求是科技

# Delphi

# 数据库通用模块及典型系统开发

求是科技
周新会　周金根　编著

实例导航

附源代码光盘
CD-ROM

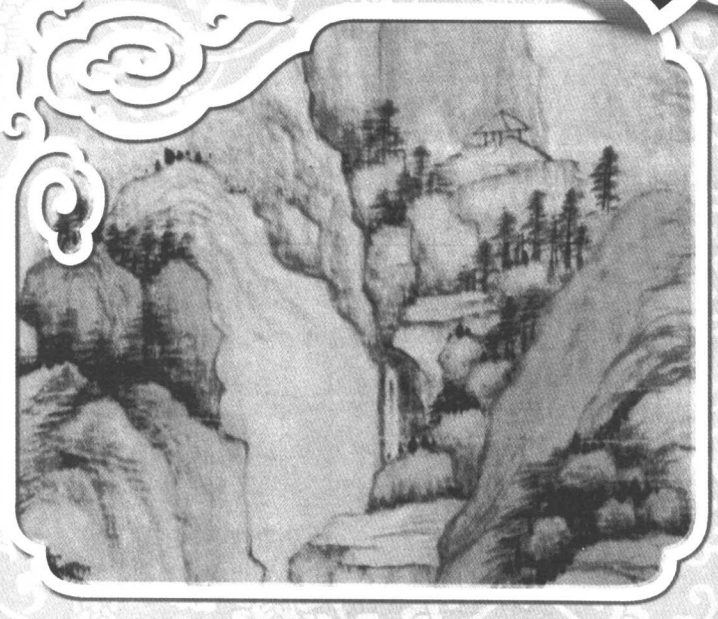
人民邮电出版社
POSTS & TELECOM PRESS

# Delphi

# 数据库通用模块及典型系统开发

● 求是科技
周新会　周金根　编著

实例导航



人民邮电出版社
POSTS & TELECOM PRESS

**图书在版编目（CIP）数据**

Delphi 数据库通用模块与典型系统开发实例导航 / 求是科技编著.
—北京：人民邮电出版社，2006.2
ISBN 7-115-14331-5

Ⅰ. D... Ⅱ. 求... Ⅲ. 软件工具—程序设计 Ⅳ. TP311.56

中国版本图书馆 CIP 数据核字（2006）第 006970 号

## 内 容 提 要

本书详细介绍了多个 Delphi 通用模块和典型系统，用实例的方式分别讲述了系统登录、用户及权限管理、数据库连接、数据访问和数据显示及打印等通用模块的具体实现，同时还对人事管理、考勤管理、工资管理、固定资产管理、物资管理和销售管理等典型应用系统的系统分析、数据库设计以及编码实现的全过程做了详尽的描述。

本书中介绍的通用模块代表性强，典型系统案例内容丰富、设计专业、功能详实。

本书适合大中专院校的学生、软件项目开发人员以及 Delphi 开发爱好者学习和参考。

**Delphi 数据库通用模块及典型**
**系统开发实例导航**

◆ 编　　著　求是科技　周新会　周金根
责任编辑　张立科

◆ 人民邮电出版社出版发行　　北京市崇文区夕照寺街 14 号
邮编　100061　　电子函件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京鸿佳印刷厂印刷
新华书店总店北京发行所经销

◆ 开本：787×1092　1/16
印张：24
字数：585 千字　　2006 年 2 月第 1 版
印数：1 – 6 000 册　　2006 年 2 月北京第 1 次印刷

ISBN 7-115-14331-5/TP・5189

定价：42.00 元（附光盘）

**读者服务热线：(010)67132692　印装质量热线：(010)67129223**

# 前　言

近年来，由 Borland 公司推出的 Delphi 开发工具，以其功能强大、快速高效的特点，深受广大应用软件开发人员的喜爱。因此，在许多应用软件开发，尤其是数据库应用系统的开发中广泛使用。

