

网 络 与 数 据 库 编 程 百 例 丛 书

Delphi 6

网络与数据库编程 百例

郑荣贵 编著



中国电力出版社

www.infopower.com.cn

网 络 与 数 据 库 编 程 百 例 丛 书

Delphi 6

网络与数据库编程 百例

郑荣贵 编著

中国电力出版社

内 容 提 要

本书通过 100 个经典实例来介绍如何利用 Delphi 开发网络和数据库应用程序。在每章中都有重点介绍的实例，并在这些例子后面辅以必要的技术内幕，来讲解相关的编程技巧。本书中的实例应用技术涉及范围较广，包括基本的网络操作组件、数据库操作组件，同时结合 Windows API、DOS Shell 以及 dbExpress、XML、Web Broker 和 WebSnap 等技术。通过本书的学习，读者可在编程过程中充分享受编程的乐趣，同时掌握更多的编程技巧。

本书适用于有一定 Delphi 编程基础并想在开发网络和数据库方面获得更多技巧的编程人员。同时也适用于广大的 Delphi 编程爱好者。

中国电力出版社出版、发行

(北京三里河路 6 号 100044 <http://www.infopower.com.cn>)

汇鑫印务有限公司印刷

各地新华书店经售

*

ISBN 7-900038-63-9/TP-50

2003 年 7 月第一版 2003 年 7 月北京第一次印刷

787 毫米×1092 毫米 16 开本 24.5 印张 548 千字

定价 39.00 元

版 权 所 有 翻 印 必 究

(本书如有印装质量问题，我社发行部负责退换)

前 言

Delphi 是由软件界著名的 Borland 公司开发的可视化编程语言，由于其可视化程度高，代码简洁易读，因此是最容易入门的编程语言之一。经过 Borland 公司的不断完善、第三方资源的大量加入以及广大爱好者的大力支持，其 Delphi 6.0 版本已经成为功能十分强大、使用人数众多的面向对象的编程工具。

Delphi 6.0 于 2001 年 6 月推出。与以前的版本相比，Delphi 6.0 被称为“新一代电子商务应用开发工具”，其在数据库、网络、跨平台开发方面增加了很多激动人心的功能。

本书针对 Delphi 6.0 在开发网络和数据库方面精心选取了 100 个编程实例，为了体现实用的原则，全书以实例为主进行介绍，并将部分 Delphi 6.0 的新增功能融入到实例中介绍。每一章中都有重点介绍的例子，同时在技术内幕中加以扩展，以帮助读者更好地了解相关的技术。

全书共 2 章，分别是网络篇和数据库篇。

本书的配套光盘中包含本书的所有实例，全部经过调试、运行通过，数据库方面路径设置根据不同的需要，读者可以自己动手配置。

参加本书编写的有陈显锋、吴斌、倪成群等同志，这些作者都有着多年的 Delphi 使用以及编程经验，希望这本书能够使无论是 Delphi 的编程新手，还是有编程经验的用户，都有不同的收益。虽然作者投入了大量的时间和精力进行编写、创作，但百密难免一疏，其中不免有疏漏之处，恳请广大的热心读者批评指正。

作 者
2003 年 4 月

目 录

前言

第 1 章 网 络 篇

实例 1 收发 ASCII 码消息	1
实例 2 利用 Socket 进行通信	15
实例 3 测试本地机器状态	16
实例 4 测试是否安装网络协议	17
实例 5 获取机器的 NetBios	17
实例 6 局域网中的 UDP 聊天程序	18
实例 7 用 Cookies 统计访问次数	34
实例 8 使用 NMDayTime 显示主机 IP 地址	34
实例 9 使用 WebSnap 向导来创建应用程序	35
实例 10 利用 ISAPI 建立网络页面程序浏览	36
实例 11 FTP 下载	36
实例 12 通过应用程序设置机器名	55
实例 13 获取磁盘空间信息	55
实例 14 从网邻上拷文件	56
实例 15 查看网邻上的共享资源	56
实例 16 我的 WEB 浏览器	57
实例 17 删除浏览器临时文件	73
实例 18 实现超级连接	74
实例 19 读取 IE 收藏夹内网址的信息	74
实例 20 实现本页面的超级链接	75
实例 21 在 Delphi 6.0 中开发 PING 程序	76
实例 22 获得本机的 IP 地址	89
实例 23 检测是否安装了 IP 协议	90
实例 24 查找工作组中的所有计算机	90
实例 25 获取路由信息	91
实例 26 显示局域网中的工作组名称	91
实例 27 通过计算机名获得 IP	107
实例 28 获取所有用户的信息	107
实例 29 访问网络计算机的注册表	108

