_error:

```

```

17   If Err.Number = 3426 Then
18     MsgBox " 已经使用过添加记录!  错误: " & Err & ", " & Err.Description, 0, " 提示 "
19     Unload Me
20   End If
21   Toolbar4.Buttons(1).Visible = False
22   Toolbar4.Buttons(2).Visible = False
23   Toolbar4.Buttons(3).Visible = False
24   Toolbar4.Buttons(4).Visible = True
25   Toolbar4.Buttons(5).Visible = True
26   Toolbar4.Buttons(6).Visible = False
27   Case 4 ' 确定
28     On Error GoTo update_error
29     If Text3.Text = "" Then
30       MsgBox " 必须填写售电员! ", 0, " 提示 "
31     Exit Sub
32   End If
33   Data4.UpdateRecord
34   Data4.Recordset.Bookmark = Data1.Recordset.LastModified
35   update_error:
36   If Err.Number = 3020 Then
37     MsgBox " 请先添加记录!  错误: " & Err & ", " & Err.Description, 0, " 提示 "
38     Unload Me
39   End If
40   Toolbar4.Buttons(1).Visible = True
41   Toolbar4.Buttons(2).Visible = True
42   Toolbar4.Buttons(3).Visible = True
43   Toolbar4.Buttons(4).Visible = True
44   Toolbar4.Buttons(5).Visible = True
45   Toolbar4.Buttons(6).Visible = True
46   Case 5 ' 取消
47     On Error GoTo CanceErr
48     Data4.Recordset.Edit
49     Data4.Recordset.CancelUpdate
50     Toolbar4.Buttons(1).Visible = True
51     Toolbar4.Buttons(2).Visible = True
52     Toolbar4.Buttons(3).Visible = True
53     Toolbar4.Buttons(4).Visible = True
54     Toolbar4.Buttons(5).Visible = True
55     Toolbar4.Buttons(6).Visible = True
56     CanceErr:

```



```

57 If Err.Number = 3021 Then
58     MsgBox " 没有记录!  错误: " & Err & ", " & Err.Description, 0, " 提示 "
59 End If
60 Case 6 ' 删除
61 On Error GoTo del_error
62     a = MsgBox("真的删除吗? ", vbExclamation + vbOKCancel + vbApplicationModal, "删除记录")
63 If a = 1 Then
64     Data4.Recordset.Delete
65     Data4.Recordset.MoveLast
66 End If
67 If a = 2 Then Exit Sub
68 del_error:
69     If Err.Number = 3426 Then
70         MsgBox " 已经全部删除!  错误: " & Err & ", " & Err.Description, 0, " 提示 "
71         Unload Me
72     End If
73 End Select
74 End Sub

```

(6) 单击选择【货物管理】选项卡，双击 Toolbar5 控件，在弹出的代码窗口中添加以下代码（代码 29-2-6.txt）。

```

01 Private Sub Toolbar5_ButtonClick(ByVal Button As MSComctlLib.Button)
02     Select Case Button.Index
03     Case 1 ' 前进
04         Data5.Recordset.MovePrevious
05         If Data5.Recordset.BOF = True Then
06             Data5.Recordset.MoveFirst
07         End If
08     Case 2 ' 后退
09         Data5.Recordset.MoveNext
10         If Data5.Recordset.EOF = True Then
11             Data5.Recordset.MoveLast
12         End If
13     Case 3 ' 增加
14         On Error GoTo add_error
15         Data5.Recordset.AddNew
16     add_error:
17         If Err.Number = 3426 Then
18             MsgBox " 已经使用过添加记录!  错误: " & Err & ", " & Err.Description, 0, " 提示 "
19             Unload Me

```

```

20 End If
21 Toolbar5.Buttons(1).Visible = False
22 Toolbar5.Buttons(2).Visible = False
23 Toolbar5.Buttons(3).Visible = False
24 Toolbar5.Buttons(4).Visible = True
25 Toolbar5.Buttons(5).Visible = True
26 Toolbar5.Buttons(6).Visible = False
27 Case 4 ' 确定
28 On Error GoTo update_error
29 If Text3.Text = "" Then
30 MsgBox " 必须填写! ", 0, " 提示 "
31 Exit Sub
32 End If
33 Data5.Recordset(" 分类 ") = DBCombo1.Text
34 Data5.UpdateRecord
35 Data5.Recordset.Bookmark = Data1.Recordset.LastModified
36 update_error:
37 If Err.Number = 3020 Then
38 MsgBox " 请先添加记录! 错误: " & Err & ", " & Err.Description, 0, " 提示 "
39 Unload Me
40 End If
41 Text21.Visible = True
42 Toolbar5.Buttons(1).Visible = True
43 Toolbar5.Buttons(2).Visible = True
44 Toolbar5.Buttons(3).Visible = True
45 Toolbar5.Buttons(4).Visible = False
46 Toolbar5.Buttons(5).Visible = False
47 Toolbar5.Buttons(6).Visible = True
48 Case 5 ' 取消
49 On Error GoTo CanceErr
50 Data5.Recordset.Edit
51 Data5.Recordset.CancelUpdate
52 Text21.Visible = True
53 Toolbar5.Buttons(1).Visible = True
54 Toolbar5.Buttons(2).Visible = True
55 Toolbar5.Buttons(3).Visible = True
56 Toolbar5.Buttons(4).Visible = False
57 Toolbar5.Buttons(5).Visible = False
58 Toolbar5.Buttons(6).Visible = True
59 CanceErr:

```

```

60 If Err.Number = 3021 Then
61     MsgBox "没有记录! 错误: " & Err & ", " & Err.Description, 0, "提示"
62 End If
63 Case 6 '删除
64 On Error GoTo del_error
65     a = MsgBox("真的删除吗?", vbExclamation + vbOKCancel + vbApplicationModal, "删除记录")
66 If a = 1 Then
67     Data5.Recordset.Delete
68     Data5.Recordset.MoveLast
69 End If
70 If a = 2 Then Exit Sub
71 del_error:
72 If Err.Number = 3426 Then
73     MsgBox "已经全部删除! 错误: " & Err & ", " & Err.Description, 0, "提示"
74 Unload Me
75 End If
76 End Select
77 End Sub

```

### 29.4.3 添加【进货记录操作】窗体代码

本小节为【进货记录操作】窗体添加代码。

(1) 在【工程管理】窗口中选择 frmImport 窗体, 然后双击窗体空白处, 在弹出的代码窗口中添加以下代码(代码 29-3-1.txt)。

```

01 Private Sub Form_Load()
02     Data1.DatabaseName = App.Path & "\库存管理.mdb"
03     Data1.RecordSource = "货物分类"
04     Data1.Refresh
05     Data1.Visible = False
06     Data2.DatabaseName = App.Path & "\库存管理.mdb"
07     Data2.RecordSource = "货物库存表"
08     Data2.Refresh
09     Data2.Visible = False
10     Data3.DatabaseName = App.Path & "\库存管理.mdb"
11     Data3.RecordSource = "供应商表"
12     Data3.Refresh
13     Data3.Visible = False
14     Data4.DatabaseName = App.Path & "\库存管理.mdb"
15     Data4.RecordSource = "员工表"

```