本书的目的旨在帮助使用 Delphi 进行应用系统开发的读者，使他们能够通过实例的方式快速掌握利用 Delphi 进行应用程序开发的技术，了解应用系统开发的分析、设计、实现等环节，并能够在自己的学习和开发实践中加以运用。

本书采用了大量的实例，内容丰富，介绍详尽。所讲述的各通用模块代表性强，读者通过各章节的介绍，不仅能够了解这些通用模块的原理和方法，并且可以轻松掌握其具体实现。书中所介绍的典型系统完整、贴近实际，并且几个典型系统实例采用了各不相同的实现方法，读者可以通过它们充分了解和掌握利用 Delphi 构建应用系统的技术手段和开发过程。

本书包括 Delphi 通用模块和典型系统两部分内容，具体安排如下。

第一部分为第 1 章～第 5 章。介绍了 Delphi 通用模块的实现方法，这些模块包括系统登录通用模块、用户及管理权限管理通用模块、数据库连接通用模块、数据访问通用模块和数据显示及打印通用模块。

第二部分为第 6 章～第 11 章。对几个典型的 Delphi 系统从系统分析、数据库设计和编码的实现几方面进行了介绍，这几个典型的系统包括人事管理系统、考勤管理系统、工资管理系统固定资产管理系统、物资管理系统和销售管理系统等。

此外，本书所附光盘中包括所有章节的代码实例，以便于读者学习和参考。

真诚地希望本书能给广大读者在 Delphi 开发方面带来帮助，并恳请您对本书提出宝贵的建议和意见。

编者

2006 年 2 月

# 目　录

# 第 1 章　系统登录通用模块

大多数应用软件都包含登录模块，以便让合法的用户能够使用软件系统。在这一章里将会介绍通用的系统登录模块。

一般地，系统会以某种方式存储用户的信息，包括账号、密码等，在登录软件时查找资料中是否有该登录用户的记录，还要验证密码是否正确，如果用户信息匹配则成功登录系统，并可以使用软件提供的各项功能，否则不能进入该软件系统。

本章将介绍两种不同的用户信息存储方式，一种是利用文件存储用户信息，另一种是利用数据库存储用户信息。另外，本章还将介绍如何对用户密码进行特殊处理，以免被他人非法获取。

## 1.1　模块说明

本模块在打开程序主窗体之前弹出一个对话框，需要用户输入账号和密码，如果系统存在此用户并且密码正确，则打开软件的主窗体，可以使用软件的所有功能；如果用户不存在或者密码不正确则提示用户不能登录系统。

在介绍利用文件存储用户信息时将针对如下 3 种类型文件分别介绍其实现方法：

- 文本文件存储；
- 记录文件存储；
- 无类型文件存储。

## 1.2　模块设计

首先建立一个用户登录窗体，在主窗体打开之前弹出该窗体。登录窗体中包括账号和密码的输入框和“确定”、“取消”按钮，在用户输入完成并单击“确认”按钮后，系统将在存储的用户信息中查找与输入的账号和密码匹配的信息，如果找到匹配信息则登录成功打开主窗体，否则给出“账号或密码错误，请重新输入！”的提示信息。

对于用户信息的存储方式，可以根据软件的具体应用情况来选择。如果系统本身是数据库应用程序，则可以考虑采用数据库存储方式；如果是一个小程序，或者不涉及数据库应用，则采用文件存储方式最简单也最方便。

文件存储有 3 种方式，一种是文本文件存储方式，文本文件允许用户直接用类似记事本程序这样的工具打开编写，采用每行一条用户信息的方式；第二种是记录文件存储方式，记录文件可以由开发人员自己定义记录格式，因此一般需要一个用户管理工具进行文件的编辑，每条记录的长度固定；第三种是无类型文件存储方式，无类型文件能够存储可变长度记录，可以利用 Delphi 提供的相应函数进行读写等操作，用这种文件类型存储用户信息的应用并不多见，因此在本章中仅做简单介绍。