实例 30	利用 NMURL 组件获取解码数据	108
实例 31	电子邮件发送程序	109
实例 32	检验 E-mail 是否合法	128
实例 33	利用组件转化数据库数据	128
实例 34	利用 NMUUProcessor 组件实现编码	129
实例 35	利用 WEBBrowser 组件装载网络图片	129
实例 36	OLE 服务器	130
实例 37	利用 DLL 来显示数据	143
实例 38	创建应用程序服务器	144
实例 39	Delphi 6.0 的 ActiveX 发布	145
实例 40	实现 Delta 状态的无关性	145
实例 41	分布式应用程序	146
实例 42	调用 DLL 显示数据	164
实例 43	明细表的实现	165
实例 44	调用 MTS 客户端程序	165
实例 45	利用向导生成 WEB 应用程序	166
实例 46	利用 TcpServer 和 TcpClient 发送消息	167
实例 47	开发 CGI 程序	184
实例 48	开发 ISAPI 程序	185
实例 49	使用 PageProducer 组件显示	185
实例 50	实现域名到 IP 的转换	186

第 2 章 数据库篇

实例 51	dbExpress 的使用	187
实例 52	利用向导生成数据库图表分析	204
实例 53	动态给 Paradox 表加密码	204
实例 54	利用 DBLookupComboBox 组件来显示数据	205
实例 55	使用 DecisionCube 组件来操作数据	205
实例 56	DBGRID 列的控制	206
实例 57	给 DBGRID 中的字段设定不同的颜色	221
实例 58	使用 DBGRID 编辑数据	221
实例 59	DBGrid 的下拉列表	222
实例 60	如何在 DBGrid 组件中显示图像	223
实例 61	把数据从文本中导入数据库	223
实例 62	显示数据库中的 BLOB 图片	239
实例 63	利用 DBChart 组件生成动态图表	239
实例 64	数据库事务操作	240

实例 65	数据表的浏览	241
实例 66	查看 BDE 信息	241
实例 67	获取 BDE 版本信息	254
实例 68	数据表的打印预览	255
实例 69	获取数据库别名的实际路径	255
实例 70	获取 BDE 错误信息	256
实例 71	利用 STRGRID 显示数据	256
实例 72	利用 IBQuery 组件来操作 InteBase 数据库	275
实例 73	利用 DecisionQuery 组件显示数据表	275
实例 74	利用 DecisionGrid 组件显示数据	276
实例 75	获取数据上的存储过程信息	277
实例 76	利用 SQL 查询	277
实例 77	使用计算字段	297
实例 78	利用 ADOQuery 组件操作数据库	298
实例 79	TADOTable 组件的应用	299
实例 80	显示数据库的 MEMO 记录	299
实例 81	利用线程同步进行查询	300
实例 82	利用 DBCTRLGrid 组件来显示数据	314
实例 83	利用 SQL 操作数据库表	314
实例 84	操作 InteBase 数据库	315
实例 85	设置数据库标签	315
实例 86	浏览数据库信息	316
实例 87	显示多个数据库表	330
实例 88	访问数据库时间字段	331
实例 89	利用 SQLClientDataSet 组件操作数据库	331
实例 90	字符列表编辑	332
实例 91	如何过滤数据表	333
实例 92	操作 SQL SERVER 存储过程	356
实例 93	利用 BatchMove 组件实现数据转移	357
实例 94	使用 ADOCommand 组件操作数据库	358
实例 95	利用 UpdateSQL 组件更新记录	358
实例 96	转换数据库	359
实例 97	利用 IBClientDataSet 组件操作数据库	377
实例 98	利用 IBSQL 组件来显示数据	377
实例 99	获取数据表信息	378
实例 100	获取数据库信息	379

第 1 章 网 络 篇

本章介绍了利用 Delphi 6.0 自带的 WEB 组件, 结合对 Windows API 函数及 SHELL API 函数的调用, 列举了如何实现跟远程机器的交互的一些例子, 还介绍了开发基于 WINDOWS 的 OLE 应用程序服务器技术和当前非常流行的分布式网络应用程序的开发。例子实现的功能包括 WEB 浏览、FTP 下载、网络聊天以及工作组内信息查看等。同时在各章节后面详细介绍了应用到的技术组件或部分 API 函数的内幕。