```

16 Data4.Refresh
17 Data4.Visible = False
18 Data5.DatabaseName = App.Path & "\库存管理.mdb"
19 Data5.RecordSource = "进货记录表"
20 Data5.Refresh
21 Data5.Visible = False
22 End Sub

```

(2) 双击【确定】按钮，在弹出的代码窗口中添加以下代码（代码 29-3-2.txt）。

```

01 Private Sub cmdOK_Click()
02 If Val(Text1.Text) <= 0 Then
03 MsgBox "请输入进货数量！"
04 Exit Sub
05 End If
06 If DBCombo2.Text = "" Then
07 MsgBox "请选择进货货物名称！"
08 Exit Sub
09 End If
10 If DBCombo4.Text = "" Then
11 MsgBox "请选择经手人！"
12 Exit Sub
13 End If
14 Dim a As Integer
15 a = MsgBox("***** 你确定此操作吗？*****" & vbCrLf _
16     & "名称：" & DBCombo2.Text & vbCrLf _
17     & "原库存量：" & Label4.Caption & Label5.Caption & vbCrLf _
18     & "本次进货：" & Text1.Text & Label5.Caption & " 单价：" & Text2.Text & " 元" & vbCrLf _
19     & "供应商：" & DBCombo3.Text & vbCrLf _
20     & "经手人：" & DBCombo4.Text & vbCrLf _
21     , vbExclamation + vbOKCancel + vbApplicationModal, "提示")
22 If a = 1 Then
23 Data5.Recordset.AddNew
24 Data5.Recordset("名称") = DBCombo2.Text
25 Data5.Recordset("供应商") = DBCombo3.Text
26 Data5.Recordset("数量") = Text1.Text
27 Data5.Recordset("单价") = Text2.Text
28 Data5.Recordset("经手") = DBCombo4.Text
29 Data5.Recordset("日期") = Date
30 Data5.Recordset("时间") = Time
31 Data5.UpdateRecord

```

```

32 Data5.Recordset.Bookmark = Data5.Recordset.LastModified
33 Data2.Recordset.Edit
34 Data2.Recordset("数量") = Val(Label4.Caption) + Val(Text1.Text)
35 Data2.Recordset.Update
36 MsgBox "进货成功! "
37 End If
38 If a = 2 Then Exit Sub
39 End Sub

```

(3) 双击【取消】按钮，在弹出的代码窗口中添加以下代码（代码 29-3-3.txt）。

```

01 Private Sub cmdCancel_Click()
02 Unload Me
03 End Sub

```

(4) 双击 Timer1 控件，在弹出的代码窗口中添加以下代码（代码 29-3-4.txt）。

```

01 Private Sub Timer1_Timer()
02 Data2.Recordset.FindFirst "名称 =" & DBCombo2.Text & ""
03 If Data2.Recordset.NoMatch = True Then
04 MsgBox "Err", 0, "提示"
05 Timer1.Enabled = False
06 Exit Sub
07 Timer1.Enabled = False
08 End If
09 Label4.Caption = Data2.Recordset("数量")
10 Label5.Caption = Data2.Recordset("单位")
11 Timer1.Enabled = False
12 Dim MyStr As String
13 MyStr = CStr(DBCombo2.Text) & " 原库存量" & CStr(Label4.Caption) & CStr(Label5.Caption)
14 End Sub

```

(5) 双击 Timer2 控件，在弹出的代码窗口中添加以下代码（代码 29-3-5.txt）。

```

01 Private Sub Timer2_Timer()
02 If DBCombo3.Text = "" Then
03 MsgBox "没有选择供应商! "
04 Timer2.Enabled = False
05 Exit Sub
06 End If
07 Data3.Recordset.FindFirst "名称 =" & DBCombo3.Text & ""
08 If Data3.Recordset.NoMatch = True Then

```

```

09  MsgBox "Err", 0, "提示"
10  Timer2.Enabled = False
11  Exit Sub
12 End If
13 Dim MyStr As String
14  MyStr = CStr(Data3.Recordset("名称")) & " 地址:" & CStr(Data3.Recordset("地址")) & " 联
系人:" & CStr(Data3.Recordset("联系人")) & " 电话:" & CStr(Data3.Recordset("电话")) & " 备注:" &
CStr(Data3.Recordset("备注"))
15  Timer2.Enabled = False
16 End Sub

```

(6) 双击 Timer3 控件，在弹出的代码窗口中添加以下代码（代码 29-3-6.txt）。

```

01 Private Sub Timer3_Timer()
02 If DBCombo4.Text = "" Then
03  MsgBox "没有选择供应商！"
04  Timer3.Enabled = False
05  Exit Sub
06 End If
07 Data4.Recordset.FindFirst "姓名=" & DBCombo4.Text & ""
08 If Data4.Recordset.NoMatch = True Then
09  MsgBox "Err", 0, "提示"
10  Timer3.Enabled = False
11  Exit Sub
12 End If
13 Dim MyStr As String
14  MyStr = CStr(Data4.Recordset("姓名")) & " 职务:" & CStr(Data4.Recordset("职务")) & " 电话:"
& CStr(Data4.Recordset("电话"))
15  Timer2.Enabled = False
16 End Sub

```

(7) 在代码窗口中添加其他的相关代码（代码 29-3-7.txt）。

```

01 Option Explicit
02 Private Sub DBCombo2_Click(Area As Integer)
03 If DBCombo1.Text = "" Then
04  MsgBox "请先选择类别！"
05  Exit Sub
06 End If
07 Dim SQL As String
08  SQL = "SELECT ID, 分类, 单位, 名称, 备注, 数量 FROM 货物库存表 WHERE 分类=" &
CStr(DBCombo1.Text) & ""

```

```

09 Data2.Visible = False
10 Data2.DatabaseName = App.Path & "\库存管理 .mdb"
11 Data2.RecordSource = SQL
12 Data2.Refresh
13 DBCombo2.Refresh
14 Timer1.Enabled = True
15 Timer1.Interval = 2000
16 End Sub
17 Private Sub DBCombo3_Click(Area As Integer)
18 Timer2.Enabled = True
19 Timer2.Interval = 2000
20 End Sub
21 Private Sub DBCombo4_Click(Area As Integer)
22 Timer3.Enabled = True
23 Timer3.Interval = 2000
24 End Sub

```

#### 29.4.4 添加【出货记录操作】窗体代码

本小节为【出货记录操作】窗体添加代码。

(1) 在【工程管理】窗口中, 选择 frmExport 窗体, 双击窗体空白处, 在弹出的代码窗口中添加以下代码 (代码 29-4-1.txt)。