另外，还需要考虑在软件系统启动后的什么时机弹出登录对话框。一般有两种方式，一种是在主窗体 Form 的 OnCreate 事件中生成登录对话框的实例；另一种则是在项目文件的主程序生

成主窗体 Form 之前调用登录对话框，如果登录成功再生成主窗体。从一般程序执行的流程考虑，本章介绍的登录模块采用了第二种方式。

在这个模块中，最核心的就是检查信息的匹配。在实现该功能的函数中，首先要打开存储用户信息的文件或数据库，然后与用户输入的账号和密码进行对比。本章下面各节将分别介绍采用不同存储方式时的用户信息匹配检查。

系统登录模块中另外一个重要的设计就是密码的加/解密处理，为了避免他人非法获取用户登录密码，需要对密码明文进行加密处理。也就是说文件或数据库中存储的是经过加密处理的密码，用户登录时输入的密码，需要经过转换后才能验证其是否正确，这样可以更好地保护用户登录信息的安全。本章 1.8 节中将对用户登录密码的加/解密处理作详细介绍。

## 1.3 创建登录窗体

### 1.3.1 窗体界面设计

登录窗体的设计非常简单，在项目中添加一个窗体，命名为 LoginDlg。窗体界面如图 1-1 所示。

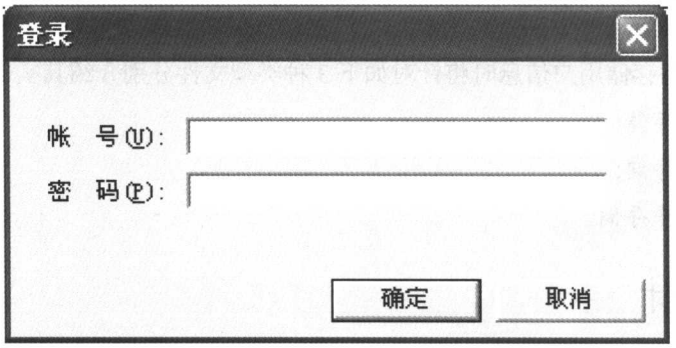



图 1-1 登录窗体界面

然后在窗体中添加各组件并设置相应的属性，登录窗体（LoginDlg）中的组件及其属性如表 1-1 所示。

表 1-1 LoginDlg 窗体主要组件属性设置和说明

| 组件 | 名称 | 属性设置 | 说明 |
|---|---|---|---|
| TForm | LoginDlg | Caption：登录<br>BorderStyle：bsDialog<br>Position：poScreenCenter | 显示登录窗体 |
| TEdit | edtAccount | Text：空 | 输入账号 |
| TEdit | edtPassword | Text：空<br>PasswordChar：* | 输入密码 |

续表

| 组件 | 名称 | 属性设置 | 说明 |
| --- | --- | --- | --- |
| TLabel | lblAccount | Caption：账　号(&U):<br>FocusControl：edtAccount | |
| TLabel | lblPassword | Caption：密　码(&P):<br>FocusControl：edtPassword | |
| TButton | btnOK | Default：True | 确定输入用户信息，开始进行验证 |
| TButton | btnCancel | ModalResult：mrCancel | 取消登录 |

## 1.3.2　窗体代码实现

### 1.　声明窗体级函数

在登录窗体中，需要提供两个方法，一个是提供从外部调用登录窗体的 Execute 类方法，另一个是验证用户是否合法的 VerifyAccount 方法，具体的声明如下：

```
public
// 一个类方法，以便在主程序中调用此方法来执行登录，如果登录成功返回 True，否则返回 False
class function Execute: Boolean;
// 验证账号和密码是否匹配，具体实现因不同的用户信息存储方式而不同
function VerifyAccount: Boolean;
```