实例 1 收发 ASCII 码消息

实例目标

现在创建一个可以用来实现基于局域网和广域网上发送简单的 ASCII 码文本消息功能的程序。应用这个程序, 我们就可以很方便地实现对任意 IP 的计算机进行文本消息的发送与接收功能, 类似于 Windows NT/2000 中的【Net Send】命令。我们能够对指定 IP 的计算机收发基于 ASCII 码的消息, 而需要的仅仅是对方的 IP。

在这个应用程序中将使用 Delphi 6.0 组件面板 FastNet 选项页中的 TNMMsg 组件和 TNMMsgServ 组件来实现收发 ASCII 码文本消息的功能。

程序运行的最终显示结果如图 1-1 所示。

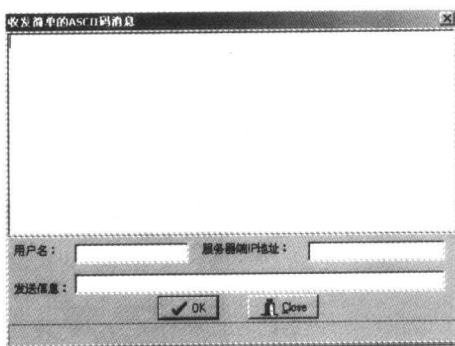


图 1-1 收发 ASCII 码消息运行结果

实现技术

在该应用程序中, 考虑到每一方都要实现接收消息和发送消息, 所以应用程序需要同时有 TNMMsg 组件和 TNMMsgServ 组件。其中 TNMMsg 组件在每次发送消息时都会先触发一个 Connected 事件, 在连接成功后会自动处理一个 DisConnected 事件, 从而断开连接, 等待下一次触发事件。

在设置 TNMMsg 组件和 TNMMsgServ 组件的 Port 时, 应使它们的设置保持一致, 这是因为在多台计算机的用户之间同时存在着发送消息时, 各自的 TNMMsg 组件会向设置成该 Port 的目标机器发送消息, 同时 TNMMsgServ 组件也是监视自己的 Port 等待其他机

器的消息发送。

TNMMsgServ 组件在监视的端口接收到信息时,调用它的 OnMSG 事件来处理该信息,该事件在 Delphi 6.0 中是这样定义的:

```
property OnMSG: TMsgEvent;
```

其中 TMsgEvent 类定义如下:

```
type TMSGEvent = procedure(Sender: TComponent; const sFrom, sMsg: string)
of object;
```

传递的参数 sFrom 用于获取发送者地址,参数 sMsg 用于获取接收到的信息。

除了上述调用外,还利用 TNMMsgServ 组件的 OnStatus 事件来获取 TNMMsgServ 组件当前的状态信息。

利用 TNMMsg 组件的 PostIt 方法来发送指定的信息到服务器端,该方法定义如下:

```
function PostIt(const sMsg: string): string;
```

其中的参数 sMsg 是用于传递客户端需要传送的信息。

除此以外,还利用 TNMMsgServ 组件和 TNMMsg 组件的其他事件和方法来捕捉信息在发送和接收过程中的各种信息,以及各种的异常处理。

实现步骤

建立一个新工程

(1) 启动 Delphi 6.0,在此 IDE 环境下选择【File】|【New】|【Application】菜单项,新建一个新的主窗体界面,如图 1-2 所示。

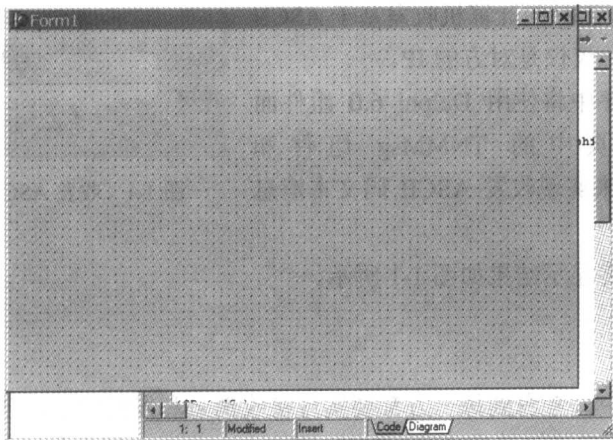


图 1-2 创建一个新的主窗体界面

(2) 选择 Delphi 6.0 组件栏上的 FastNet 选项页,在 FastNet 选项页中,一共有 19 个常用的组件,这些组件大部分都是由 Windows Socket 所发展出来的,所以在所用的 Windows 操作系统上必须要有 WinSock.DLL 网络组件才行。在本实例中,主要用的是 TNMMsg 组件和 TNMMsgServ 组件,如图 1-3 所示,其他的组件将在以后的章节中陆续