```

01 Private Sub Form_Load()
02 Data1.DatabaseName = App.Path & "\库存管理 .mdb"
03 Data1.RecordSource = "货物分类"
04 Data1.Refresh
05 Data1.Visible = False
06 Data2.DatabaseName = App.Path & "\库存管理 .mdb"
07 Data2.RecordSource = "货物库存表"
08 Data2.Refresh
09 Data2.Visible = False
10 Data3.DatabaseName = App.Path & "\库存管理 .mdb"
11 Data3.RecordSource = "客户表"
12 Data3.Refresh
13 Data3.Visible = False
14 Data4.DatabaseName = App.Path & "\库存管理 .mdb"
15 Data4.RecordSource = "员工表"
16 Data4.Refresh
17 Data4.Visible = False

```



```

18 Data5.DatabaseName = App.Path & "\库存管理.mdb"
19 Data5.RecordSource = "出货记录表"
20 Data5.Refresh
21 Data5.Visible = False
22 End Sub

```

(2) 双击【确定】按钮，在弹出的代码窗口中添加以下代码 (代码 29-4-2.txt)。

```

01 Private Sub cmdOK_Click()
02 If Val(Label4.Caption) - Val(Text1.Text) < 0 Then
03 MsgBox "库存量不够! 请及时采购。"
04 Exit Sub
05 End If
06 If DBCombo2.Text = "" Then
07 MsgBox "请选择出货货物名称!"
08 Exit Sub
09 End If
10 If DBCombo4.Text = "" Then
11 MsgBox "请选择经手人!"
12 Exit Sub
13 End If
14 Dim a As Integer
15 a = MsgBox("***** 你确定此操作吗? *****" & vbCrLf _
16     & "名称:" & DBCombo2.Text & vbCrLf _
17     & "原库存量:" & Label4.Caption & Label5.Caption & vbCrLf _
18     & "本次出货:" & Text1.Text & Label5.Caption & vbCrLf _
19     & "服务客户:" & DBCombo3.Text & vbCrLf _
20     & "经手人:" & DBCombo4.Text & vbCrLf _
21     , vbExclamation + vbOKCancel + vbApplicationModal, "提示")
22 If a = 1 Then
23 Data5.Recordset.AddNew
24 Data5.Recordset("名称") = DBCombo2.Text
25 Data5.Recordset("用途") = DBCombo3.Text
26 Data5.Recordset("数量") = Text1.Text
27 Data5.Recordset("经手人") = DBCombo4.Text
28 Data5.Recordset("日期") = Date
29 Data5.Recordset("时间") = Time
30 Data5.UpdateRecord
31 Data5.Recordset.Bookmark = Data5.Recordset.LastModified
32 Data2.Recordset.Edit
33 Data2.Recordset("数量") = Val(Label4.Caption) - Val(Text1.Text)

```

```

34 Data2.Recordset.Update
35 MsgBox " 出货完成! "
36 End If
37 End Sub

```

(3) 双击【取消】按钮，在弹出的代码窗口中添加以下代码 (代码 29-4-3.txt)。

```

01 Private Sub cmdCancel_Click()
02 Unload Me
03 End Sub

```

(4) 双击 Timer1 控件，在弹出的代码窗口中添加以下代码 (代码 29-4-4.txt)。

```

01 Private Sub Timer1_Timer()
02 Data2.Recordset.FindFirst " 名称 =" & DBCombo2.Text & ""
03 If Data2.Recordset.NoMatch = True Then
04 MsgBox "Err", 0, " 提示 "
05 Timer1.Enabled = False
06 Exit Sub
07 Timer1.Enabled = False
08 End If
09 Label4.Caption = Data2.Recordset(" 数量 ")
10 Label5.Caption = Data2.Recordset(" 单位 ")
11 Timer1.Enabled = False
12 Dim MyStr As String
13 MyStr = CStr(DBCombo2.Text) & " 原库存量 " & CStr(Label4.Caption) & CStr(Label5.Caption)
14 End Sub

```

(5) 双击 Timer2 控件，在弹出的代码窗口中添加以下代码 (代码 29-4-5.txt)。

```

01 Private Sub Timer2_Timer()
02 If DBCombo3.Text = "" Then
03 MsgBox " 没有选择服务客户! "
04 Timer2.Enabled = False
05 Exit Sub
06 End If
07 Data3.Recordset.FindFirst " 名称 =" & DBCombo3.Text & ""
08 If Data3.Recordset.NoMatch = True Then
09 MsgBox "Err", 0, " 提示 "
10 Timer2.Enabled = False
11 Exit Sub
12 End If

```

```

13 Dim MyStr As String
14 MyStr = CStr(Data3.Recordset("名称")) & " 地址: " & CStr(Data3.Recordset("地址")) & " 联系人: " & CStr(Data3.Recordset("联系人")) & " 电话: " & CStr(Data3.Recordset("电话")) & " 需求产品: " & CStr(Data3.Recordset("产品")) & " 数量: " & CStr(Data3.Recordset("数量")) & " 备注: " & CStr(Data3.Recordset("备注"))
15 Timer2.Enabled = False
16 End Sub

```

(6) 双击 Timer3 控件，在弹出的代码窗口中添加以下代码 (代码 29-4-6.txt)。

```

01 Private Sub Timer3_Timer()
02 If DBCombo4.Text = "" Then
03 MsgBox "没有选择经手人! "
04 Timer3.Enabled = False
05 Exit Sub
06 End If
07 Data4.Recordset.FindFirst "姓名=" & DBCombo4.Text & ""
08 If Data4.Recordset.NoMatch = True Then
09 MsgBox "Err", 0, "提示"
10 Timer3.Enabled = False
11 Exit Sub
12 End If
13 Dim MyStr As String
14 MyStr = CStr(Data4.Recordset("姓名")) & " 职务: " & CStr(Data4.Recordset("职务")) & " 电话: " & CStr(Data4.Recordset("电话"))
15 Timer3.Enabled = False
16 End Sub

```

(7) 在代码窗口中添加其他的相关代码 (代码 29-4-7.txt)。


```

01 Private Sub DBCombo2_Click(Area As Integer)
02 If DBCombo1.Text = "" Then
03 MsgBox "请先选择类别! "
04 Exit Sub
05 End If
06 Dim SQL As String
07 SQL = "SELECT ID, 分类, 单位, 名称, 备注, 数量 FROM 货物库存表 WHERE 分类 = " & CStr(DBCombo1.Text) & ""
08 Data2.Visible = False
09 Data2.DatabaseName = App.Path & "\库存管理.mdb"
10 Data2.RecordSource = SQL
11 Data2.Refresh

```