### 2.　实现 Execute 方法

Execute 方法通过判断模态窗体 ModalResult 的值来决定返回值，ModalResult 的赋值将在 VerifyAccount 中进行，如果用户信息匹配则给 ModalResult 赋值为 True，否则为 False。当给 ModalResult 赋值为非 mrNone 时，会关闭登录窗体。具体的代码如下：

```
class function TLoginDlg.Execute: Boolean;
begin
  with TLoginDlg.Create(nil) do
  try
    Result := ShowModal = mrOk;
  finally
    Free;
  end;
end;
```

### 3.　实现 VerifyAccount 方法

此方法用于实现验证用户信息的功能，由于每种存储方式都不一样，读者可以在接下来的各节中分别看到具体的实现。

### 4.　btnOK 按钮的 OnClick 事件

在 btnOK 按钮的 OnClick 事件中添加检查账号和密码的代码如下：

```
procedure TLoginDlg.btnOKClick(Sender: TObject);
begin
  if not VerifyAccount then
    ShowMessage('账号或密码错误，请重新输入!');
end;
```

### 1.3.3 在项目文件中调用登录窗体

为了在主窗体生成之前调用登录窗体，需要修改一下由 Delphi 自动生成的项目文件源码。如果 TLoginDlg.Execute 返回为 True，则生成主窗体并调用 Application.Run，否则调用 Application.Terminate 中断并退出程序。具体的代码如下：

```
program pLogin1;

uses
  Forms,
  uMain in 'uMain.pas' {MainForm},
  uLogin in 'uLogin.pas' {LoginDlg};

{$R *.res}

begin
  Application.Initialize;
  if TLoginDlg.Execute then                          // 调用登录窗体
  begin
    Application.CreateForm(TMainForm, MainForm);   // 创建主窗体
    Application.Run;
  end
  else
    Application.Terminate;
end.
```

以上介绍了登录窗体中一些公共部分的内容，下面将逐一介绍采用不同存储方式时的处理。

## 1.4 文本文件存储用户信息

利用文本文件存储用户信息时，需要考虑到文本文件的特性。文本文件是以行为单位进行读、写操作的，由于每一行长度不一定相同，不能计算出给定行在文件中的确切位置，因而只能顺序地读写。另外，文本文件只能单独为读或写而打开，在一个打开的文本文件上同时进行读、写操作是不允许的。

下面首先来了解一下程序中与文本文件相关的操作，然后再介绍利用文本文件存储用户信息时，实现系统登录的方法。

### 1.4.1 文本文件的打开和关闭

#### 1. 文本文件的变量声明

首先来看一下文本文件类型的变量声明，代码如下：

```
Var
  TextFileVar: TextFile;
```

### 2. 打开文本文件

文本文件的打开需要如下两个步骤。

（1）文件变量与文件名关联

关联文件变量与文件名调用 AssignFile 标准过程，代码如下：

```
AssignFile ( TextFileVar , FileName ) ;
```

其中 FileName 既可以是全路径名，也可以仅是文件名，如果文件名没有包括路径，系统将会在当前目录下查找该文件。

（2）初始化读写

初始化读写有如下 3 种方式。

- Reset 为读打开文本文件并把文件指针移动到文件首，代码如下：

```
Reset ( TextFileVar ) ;
```

- Rewrite 为写创建一个新文本文件，代码如下：

```
Rewrite ( TextFileVar ) ;
```

- Append 为写打开存在的文本文件并把文件指针定位在文件尾，代码如下：

```
Append ( TextFileVar ) ;
```

当使用 Reset 或 Append 过程而文件不存在时，将会引发一个 I/O 异常。

### 3. 关闭文本文件

文本文件的关闭很简单，只需调用 CloseFile 过程即可。代码如下：

```
CloseFile ( TextFileVar ) ;
```

虽然 Delphi 应用程序在退出时会自动关闭所有打开的文件，但自己动手关闭文件可以确保释放文件句柄，并使程序的可移植性增强。

## 1.4.2　文本文件的读写

### 1. 从文本文件读取数据

从文本文件中读取信息要用到 Read 或 Readln 标准过程，Read 过程从文本文件中读取指定的数据，Readln 过程则从文本文件中读取一行数据。

利用 Read 过程读取数据，当读入数据为数值时，它默认数值是用一个或多个空格分开的，而不是逗号、分号或其他字符。如下面的语句：

```
Read ( TextFileVar , Num1, Num2, Num3 ) ;
```