介绍。



图 1-3 FastNet 选项页

(3) 选择 TNMMsg 组件  和 TNMMsgServ 组件  加入到应用程序窗体上，具体位置如图 1-4 所示。同时添加一个 Memo 组件 ，一个 StatusBar 组件 ，一个 BitBtn 组件 ，两个 Pannel 组件 ，三个 Label 组件  和三个 Edit 组件  到应用程序窗体上。

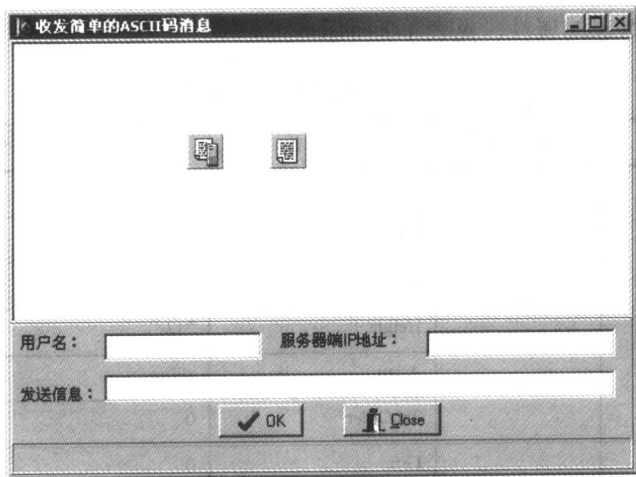


图 1-4 显示 TNMMsg 和 TNMMsg 组件的图标界面

窗体间的属性设置

主程序窗体中各个组件的属性设置如表 1-1~表 1-9 所示：

设置 TForm 对象的属性，见表 1-1 所示。

表 1-1 TForm 对象的属性

组 件	组件说明	属 性	属 性 值
TForm1	TForm	Left	287
		Top	218
		Width	485
		Height	345
		Caption	收发简单的 ASCII 码消息
		Color	clBtnFace
		Font.Charset	DEFAULT_CHARSET
		Font.Color	clWindowText
		BorderStyle	bsDialog
		Font.Height	-11
		Font.Name	MS Sans Serif

续表

组 件	组件说明	属 性	属 性 值
TForm1	TForm	Font.Style	
		OldCreateOrder	False
		OnCreate	FormCreate
		PixelsPerInch	96
		TextHeight	13

设置 TPanel 对象的属性, 见表 1-2 所示。

表 1-2 TPanel 对象的属性

组 件	组件说明	属 性	属 性 值
Tpanel	TPanel	Left	0
		Top	0
		Width	477
		Height	217
		Align	alTop
		TabOrder	0
Panel2	TPanel	Left	0
		Top	217
		Width	477
		Height	98
		Align	alClient
		TabOrder	1

设置 TMemo 对象的属性, 见表 1-3 所示。

表 1-3 TMemo 对象的属性

组 件	组件说明	属 性	属 性 值
Memo1	TMemo	Left	1
		Top	1
		Width	475
		Height	215
		Align	alClient
		TabOrder	0

设置 TLabel 对象的属性, 见表 1-4 所示。

表 1-4 TLabel 对象的属性

组 件	组件说明	属 性	属 性 值
Label2	TLabel	Left	8
		Top	8
		Width	48
		Height	13
		Caption	用户名:
Label3	TLabel	Left	216
		Top	8
		Width	46
		Height	13
		Caption	主机 IP:

设置 TEdit 对象的属性, 见表 1-5 所示。

表 1-5 TEdit 对象的属性

组 件	组件说明	属 性	属 性 值
Edit1	TEdit	Left	72
		Top	8
		Width	121
		Height	21
		TabOrder	0
Edit2	TEdit	Left	264
		Top	8
		Width	177
		Height	21
		TabOrder	1
Edit3	TEdit	Left	72
		Top	40
		Width	329
		Height	21
		TabOrder	2