```
12 DBCombo2.Refresh
13 Timer1.Enabled = True
14 Timer1.Interval = 2000
15 End Sub
16 Private Sub DBCombo3_Click(Area As Integer)
17 Timer2.Enabled = True
18 Timer2.Interval = 2000
19 End Sub
20 Private Sub DBCombo4_Click(Area As Integer)
21 Timer3.Enabled = True
22 Timer3.Interval = 2000
23 End Sub
```

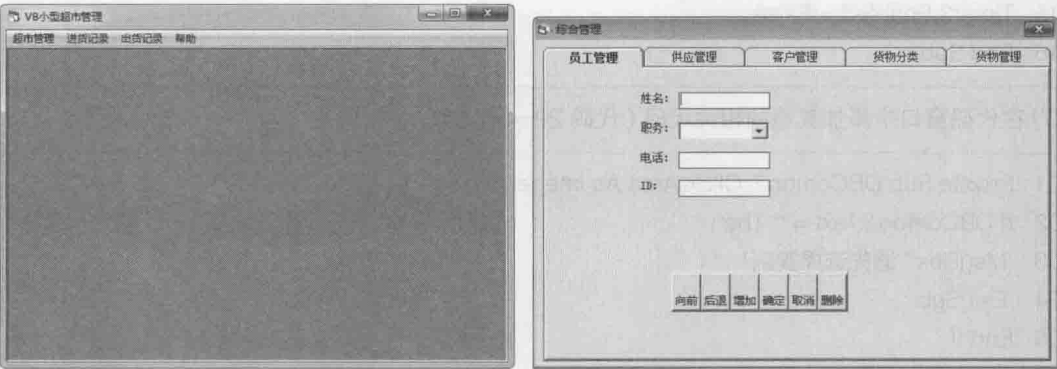
## 29.5 系统运行

 本节视频教学录像: 1 分钟

所有的工作都做好后, 就可以开始测试使用 VB 小型超市管理系统来管理你的超市了。

### 29.5.1 员工管理

- (1) 按【F5】快捷键运行程序。
- (2) 选择【超市管理】>【综合管理】菜单命令, 打开【综合管理】对话框。



- (3) 单击选择【员工管理】选项卡, 单击【增加】按钮, 在【姓名】、【电话】和【ID】文本框中输入相关信息, 在【职务】下拉列表中选择相应的选项, 然后单击【确定】按钮, 即可添加员工信息。
- (4) 完成添加员工的信息后, 单击【向前】和【后退】按钮, 即可浏览员工信息。

The image shows two screenshots of the 'Supply Management' (供应管理) window. The left window shows the 'Add' (增加) button highlighted. The right window shows the 'Forward' (向前) and 'Back' (后退) buttons highlighted.

## 29.5.2 供应管理

(1) 单击选择【供应管理】选项卡，单击【增加】按钮，在【名称】、【电话】、【地址】、【联系人】和【备注】等文本框中输入相关信息，在【级别】下拉列表中选择相应的选项，然后单击【确定】按钮，即可添加供应商信息。

(2) 完成添加供应商的信息后，单击【向前】和【后退】按钮，即可浏览供应商信息。

The image shows two screenshots of the 'Supply Management' (供应管理) window. The left window shows the 'Add' (增加) button highlighted. The right window shows the 'Forward' (向前) and 'Back' (后退) buttons highlighted.

## 29.5.3 客户管理

(1) 单击选择【客户管理】选项卡，单击【增加】按钮，在【名称】、【地址】、【联系人】、【电话】、【产品】、【数量】和【备注】等文本框中输入相关信息，然后单击【确定】按钮，即可添加客户信息。

(2) 完成添加客户的信息后，单击【向前】和【后退】按钮，即可浏览客户信息。

The image shows two screenshots of the 'Customer Management' (客户管理) window. The left window shows the 'Add' (增加) button highlighted. The right window shows the 'Forward' (向前) and 'Back' (后退) buttons highlighted.

## 29.5.4 货物分类管理

(1) 单击选择【货物分类】选项卡，单击【增加】按钮，在【编号】和【分类名称】文本框中输入相关信息，然后单击【确定】按钮，即可添加货物分类信息。

(2) 完成添加货物分类的信息后，单击【向前】和【后退】按钮，即可浏览货物分类信息。

## 29.5.5 货物管理

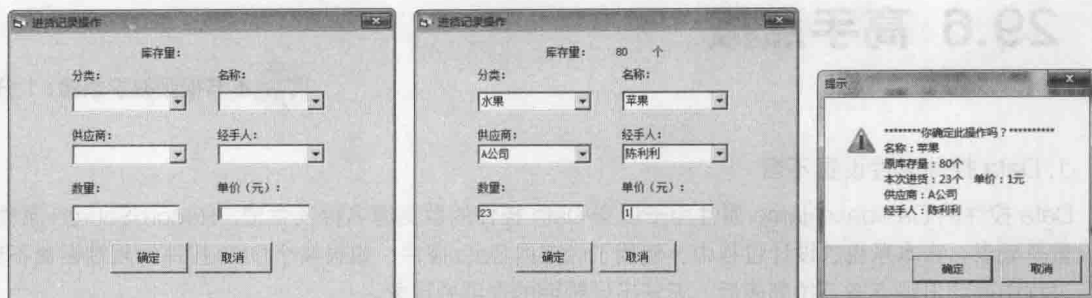
(1) 单击选择【货物管理】选项卡，单击【增加】按钮，在【名称】、【分类】、【数量】和【备注】等文本框中输入相关信息，在【单位】下拉列表中选择相应的选项，然后单击【确定】按钮，即可添加货物信息。

(2) 完成添加货物的信息后，单击【向前】和【后退】按钮，即可浏览货物信息。

## 29.5.6 进货记录管理

(1) 选择【进货记录】>【进货记录操作】菜单命令，打开【进货记录操作】对话框。

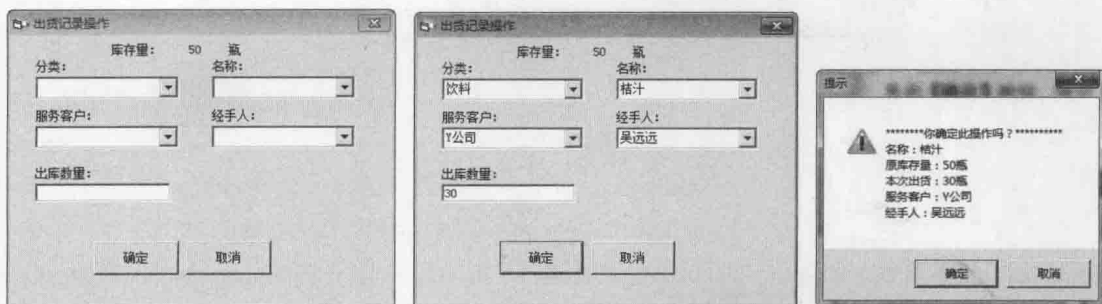
(2) 在【分类】、【名称】、【供应商】和【经手人】等下拉列表中选择相应的选项，在【数量】和【单价(元)】文本框中输入相应的数值，单击【确定】按钮，弹出【提示】对话框，然后单击【确定】按钮，即可完成进货记录操作。



## 29.5.7 出货记录管理

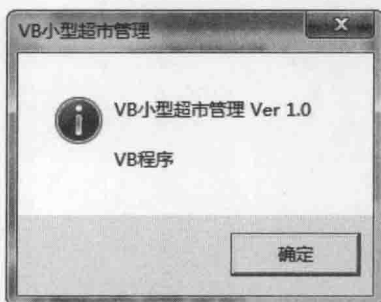
(1) 选择【出货记录】>【出货记录操作】菜单命令，打开【出货记录操作】对话框。

(2) 在【分类】、【名称】、【服务客户】和【经手人】等下拉列表中选择相应的选项，在【出库数量】文本框中输入相应的数值，单击【确定】按钮，弹出【提示】对话框，然后单击【确定】按钮，即可完成出货记录操作。



## 29.5.8 显示【关于】对话框

选择【帮助】>【关于】菜单命令，即打开关于软件信息的对话框。





## 29.6 高手点拨



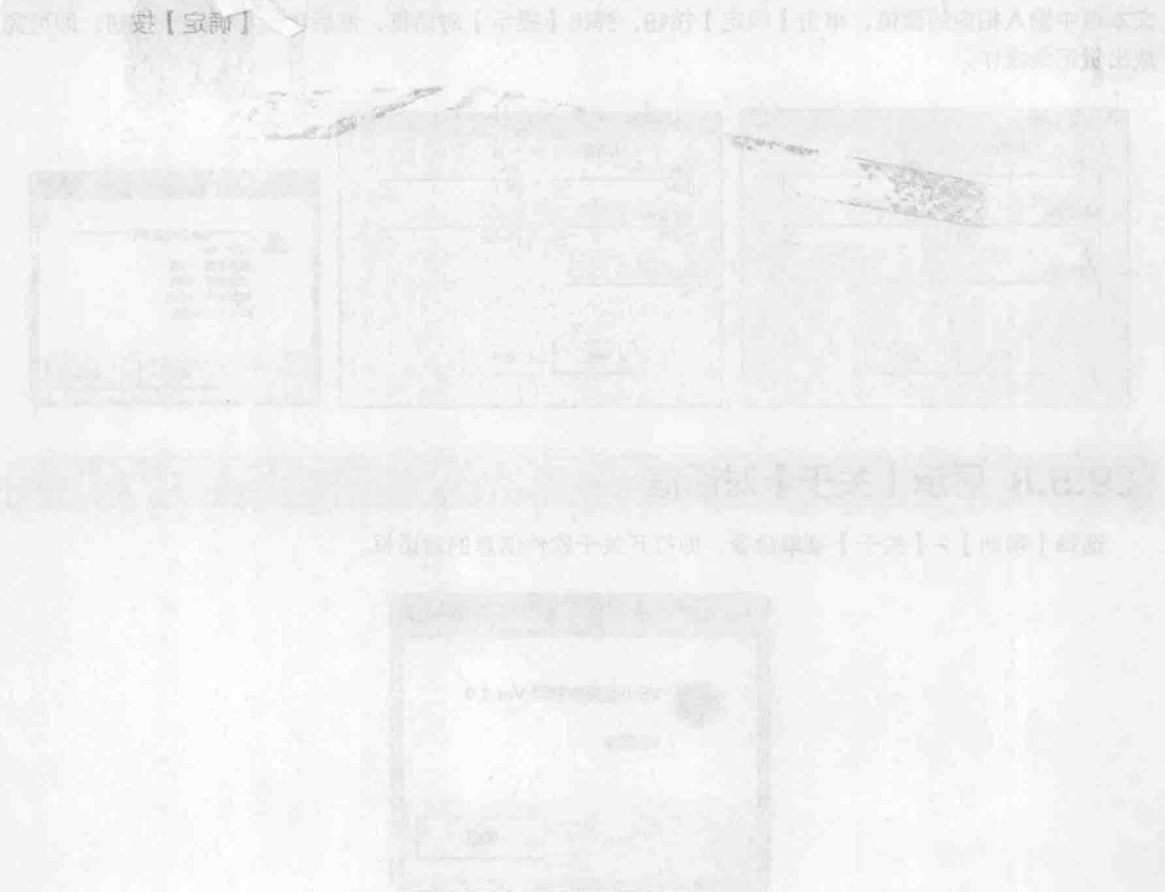
本节视频教学录像: 1 分钟

### 1. Data 控件属性设置不当

Data 控件的 DatabaseName 属性用于设置 Data 控件的数据源名称及位置, RecordSource 属性用于设置基础表。在本系统的设计过程中,使用了较多的 Data 控件,如果某个 Data 控件的属性设置不当,运行过程中就会出现点击下拉列表后,无法出现预期的选项的现象。