如果文本文件中的数据是“100 200 300”，则能够成功读入数据；如果文本文件中的数据是“100 200, 300”，则 Read 过程读入“200,”并在试图把它转换为一个数值时将会引发一个异常。

利用 Read 或 Readln 过程读取字符串数据时，它总是读取尽可能多的字符填充到字符串变量或一直读到行结束符为止。因此从文本文件中读取格式化的字符串数据，必须声明与其长度相匹配的字符串变量。如果要从文本文件中读取单词，需要先把文件中的每一行读入字符串变量，然后再从该字符串中逐个分析出单词，或者一次只从文本文件中读入一个字符并测试每个字符后是

否是单词的断开处。

2. 向文本文件写入数据

向文本文件写入数据要用到 Write 或 Writeln 标准过程，Write 过程向文本文件写入指定的数据，Writeln 过程则向文本文件中写入一行数据。由于本章主要利用文本文件取得用户信息实现系统登录，因此对文本文件的写操作不做过多介绍。

## 1.4.3 文本文件的编辑

在 Delphi 中实现对一个文本文件的编辑，只要将其与一个 TMemo 控件建立关联即可，然后利用 Memo 控件的 LoadFromFile 方法将文本文件载入。代码如下：

```
Memo.Lines.LoadFromFile ( TextFileName ) ;
```

用户在 Memo 控件中对文本文件内容进行修改后，只要再调用 Memo 控件的 SaveToFile 方法后就会将最终结果保存到文本文件中去。代码如下：

```
Memo.Lines.SaveToFile ( TextFileName ) ;
```

本例中不单独给出对文本文件编辑的实现，而直接采用记事本这样的文本编辑工具对存储用户信息的文本文件进行修改。有兴趣的读者可以参照上面介绍的方法，实现对文本文件的编辑功能。

## 1.4.4 代码实现

利用文本文件存储用户信息实现系统登录功能时，需要创建一个名为 user.txt 的文本文件，该文件中每行存储一个用户信息，代码如下：

```
Admin=password1
Guest=password2
……
```

其中，“=”号左边是账号，右边是该账号对应的密码。下面介绍使用文本文件存储用户信息时对账号和密码进行合法性验证的方法。

在验证用户信息的 VerifyAccount 方法中，首先要声明一个 TextFile 类型的变量 TextFileVar，然后用 AssignFile 过程给文件变量赋值，关联当前目录下的用户信息文件“user.txt”，接着调用 Reset 初始化文件以便能够开始读取，然后利用 Readln 过程逐行读取文件，并通过字符串操作获取账号和密码，与登录时用户输入的账号和密码进行比较，直至找到匹配的用户信息或者文本文件读完为止，最后还需调用 CloseFile 过程及时释放文件句柄。

VerifyAccount 方法的代码如下：

```
function TLoginDlg.VerifyAccount: Boolean;
var
  TextFileVar: TextFile;
  sUserName, sPassword, sLine: string;
begin
  Result := False;
  AssignFile(TextFileVar, 'user.txt');
  Reset(TextFileVar);
```

```
  while not Eof(TextFileVar) do
  begin
    ReadLn(TextFileVar, sLine);
    sUserName := Copy(sLine, 0, Pos('=', sLine)-1);
    sPassword := Copy(sLine, Pos('=', sLine)+1, MaxInt);
    if SameText(sUserName, edtAccount.Text) and (sPassword = edtPassword.Text) then
    begin
      ModalResult := mrOk;
      Break;
    end;
  end;
  CloseFile(TextFileVar);
  Result := ModalResult = mrOk;
end;
```

