Delphi 中可重用组件的制作

韦向理 （杭州九莲新村 310012）

摘要：本文讨论了 DLLs、ActiveX 控件及在 Delphi 中制作这两种组件的方法和特点。

关键词：DLL ActiveX 接口 OLE

制作组件是一个复杂的代码编写过程，过去多是用 C++ 编写。现在可以轻松地在 Delphi 上编写这些组件，如 DLLs（dynamic link libraries），ActiveX 控件，Web-based Active Documents 等。本文介绍其中的两种。

1. DLLs

Wondows 环境下的动态链接库有以下特点：

(1)不同的程序使用相同的 DLL，只需将在内存中装载一次。

(2)如果 DLL 中的子例程有相同的参数，运行程序可以使用新版本的 DLL，而不需要重新编译。

(3)DLLs 独立于编程语言。

由此，当一些复杂的算法或一些复杂的窗体是多个应用程序所需时，将它们存储在 DLL 中，当同时运行多个使用该 DLL 的程序时，可以减少执行文件的规模，并节省一些内存。

另外，对于应用程序比较庞大、需要不断更新或修改的情况，将它们划分成多个可执行部分与 DLL，这样可以只对局部相关部分进行修改，而不必改动、编译整个应用程序。

此外，大多数 Windows 编程环境，包括大部分宏语言，都允许程序员调用存储在 DLL 中的子例程。

在应用程序中，几乎每个子模块都可能用到 about 弹出窗口。而这些窗口的内容往往是相似的：精致的公司图标、版本号及公司的广告词等。为使这样的 about 窗口风格统一、易于维护，可将其做成 DLL，由每个小应用分别调用。About DLL 工程文件如下：

```
library About;
uses
    uAbout in 'uAbout.pas';//所使用的执行单元文件名
{$R *.RES}
exports //输出的子例程
ShowAbout;

begin //初始化
end.
```

传统 windows 中的动态链接库的编写，需要两个标准函数：LibMain 和 WEP，用于启动和关闭 DLL：开锁 DLL 数据段、分配内存、初始化变量；释放内存。Delphi 用自己独特的方式实现这两个标准函数的功能，即在工程文件中的 Begin...End 部分添加了初始化代码。

执行单元 uAbout 的 Pascal 代码如下：

```
unit uAbout;
interface
uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs, StdCtrls, But
  tons, ExtCtrls;
type
    TfrmAbout = class(TForm)
    btbOK: TBitBtn;
    Image1: TImage;
    lbVersion: TLabel;
    lbModuleName: TLabel;
  private
  public
  end;

function ShowAbout(ModuleName: PChar): Boolean; Stdcall;//输出子例程声明
//Stdcall:使用标准的 Win32 参数传递技术
implementation
{$R *.DFM}

function ShowAbout(ModuleName: PChar): Boolean;//传入的参数是模块名称
```

```
var
  AboutForm: TfrmAbout;
begin
  AboutForm: = TfrmAbout. Create(Application);
  try
    with AboutForm do
    begin
      lbModuleName. Caption: = ModuleName;
      if ShowModal = mrOK then
        Result: = True
      else
        Result: = False;
    end;
  finally
    AboutForm. Free;//
  end;
end;
end.
```

以上是用 Delphi 在 DLL 中实现窗体重用的例子。在 Delphi 中制作 DLL 可以不必直接使用 Windows API，因为它提供了丰富的函数。

2. ActiveX 控件

控件通常是窗口，带有定义自己操作的相关代码。不同种类的控件之间的主要区别在区：控件与应用程序其余部分之间的交互方式不同，即接口不同。

ActiveX 控件包含许多元素：VCL 控件、属性、方法、事件、相关的类型库。它使用了 OLE Automation 接口来显示其属性、方法和事件。在 Delphi 3 中，由于提供了 ActiveX Control Wizard 而使得制作 ActiveX 控件变得简单。它自动地将 ActiveX 定义的协议及对象通信机制包装在一个 VCL 控件上，生成内含该对象的 OCX（仍称 OCX，但定义与过去不同）。例如，用 ActiveX Control Wizard 将 Delphi 的一个 VCL 控件、SpinEdit、包装成 ActiveX 控件时，Wizard 将自动为它生成一个 ActiveX 库工程文件、类型库、执行单元（Pacal 文件）及一个包含所有类型库的声明的 Pascal 文件等。其中 Delphi 向控件添加的类型库极其重要，它包含了库的 GUID、控件的 GUID 以及一些复杂的接口定义，而这些代码可以被 Wizard 自动生成。以下是它的接口定义：

```
  ISpinEditX = interface(IDispatch)
    ['{F000C681 - 1926 - 11D2 - 9CEC - F013BBB48438}'] //ISpinEditX 接口的 GUID
    function Get-AutoSelect: WordBool; safecall;
    procedure Set-AutoSelect(Value: WordBool); safecall;
      .
      .
      .
    property Cursor: Smallint read Get-Cursor write Set-Cursor;
  end;

  ISpinExitXDisp = dispinterface
    ['{F000C681 - 1926 - 11D2 - 9CEC - F013BBB48438}']
    property AutoSelect: WordBool dispid 1;
    property AutoSize: WordBool dispid 2;
      .
      .
      .
    property Cursor: Smallint dispid 30;
  end;
```

Wizard 生成了一个 dual interface（ISpinEditX 接口和 ISpinEditXDisp 接口），使用两种不同方式（动态或静态）定义控件的对象方法和属性，也就是说，访问该服务器的对象方法和属性可以在两种方式之间选择：使用滞后联编和由 dispinterface 提供的机制，或使用初期联编和基于 VTables 接口类型的机制。

下面是控件的事件接口：

```
  ISpinEditXEvents = dispinterface // 事件接口的 GUID
    ['{F000C683 - 1926 - 11D2 - 9CEC - F013BBB48438}']
    procedure OnChange; dispid 1;
    procedure OnClick; dispid 2;
    procedure OnDblClick; dispid 3;
    procedure OnKeyPress(var Key: Smallint); dispid 4;
  end;
```

在类型库中，还定义新类 TSpinEditX = class(TOleControl)，它封装了 ActiveX 控件的基本功能，提供了访问 SpinEditX 控件的内部 OLE 对象的方法。

如果需要添加新属性、对象方法或事件，可使用 Add To Interface 命令，在对话框中填写。这样做的好处是：对话框中输入的声名会自动添加到控件的类型库中，使之保持同步。所需的工作只是填充对象方法。如果需要在 Delphi 中添加控件的属性页也很容易。

该控件编译后生成 OCX 文件，经注册，即可在 Delphi、Visal Basic 及其他支持 ActiveX 技术的环境中使用。

（来稿时间：1998 年 7 月）