### 2. 文本框、组合框控件属性设置不当

与 Data 控件相似,在本系统中使用了较多的文本框和组合框用于显示或输入信息,供显示信息的文本框和组合框的 DataSource 属性用于设置其数据源, DataField 属性用于设置其显示的数据。如果设置错误,同样会出现上面所讲到的现象,因此在设置时一定要小心谨慎。



封面页

书名页

版权页

前言页

目录页

第0章 Visual Basic学习指南

0.1 Visual Basic的来源

0.2 Visual Basic的特点

0.3 Visual Basic无处不在

0.4 Visual Basic学习经验

0.5 Visual Basic的学习路线

第1篇 基础知识

第1章 步入VB开发之门——初识Visual Basic 6.0

1.1 Visual Basic简介

1.1.1 Visual Basic的发展

1.1.2 Visual Basic 6.0的功能特点

1.2 Visual Basic 6.0的安装与启动

1.2.1 Visual Basic 6.0的安装

1.2.2 Visual Basic 6.0开发环境的定制

1.2.3 启动与退出

1.3 Visual Basic 6.0的集成开发环境

1.3.1 认识Visual Basic 6.0的工作界面

1.3.2 主窗口

1.3.3 窗体设计/代码设计窗口

1.3.4 属性窗口

1.3.5 工程资源管理器窗口

1.3.6 工具箱窗口

1.3.7 其他窗口

1.3.8 Visual Basic帮助系统的使用

1.4 用Visual Basic 6.0管理工程

1.4.1 工程介绍

1.4.2 新建、保存工程

1.4.3 向工程中添加窗体和模块

1.4.4 运行和关闭工程

1.4.5 删除工程

1.4.6 生成可执行文件

1.5 来自VB世界的第一声问候——第1个应用程序

1.5.1 VB程序设计的一般步骤

1.5.2 创建应用程序的界面

1.5.3 设置控件属性

1.5.4 编写代码

1.5.5 调试、运行程序

## 1.6 实战练习

# 第2章 Visual Basic的入门钥匙——Visual Basic语言基础

## 2.1 标识符和数据类型

### 2.1.1 标识符

### 2.1.2 数据类型

## 2.2 常量和变量

### 2.2.1 常量

### 2.2.2 变量

## 2.3 运算符

### 2.3.1 算术运算符

### 2.3.2 赋值运算符

### 2.3.3 关系运算符

### 2.3.4 逻辑运算符

### 2.3.5 连接运算符

### 2.3.6 特殊运算符

### 2.3.7 运算符的优先级

## 2.4 表达式

### 2.4.1 算术表达式

### 2.4.2 字符串表达式

### 2.4.3 日期表达式

## 2.5 代码编写规范

### 2.5.1 Visual Basic 6.0标识符的定义规则

### 2.5.2 Visual Basic 6.0中变量及控件的命名规则

### 2.5.3 程序书写规则

### 2.5.4 添加注释

### 2.5.5 格式化缩排程序语句

## 2.6 高手点拨

## 2.7 实战练习

# 第3章 Visual Basic的秘密——算法和程序控制结构

## 3.1 算法

## 3.2 结构化程序设计

## 3.3 顺序结构

### 3.3.1 赋值运算符

### 3.3.2 数据的输入与输出

## 3.4 选择结构

### 3.4.1 If条件语句

### 3.4.2 Select case语句

### 3.4.3 条件函数

## 3.5 循环结构

### 3.5.1 For循环语句

### 3.5.2 Do...Loop循环语句

### 3.5.3 循环的嵌套

### 3.6 其他辅助控制语句

#### 3.6.1 End结束语句

#### 3.6.2 Exit退出语句

#### 3.6.3 GoTo跳转语句

#### 3.6.4 On Error语句

#### 3.6.5 复用语句With End With

### 3.7 高手点拨

### 3.8 实战练习

## 第4章 同类型批量数据管理的技巧——数组

### 4.1 数组的概念

#### 4.1.1 定长数组及声明

#### 4.1.2 动态数组及声明

### 4.2 数组基本操作

#### 4.2.1 数组的引用

#### 4.2.2 数组的初始化

#### 4.2.3 数组元素的输入、输出

#### 4.2.4 数组元素的插入、删除和查找

#### 4.2.5 数组元素的应用及排序

### 4.3 数组相关函数及语句

#### 4.3.1 Array函数

#### 4.3.2 UBound函数和LBound函数

#### 4.3.3 Split函数

#### 4.3.4 Option Base语句

### 4.4 高手点拨

### 4.5 实战练习

## 第5章 应用程序提升的法宝——内置函数与过程

### 5.1 秘密武器——常用的内置函数

#### 5.1.1 数学函数

#### 5.1.2 字符串函数

#### 5.1.3 转换函数

#### 5.1.4 日期时间函数

#### 5.1.5 随机函数

#### 5.1.6 判断函数

#### 5.1.7 格式化函数

#### 5.1.8 Shell函数

### 5.2 提升法宝——过程

#### 5.2.1 事件过程

#### 5.2.2 Sub过程（子过程）

#### 5.2.3 Function过程（函数过程）

#### 5.2.4 参数的传递

#### 5.2.5 过程的嵌套与递归

### 5.3 高手点拨

#### 5.4 实战练习

### 第2篇 核心技术

## 第6章 应用程序的精髓——可视化编程

### 6.1 对象概念

#### 6.1.1 对象和类

#### 6.1.2 VB中对象的建立和编辑

### 6.2 对象的属性、方法和事件

#### 6.2.1 对象的属性及设置

#### 6.2.2 对象的方法及调用

#### 6.2.3 