下面来看看利用记录文件存储用户信息。

## 1.5　记录文件存储用户信息

记录文件是一种操作更为灵活的文件类型。它的特点是允许同时为读和写打开，而且由于记录文件中每条记录的长度固定，所以可随机存取。

对记录文件中数据的处理，类似数据库表中的记录。可以把 ObjectPascal 的数据结构保存到磁盘文件中，也可以将数据从文件中直接读入数据结构中，保存 Pascal 数据结构的文件称为记录文件。

利用记录文件存储用户信息，需要声明一个 record 类型的记录结构 TUserRec，该记录结构包含两个字段，一个是账号 UserName，另一个是密码 Password，均为字符串类型，代码如下：

```
type
  TUserRec = record
    UserName: string;       // 账号
    Password: string;       // 密码
  end;
```

■　注意：包含 Ansi 字符串、变量、类实例、接口或动态数组的记录不能写入类型文件。

### 1.5.1　记录文件的打开和关闭

#### 1. 记录文件的变量声明

首先来看一下记录文件类型的变量声明，代码如下：

```
Var
  RecordFileVar: file of TUserRec;
```

其中的 TUserRec 就是前面定义的用户信息记录结构类型。

#### 2. 打开记录文件

记录文件的打开和创建同文本文件一样也需要关联和初始化两个步骤，与文本文件惟一的不同是不能使用 Append 过程。

（1）关联文件变量与文件名调用 AssignFile 标准过程，代码如下：

```
AssignFile ( RecordFileVar , FileName ) ;
```

与打开文本文件一样，其中的 FileName 是记录文件的文件名，如果该文件名没有包括路径，系统将会在当前目录下查找。

（2）初始化读写有如下两种方式。

■ Reset 为读写打开记录文件并把文件指针移动到文件首，代码如下：

```
Reset ( RecordFileVar ) ;
```

■ Rewrite 为写创建一个新记录文件，代码如下：

```
Rewrite ( RecordFileVar ) ;
```

### 3. 关闭记录文件

与关闭文本文件一样，记录文件的关闭也是调用 CloseFile 过程，代码如下：

```
CloseFile ( RecordFileVar ) ;
```

## 1.5.2 记录文件的读写

记录文件缺省是以读写方式打开的，如果想以只读或只写方式打开，则需要修改 FileMode 的值。FileMode 是一个全局变量，对它的每次修改都将影响所有 Reset 的操作，因此在打开自己的文件后应及时还原它的值。

在 Delphi 的 SysUtils 单元中，有如下对文件打开模式的常量定义：

```
fmOpenRead       = $0000;
fmOpenWrite      = $0001;
fmOpenReadWrite  = $0002;
```

在打开文件之前，就可以将 FileMode 变量的赋值为上面的某一常量，以使文件按所设定的模式打开。文件打开模式常量的定义及说明如表 1-2 所示。

表 1-2　　文件打开模式常量定义及说明

| 打开模式 | 值 | 说明 |
|---|---|---|
| fmOpenRead | 0 | 只读打开 |
| fmOpenWrite | 1 | 只写打开 |
| fmOpenReadWrite | 2 | 读写打开 |

记录文件打开后，就可以通过 Read 过程和 Write 过程对文件进行读写操作了。

## 1.5.3 记录文件的编辑

由于记录文件的结构是自定义的，因此它的编辑与文本文件的编辑不同，一般需要针对其特定的结构提供编辑工具。记录文件的操作过程和方法如表 1-3 所示，具体的应用将在本书的第 2 章进行介绍。

表 1-3　　文件操作过程和方法

| 方法 | 说明 |
|---|---|
| AssignFile | 把一个外部文件名和一个文件变量相关联 |