设置 TBitBtn 对象的属性, 见表 1-6 所示。

表 1-6 TBitBtn 对象的属性

组 件	组件说明	属 性	属 性 值
BitBtn1	TBitBtn	Left	408
		Top	40
		Width	65
		Height	25
		TabOrder	3
		OnClick	BitBtn1Click
		Kind	bkOK

设置 TStatusBar 对象的属性, 见表 1-7 所示。

表 1-7 TStatusBar 对象的属性

组 件	组件说明	属 性	属 性 值
StatusBar1	TStatusBar	Left	1
		Top	72
		Width	475
		Height	25
		Panels	<>
		SimplePanel	False

设置 TNMMsg 对象的属性, 见表 1-8 所示。

表 1-8 TNMMsg 对象的属性

组 件	组件说明	属 性	属 性 值
NMMsg1	TNMMsg	Port	6711
		ReportLevel	0
		OnDisconnect	NMMsg1Disconnect
		OnConnect	NMMsg1Connect
		OnInvalidHost	NMMsg1InvalidHost
		OnHostResolved	NMMsg1HostResolved
		OnStatus	NMMsg1Status
		OnConnectionFailed	NMMsg1ConnectionFailed
		OnMessageSent	NMMsg1MessageSent
		Left	200
		Top	72

设置 TNMMSGServ 对象的属性, 见表 1-9 所示。

表 1-9 TNMMSGServ 对象的属性

组 件	组件说明	属 性	属 性 值
NMMSGServ1	TNMMSGServ	Port	6711
		ReportLevel	0
		OnStatus	NMMSGServ1Status
		OnClientContact	NMMSGServ1ClientContact
		OnMSG	NMMSGServ1MSG
		Left	136
		Top	72

应用程序代码实现

进行窗体初始化和控件插入

在这一部分中，主要完成窗体中一个属性的设置工作，同时完成对窗口的工作，如定义窗口类型、窗口大小和初始位置等属性，以及声明全局变量和声明将要调用的事件处理过程等。

具体的实现代码如下：

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, Psock, NMMSG, StdCtrls, Buttons, ComCtrls;

type
  TForm1 = class(TForm)
    NMMsg1: TNMMsg;
    NMMSGServ1: TNMMSGServ;
    Panel1: TPanel;
    Panel2: TPanel;
    Memo1: TMemo;
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    Edit3: TEdit;
    Label3: TLabel;
    BitBtn1: TBitBtn;
    StatusBar1: TStatusBar;
    procedure BitBtn1Click(Sender: TObject);
    procedure NMMsg1MessageSent(Sender: TObject);
    procedure NMMsg1Connect(Sender: TObject);
    procedure NMMsg1Disconnect(Sender: TObject);
    procedure NMMsg1ConnectionFailed(Sender: TObject);
    procedure NMMsg1InvalidHost(var Handled: Boolean);
    procedure NMMsg1HostResolved(Sender: TComponent);
    procedure NMMsg1Status(Sender: TComponent; Status: String);
    procedure NMMSGServ1ClientContact(Sender: TObject);
    procedure Edit3KeyDown(Sender: TObject; var Key: Word;
      Shift: TShiftState);
    procedure NMMSGServ1MSG(Sender: TComponent; const sFrom, sMsg: String);
    procedure NMMSGServ1Status(Sender: TComponent; Status: String);
    procedure FormCreate(Sender: TObject);
  private
    { Private declarations }
```

```
public
{ Public declarations }
end;
const
    MaxTime = 60000;
var
    Form1: TForm1;

implementation

{$R *.dfm}
procedure TForm1.FormCreate(Sender: TObject);
begin
    //初始化窗体时, 窗体位于屏幕的中间
    form1.Left := (Screen.Width - form1.Width) div 2;
    form1.Top := (Screen.Height - form1.Height) div 2;
end;
```

调用组件和处理连接异常

这部分的代码主要用来完成对 Delphi 6.0 中的 TNMMsg 组件和 TNMMsgServ 组件的调用。在这里将要实现连接 TNMMsg 组件和 TNMMsgServ 组件, 同时对连接过程中产生的中断异常并进行相应的处理。本部分还将完成 TNMMsg 组件和 TNMMsgServ 组件在服务器和客户端之间的联系作用, 检测当前输入的服务器的状态和对客户端进行 IP 解析。

具体的实现代码如下:

```
procedure TForm1.NMMsg1MessageSent(Sender: TObject);
begin
    //在状态栏中提示消息的时候发送
    StatusBar1.SimpleText := '消息已送出!';
end;

procedure TForm1.NMMsg1Connect(Sender: TObject);
begin
    //在状态栏中显示与服务器连接是否成功
    StatusBar1.SimpleText := '与服务器连接成功!';
end;

procedure TForm1.NMMsg1Disconnect(Sender: TObject);
begin
    //在状态栏中显示与服务器断开连接
    StatusBar1.SimpleText := '与服务器断开连接!';
end;

procedure TForm1.NMMsg1ConnectionFailed(Sender: TObject);
begin
    //用于提示连接服务器失败
    ShowMessage('连接服务器失败!');
```



```
end;

procedure TForm1.NMMsg1InvalidHost(var Handled: Boolean);
var
    ObjName:String;      //定义一个字符型变量，用于保存新的目标地址
begin
    if InputQuery('目标地址有误，请重新输入！','输入新的目标地址:',ObjName) then
        NMMsg1.Host :=ObjName;
        Handled :=true;      //执行相应的操作
        //StatusBar1.SimpleText :='目标地址有误，操作失败！';
end;

procedure TForm1.NMMsg1HostResolved(Sender: TComponent);
begin
    //在状态栏中显示域名解析或输入的 IP 地址是否正确
    StatusBar1.SimpleText :='目标地址解析成功！';
end;

procedure TForm1.NMMsg1Status(Sender: TComponent; Status: String);
begin
    //清空状态栏
    StatusBar1.SimpleText :='';
    //设置状态栏的文字为客户端 NMMsg 组件的状态值
    StatusBar1.SimpleText :=Status;
end;

procedure TForm1.NMMSGServ1ClientContact(Sender: TObject);
begin
    NMMsgServ1.ReportLevel :=Status_Basic;    //
    NMMsgServ1.TimeOut :=MaxTime;
    StatusBar1.SimpleText :='与客户端连接成功！';
end;

procedure TForm1.NMMSGServ1MSG(Sender: TComponent; const sFrom,
    sMsg: String);
begin
    //在服务器端显示接收到的消息
    Memo1.Lines.Add(sFrom+'->'+sMsg);
end;

procedure TForm1.NMMSGServ1Status(Sender: TComponent; Status: String);
begin
    //清空状态栏
    StatusBar1.SimpleText :='';
    //在状态栏中显示服务器端 TNMMsgServ 组件的状态
    StatusBar1.SimpleText :=Status;
end;
```

☞ 响应文本框输入事件

这一部分的代码，主要用于处理对用户客户端输入各种值的响应，完成了对键盘回车键按下事件的处理过程。

具体的实现代码如下：

```
procedure TForm1.Edit3KeyDown(Sender: TObject; var Key: Word;
  Shift: TShiftState);
begin
    //判断文本框中是否敲入回车键
    if Key=13 then
        //调用按钮激活事件
        BitBtn1Click(Sender);
end;
```

☞ 编写 ASCII 消息发送事件处理

以下代码是本程序响应用户单击【OK】按钮时的事件，用以完成 ASCII 码消息的发送工作，同时完成了对程序面板上几个文本框输入的初步判断。利用 TNMMsg 组件和 TNMMsgServ 组件的功能来发送特定的文本消息，是本实例的核心部分。应该注意本部分中对 TNMMsg 组件和 TNMMsgServ 组件方法的调用。

具体的实现代码如下：

```
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
    //判断用户名是否为空
    if Edit1.Text = '' then ShowMessage('请输入用户名!');
    if Edit1.Text = '' then Exit;
    //判断目标地址是否为空
    if Edit2.Text = '' then ShowMessage('请输入目标地址!');
    if Edit2.Text = '' then Exit;
    //用于连接的主机地址——(IP 地址或域名)
    NMMsg1.Host := Edit2.Text ;
    //发送端名称
    NMMsg1.FromName := Edit1.Text ;
    NMMsg1.PostIt(Edit3.Text); //发送信息
    Edit3.Text := ''; //文本框清空
end;
```

☞ 编写程序退出事件代码

处理用户单击【Close】按钮的响应，在用户使用完成后退出应用程序，将从内存和 CPU 列表中清除相关的进程。

具体实现的代码如下：

```
procedure TForm1.BitBtn2Click(Sender: TObject);
```