对象的事件及事件过程

### 6.3 高手点拨

### 6.4 实战练习

## 第7章 应用程序的脸——窗体和系统对象

### 7.1 窗体简介

#### 7.1.1 窗体的基本概念

#### 7.1.2 在工程中添加窗体的方法

### 7.2 控制窗体表情——窗体的属性、方法和事件

#### 7.2.1 窗体的属性

#### 7.2.2 窗体的方法

#### 7.2.3 窗体的事件

### 7.3 窗体的生命周期

#### 7.3.1 选择启动窗体

#### 7.3.2 快速显示窗体

#### 7.3.3 结束窗体

### 7.4 多窗体设计

#### 7.4.1 创建多窗体应用程序

#### 7.4.2 多窗体特性

### 7.5 登录窗体设计实例

### 7.6 系统对象

#### 7.6.1 应用程序APP对象

#### 7.6.2 屏幕Screen对象

#### 7.6.3 剪贴板Clipboard对象

#### 7.6.4 调试Debug对象

### 7.7 高手点拨

### 7.8 实战练习

## 第8章 标准模块和类模块

### 8.1 标准模块

#### 8.1.1 标准模块概述

#### 8.1.2 添加标准模块

### 8.2 类模块

#### 8.2.1 类模块概述

#### 8.2.2 添加类模块

8.3 标准模块和类模块的区别

8.4 高手点拨

8.5 实战练习

## 第9章 VB的简易之道——标准控件

9.1 控件概述

9.2 标签控件

9.2.1 标签控件的主要属性

9.2.2 标签控件 (Label) 的主要事件

9.2.3 标签控件应用示例

9.3 文本框控件

9.3.1 文本框的主要属性

9.3.2 文本框控件常用的事件

9.3.3 文本框控件应用示例

9.4 命令按钮控件

9.4.1 命令按钮控件的主要属性

9.4.2 命令按钮控件的事件

9.4.3 命令按钮控件应用示例

9.5 单选按钮控件

9.5.1 单选按钮的主要属性

9.5.2 单选按钮的常用事件

9.5.3 单选按钮控件应用示例

9.6 复选框控件

9.6.1 复选框的主要属性

9.6.2 复选框的常用事件

9.6.3 复选框控件应用示例

9.7 框架控件

9.7.1 框架的主要属性

9.7.2 框架控件应用示例

9.8 列表框控件

9.8.1 列表框的主要属性

9.8.2 列表框的主要事件

9.8.3 列表框控件的方法

9.8.4 列表框控件应用示例

9.9 组合框控件

9.9.1 组合框控件的主要属性

9.9.2 组合框的事件和方法

9.9.3 组合框应用示例

9.10 图像框控件

9.10.1 图像框控件的主要属性

9.10.2 图像框控件的主要事件和方法

9.10.3 图像框应用示例

9.11 滚动条控件

- 9.11.1 滚动条控件的主要属性
- 9.11.2 滚动条控件的主要事件
- 9.11.3 滚动条应用示例
- 9.12 程序中的闹钟——定时器控件
  - 9.12.1 定时器控件的主要属性
  - 9.12.2 定时器控件的主要事件
  - 9.12.3 定时器控件应用示例
- 9.13 文件系统控件
  - 9.13.1 驱动器列表框控件
  - 9.13.2 目录列表框控件
  - 9.13.3 文件列表框控件
  - 9.13.4 文件系统应用示例
- 9.14 控件数组
  - 9.14.1 控件数组的概念
  - 9.14.2 控件数组的创建
  - 9.14.3 控件数组的使用
- 9.15 高手点拨
- 9.16 实战练习
- 第10章 扩展你的需求——ActiveX控件、工具栏和状态栏
  - 10.1 ActiveX控件的使用
    - 10.1.1 ActiveX控件的添加
    - 10.1.2 ActiveX控件的删除
    - 10.1.3 ActiveX控件的注册
  - 10.2 图像列表控件
    - 10.2.1 向图像列表控件添加图片
    - 10.2.2 图像列表控件与其他控件关联
    - 10.2.3 图像列表控件的应用实例
  - 10.3 树状视图控件——统筹全局的好工具
    - 10.3.1 树状视图控件的主要属性、事件和方法
    - 10.3.2 树状视图控件的应用实例
  - 10.4 选项卡控件
    - 10.4.1 选项卡控件的主要属性
    - 10.4.2 选项卡控件的应用实例
  - 10.5 进度条控件
    - 10.5.1 进度条控件的主要属性和方法
    - 10.5.2 进度条控件的应用实例
  - 10.6 视图控件 (ListView)
    - 10.6.1 ListView控件简介
    - 10.6.2 添加数据
    - 10.6.3 创建报表视图
    - 10.6.4 创建大图标视图
  - 10.7 日期/时间控件 (DateTimePicker)



- 10.7.1 认识DateTimePicker控件
- 10.7.2 设置和返回日期
- 10.7.3 实时读取DTPicker控件中的日期
- 10.7.4 使用CheckBox属性来选择无日期
- 10.7.5 使用日期和时间的格式
- 10.7.6 使用DTPicker控件计算日期或天数
- 10.8 工具栏控件
  - 10.8.1 工具栏控件的主要属性和事件
  - 10.8.2 工具栏控件的应用实例
- 10.9 状态栏控件
  - 10.9.1 状态栏控件的属性
  - 10.9.2 状态栏控件的方法
  - 10.9.3 状态栏控件的事件
- 10.10 高手点拨
- 10.11 实战练习

## 第11章 鼠标、键盘的另类编程应用——鼠标、键盘事件

- 11.1 鼠标事件
  - 11.1.1 “鼠标按键按下”事件(MouseDown)
  - 11.1.2 “鼠标按键释放”事件(MouseUp)
  - 11.1.3 “移动鼠标”事件(MouseMove)
- 11.2 键盘事件
  - 11.2.1 “键盘按键”事件(KeyPress)
  - 11.2.2 “键盘按下”事件(KeyDown)
