DELPHI 中利用 API 函数实现多态 FORM

实现异型 FORM 并不是一件难事,本文将向您介绍如何利用 API 函数实现圆角矩形和椭圆形 FORM,并在此基础之上探讨实现 TWINcontrol 类的后裔的异型的实现。

欲改变 FORM 的形状,也就是实现对区域(region)的控制。在 Win32 API 程序参考手册有关区域(region)的定义是这样描述的:它可以是一个矩形,多边形,椭圆形(或者是两者的复合,或者是更多的形状),这些都可以被填充,画图,翻转,结构化并可以得到焦点执行。

由定义得出结论:区域(region)是可以被改变和操纵的,依据我们的需求可定义区域并制作出我们所要求的形状。

应当指出的是区域(region)也能对任何 TWINcontrol 类的后裔定义和控制(不仅仅是 FORMS),就是说,可以将区域(region)的定义运用到向 Tpanel 或 TEdit 这样的对象。在改变 TWINcontrol 类的后裔控件的形状时,需要提供一句柄并创建一些改变形状的函数。

具体实现方式一般分为两步:

1. 定义所需形状的区域边界形状(比如:椭圆形)。

2. 将已定义的区域边界形状运用到窗口。

这里,我们将通过调用 Windows API 函数完成以上两个步骤,下面就具体函数的应用予以说明:

实现第一步:定义区域边界。

在这里将调用三个 WinAPI,这三个函数是:

CreateEllipticRgn()功能是生成椭圆形区域;

CreateRoundRectRgn()功能是生成圆角矩形区域;

CreatePolygonRgn()功能是生成多边形区域,Windows 要确保使其顶点自动相连形成一封闭的区域。

这三个函数通过返回的指针变量标识所生成的区域将被第二步所应用。这些函数在 Delphi 中的函数声明及参数含义说明如下:

(1)椭圆形区域生成函数:

函数原形:HRGN CreateEllipticRgn(int nLeftRect,int nTopRect,int nRightRect,int nBottomRect);

参数含义:

nLeftRect,nTopRect:区域的左上角坐标;

nRightRect, nBottomRect:区域的右下角坐标;

(2)圆角矩形区域生成函数:

函数原形:HRGN CreateRoundRectRgn(int nLeftRect,int nTopRect,int nRightRect,int nBottomRect,int nWidthEllipse,int nHeightEllipse);

参数含义:

nLeftRect, nTopRect:区域的左上角坐标;

nRightRect, nBottomRect:区域的右下角坐标;

nWidthEllipse, nHeightEllipse:圆角的宽度和高度;

(3)多边形区域生成函数:

函数原形:HRGN CreatePolygonRgn(CONST POINT *lppt,int cPoints, int fnPolyFillMode);

参数含义:

Lppt:指向一个 POINT 类型的数组,该数组定义多边形顶点;

CPoints:定义数组中顶点数;

FnPolyFillMode:定义填充模式,可选值为 ALTERNATE 或 WINDING。

实现第二步:将返回的 HRGN 类型的区域值被设置窗口区域函数调用。

设置窗口区域函数:

函数原形:int SetWindowRgn(HWND hWnd, HRGN hRgn, BOOL bRedraw);

参数说明:

hWnd:指向所操作的窗口的句柄;

hRgn:所给区域句柄;

bRedraw:是否显示重画窗口的标志。

在每一个函数的最后都需要调用 SetWindowRgn 函数,然后由 Windows 操作系统实现区域的各种形状的设置并显示。

以下将测试的 FORM 的整个源代码列出,在 FORM 上添加了四个按钮分别控制实现:椭圆形,圆角矩形,等边多边形和星形;一个 Tpanel 控件为了演示 TWINcontrol 类的后裔的区域定义和控制;一个 SpinEdit 控件定义多边形和星形的顶点连接数目。

源程序:

```
unit form_statue;
interface
uses
 Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
Dialogs,
 StdCtrls, ExtCtrls, Spin;
type
 TForm1 = class(TForm)
    Button1: TButton;
    SpinEdit1: TSpinEdit;
    Panel1: TPanel;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
  procedure DrawRndRectRegion(wnd : HWND; rect : TRect);
  procedure DrawEllipticRegion(wnd : HWND; rect : TRect);
  procedure DrawPolygonRegion(wnd : HWND; rect : TRect; NumPoints
: Integer; DoStarShape : Boolean);
    procedure Button1Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
```

```
    private
      { Private declarations }
       rgn : HRGN;
       rect : TRect;
    public
      { Public declarations }
    end;
  var
    Form1: TForm1;
  implementation
  {$R *.DFM}
    procedure TForm1.DrawRndRectRegion(wnd : HWND; rect : TRect);
  begin
    rgn := CreateRoundRectRgn(rect.left, rect.top, rect.right, rect.
bottom, 30, 30);
    SetWindowRgn(wnd, rgn, TRUE);
  end;
  procedure TForm1.DrawEllipticRegion(wnd : HWND; rect : TRect);
  begin
    rgn := CreateEllipticRgn(rect.left, rect.top, rect.right, rect.b
ottom);
    SetWindowRgn(wnd, rgn, TRUE);
  end;

procedure TForm1.DrawPolygonRegion(wnd : HWND; rect : TRect; NumPoint
s : Integer; DoStarShape : Boolean);
  const
    RadConvert = PI/180;
    Degrees    = 360;
    MaxLines = 100;
  var
    x, y,
    xCenter,
    yCenter,
    radius,
    pts,
    I     : Integer;
    angle,
    rotation: Extended;
    arPts : Array[0..MaxLines] of TPoint;
  begin
    xCenter := (rect.Right - rect.Left) div 2;
    yCenter := (rect.Bottom - rect.Top) div 2;
    if DoStarShape then
```

```
        begin
         rotation := Degrees/(2*NumPoints);
         pts := 2 * NumPoints;
        end
         else
        begin
    rotation := Degrees/NumPoints;              //得到每个顶点的度数
     pts := NumPoints ;
        end;
     radius := yCenter;
     for I := 0 to pts - 1 do begin
        if DoStarShape then
         if (I mod 2) = 0 then
            radius := Round(radius/2)
         else
            radius := yCenter;
        angle := ((I * rotation) + 90) * RadConvert;
        x := xCenter + Round(cos(angle) * radius);
        y := yCenter - Round(sin(angle) * radius);
        arPts[I].X := x;
        arPts[I].Y := y;
     end;
     rgn := CreatePolygonRgn(arPts, pts, WINDING);
     SetWindowRgn(wnd, rgn, TRUE);
    end;
    procedure TForm1.Button1Click(Sender: TObject);
    begin
     DrawRndRectRegion(Form1.Handle, Form1.ClientRect);
    end;
    procedure TForm1.Button4Click(Sender: TObject);
    begin
     DrawPolygonRegion(Panel1.Handle, Panel1.BoundsRect, SpinEdit1.Va
lue, True);
    end;
    procedure TForm1.Button3Click(Sender: TObject);
    begin
     DrawEllipticRegion(Form1.Handle, Form1.ClientRect);
    end;
    procedure TForm1.Button2Click(Sender: TObject);
    begin
     DrawPolygonRegion(Panel1.Handle, Panel1.BoundsRect, SpinEdit1.Va
lue, False);
    end;
```

end.
源程序在 PWIN98+DELPHI5 环境下调试成功,可以直接引用。