  - 11.2.3 “键盘弹起”事件(KeyUp)
- 11.3 高手点拨
- 11.4 实战练习

## 第12章 程序与用户的交互——菜单和对话框设计

- 12.1 魅力化妆师——菜单设计
  - 12.1.1 菜单编辑器
  - 12.1.2 下拉式菜单设计
  - 12.1.3 弹出式菜单设计
  - 12.1.4 自定义菜单设计
- 12.2 模式对话框和无模式对话框
- 12.3 预定义对话框设计
  - 12.3.1 输入对话框设计
  - 12.3.2 消息对话框设计
- 12.4 通用对话框设计
  - 12.4.1 添加通用对话框控件
  - 12.4.2 通用对话框设计实例
- 12.5 高手点拨
- 12.6 实战练习

## 第13章 编程错误终结者——程序调试与错误处理

### 13.1 Visual Basic 6.0程序中的错误类型

#### 13.1.1 语法错误

#### 13.1.2 逻辑错误

#### 13.1.3 运行时错误

### 13.2 程序工作状态

#### 13.2.1 设计状态

#### 13.2.2 运行状态

#### 13.2.3 中断状态

### 13.3 程序调试

#### 13.3.1 使程序进入中断状态

#### 13.3.2 调试工具

#### 13.3.3 调试方法

### 13.4 除虫行动——Visual Basic 6.0中的错误处理

#### 13.4.1 Err对象

#### 13.4.2 On Error GoTo语句

#### 13.4.3 Resume语句

#### 13.4.4 错误处理实例

### 13.5 高手点拨

### 13.6 实战练习

## 第3篇 高级应用

### 第14章 进入数据库——数据库与SQL语言基础

#### 14.1 数据库基本概念

#### 14.2 SQL应用

##### 14.2.1 SQL语言的特点

##### 14.2.2 常用SQL语句简介

### 14.3 Select语句的使用——数据库的灵魂

#### 14.3.1 Select子语句

#### 14.3.2 From子语句

#### 14.3.3 As子语句

#### 14.3.4 Where子语句

#### 14.3.5 Order By子语句

#### 14.3.6 Group By子语句

### 14.4 SQL中的常用函数

#### 14.4.1 算术函数

#### 14.4.2 统计函数

### 14.5 利用SQL语言修改表数据

#### 14.5.1 Insert语句

#### 14.5.2 Update语句

#### 14.5.3 Delete语句

### 14.6 高手点拨

### 14.7 实战练习

### 第15章 Visual Basic与数据库的联合——Visual Basic 6.0中的数据库编程

- 15.1 英雄相惜——Visual Basic 6.0与数据库
    - 15.1.1 Visual Basic支持的常用数据库
    - 15.1.2 Visual Basic中的数据库控件
  - 15.2 数据库的建立、维护和查询
    - 15.2.1 建立数据库
    - 15.2.2 删除数据库中的表
    - 15.2.3 修改数据表结构和数据
    - 15.2.4 数据查询
    - 15.2.5 数据窗体设计器
  - 15.3 使用Data控件访问数据库
    - 15.3.1 Data控件的常用属性
    - 15.3.2 Data控件的常用方法
    - 15.3.3 Data控件的常用事件
    - 15.3.4 Data控件访问数据库实例
  - 15.4 使用ADO控件访问数据库
    - 15.4.1 添加ADO控件
    - 15.4.2 ADO控件的常用属性
    - 15.4.3 ADO控件的常用方法
    - 15.4.4 ADO控件的常用事件
    - 15.4.5 ADO控件访问数据库实例
  - 15.5 高手点拨
  - 15.6 实战练习
- 第16章 Visual Basic 6.0生成的报表——数据报表
- 16.1 数据报表简介
  - 16.2 数据报表的生成环境
  - 16.3 数据报表的生成
  - 16.4 高手点拨
  - 16.5 实战练习
- 第17章 Visual Basic编程的核心——API编程
- 17.1 API概述
    - 17.1.1 API基本数据类型
    - 17.1.2 API常见数据结构
    - 17.1.3 API浏览器
  - 17.2 API的函数分类
    - 17.2.1 窗口管理类函数
    - 17.2.2 图形设备接口类函数
    - 17.2.3 系统服务类函数
    - 17.2.4 国际特性类函数
    - 17.2.5 网络服务函数
  - 17.3 API函数的应用
    - 17.3.1 使用Declare语句手动声明API函数
    - 17.3.2 使用API浏览器声明API函数

17.3.3 API函数的调用

17.4 插上翅膀去飞翔——API编程实例

17.5 高手点拨

17.6 实战练习

## 第18章 Visual Basic中的网络世界——网络编程

18.1 邮件应用编程

18.1.1 邮件程序接口控件的属性和方法

18.1.2 实现邮件发送

18.2 互联网传输应用编程

18.2.1 互联网传输控件的属性、事件和方法

18.2.2 实现互联网文件上传

18.3 网页浏览器应用编程

18.3.1 网页浏览器控件的属性、事件和方法

18.3.2 实现自定义网页浏览器应用

18.4 高手点拨

18.5 实战练习

## 第19章 Visual Basic中的视听——图形图像与多媒体编程

19.1 图形应用编程

19.1.1 坐标系

19.1.2 颜色设置

19.1.3 绘图方法

19.2 多媒体应用编程

19.2.1 多媒体控制接口控件基本概念

19.2.2 多媒体控制接口控件的属性

19.2.3 多媒体控制接口控件的事件

19.2.4 多媒体控制接口控件应用实例

19.3 让程序动起来——动画应用编程

19.3.1 添加动画控件

19.3.2 动画控件的属性

19.3.3 动画控件的方法

19.3.4 动画控件应用实例

19.4 高手点拨

19.5 实战练习

## 第20章 用VB操纵文件——文件系统编程

20.1 文件的类型与结构

20.1.1 文件结构

20.1.2 文件类型

20.2 文件操作语句

20.3 操纵文件的魔法——文件操作函数

20.4 顺序文件

20.4.1 顺序文件的打开

20.4.2 顺序文件的读取

- 20.4.3 顺序文件的写入
- 20.4.4 顺序文件的关闭
- 20.4.5 顺序文件使用实例
- 20.5 随机文件
- 20.5.1 随机文件的打开
- 20.5.2 随机文件的读取
- 20.5.3 随机文件的写入
- 20.5.4 随机文件的关闭
- 20.5.5 随机文件使用实例
- 20.6 二进制文件
- 20.6.1 二进制文件的打开
- 20.6.2 二进制文件的读取
- 20.6.3 二进制文件的写入
- 20.6.4 二进制文件的关闭
- 20.6.5 二进制文件使用实例
- 20.7 高手点拨
- 20.8 实战练习

## 第21章 让你的程序去旅行——应用程序打包

- 21.1 打包前的准备
- 21.2 打包应用程序
- 21.3 安装应用程序
- 21.4 卸载应用程序
- 21.5 打包应注意的问题
- 21.6 高手点拨
- 21.7 实战练习

## 第4篇 应用开发

## 第22章 项目实战前的忠告——项目规划

- 22.1 项目开发流程
- 22.1.1 项目策划阶段
- 22.1.2 需求分析阶段
- 22.1.3 项目开发阶段
- 22.1.4 项目测试阶段
- 22.1.5 项目后期维护
- 22.2 满足客户需求
- 22.3 项目开发团队
- 22.3.1 项目团队组成
- 22.3.2 项目团队特征
- 22.4 项目计划说明书
- 22.5 项目开发阶段的运作
- 22.5.1 初始阶段
- 22.5.2 细化阶段
- 22.5.3 构建阶段

- 22.5.4 交付阶段
- 22.5.5 维护阶段
- 22.6 高手点拨
- 第23章 网络通信应用开发——VB实现远程控制
  - 23.1 系统分析
  - 23.2 系统设计
  - 23.3 运行系统
  - 23.4 开发过程常见问题及解决方法
  - 23.5 高手点拨
- 第24章 图形图像应用开发——仿Windows画图程序
  - 24.1 系统分析
  - 24.2 系统设计
  - 24.3 运行系统
  - 24.4 高手点拨
- 第25章 多媒体应用开发——开发自己的播放器
  - 25.1 系统分析
  - 25.2 系统设计
  - 25.3 运行系统
  - 25.4 开发过程常见问题及解决方法
  - 25.5 高手点拨
- 第26章 文件系统应用开发——文件分割与合并程序
  - 26.1 系统分析
  - 26.2 系统设计
  - 26.3 运行系统
  - 26.4 开发过程常见问题及解决方法
  - 26.5 高手点拨
- 第27章 游戏开发——VB连连看
  - 27.1 系统分析
  - 27.2 系统设计与开发
  - 27.3 运行系统
  - 27.4 高手点拨
- 第5篇 项目实战
- 第28章 数据库应用开发——个人账目管理系统
  - 28.1 系统分析
    - 28.1.1 系统需求分析
    - 28.1.2 系统功能模块设计
  - 28.2 数据库分析和设计
    - 28.2.1 数据库分析
    - 28.2.2 创建数据库
    - 28.2.3 创建表
  - 28.3 系统界面设计
    - 28.3.1 创建工程和数据库连接模块

- 28.3.2 添加控件
  - 28.3.3 系统主界面设计
  - 28.3.4 系统功能实现的各界面设计
  - 28.4 系统代码设计
    - 28.4.1 主窗体代码设计
    - 28.4.2 【日常收入】窗体代码设计
    - 28.4.3 【日常支出】窗体代码设计
    - 28.4.4 【借入款项】窗体代码设计
    - 28.4.5 【借出款项】窗体代码设计
    - 28.4.6 【月度统计】窗体代码设计
  - 28.5 运行系统
    - 28.5.1 系统主界面操作
    - 28.5.2 项目管理操作
    - 28.5.3 日常收入、支出管理操作
    - 28.5.4 借入款项、借出款项管理操作
    - 28.5.5 月度统计管理操作
  - 28.6 高手点拨
- 第29章 打造你的小型超市——超市进销存管理系统
- 29.1 需求及功能分析
  - 29.2 数据库设计
    - 29.2.1 创建数据库
    - 29.2.2 创建表
  - 29.3 系统界面设计
    - 29.3.1 【综合管理】窗体设计
    - 29.3.2 【员工管理】选项卡设计
    - 29.3.3 【供应管理】选项卡设计
    - 29.3.4 【客户管理】选项卡设计
    - 29.3.5 【货物分类】选项卡设计
    - 29.3.6 【货物管理】选项卡设计
    - 29.3.7 【进货记录操作】窗体设计
    - 29.3.8 【出货记录操作】窗体设计
    - 29.3.9 【VB小型超市管理】主窗体设计
  - 29.4 系统代码编写
    - 29.4.1 添加【VB小型超市管理】窗体代码
    - 29.4.2 添加【综合管理】窗体代码
    - 29.4.3 添加【进货记录操作】窗体代码
    - 29.4.4 添加【出货记录操作】窗体代码
  - 29.5 系统运行
    - 29.5.1 员工管理
    - 29.5.2 供应管理
    - 29.5.3 客户管理
    - 29.5.4 货物分类管理



- 29.5.5 货物管理
- 29.5.6 进货记录管理
- 29.5.7 出货记录管理
- 29.5.8 显示【关于】对话框
- 29.6 高手点拨

插页页

附录页

封底页