How to create a non-rectangular TPanel

```
unit ShapedPanel;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls,
  Forms, ExtCtrls;

type
  TShapedPanel = class(TCustomControl)
  private
    FBorderColor: TColor;
    IsLoaded: Boolean;
    FBorderWidth: Integer;
    FRgn, FRgn2: HRGN;
    RgnBrush: TBrush;
    FFillColor: TColor;
    procedure SetFillColor(const Value: TColor);
    function GetFillColor: TColor;
    procedure MakeRegion;
    procedure SetBorderColor(Value: TColor);
    procedure WMSize(var Msg: TMessage); message WM_SIZE;
  protected
    procedure Paint; override;
    procedure CreateWnd; override;
  public
    constructor Create(AOwner: TComponent); override;
    destructor Destroy; override;
  published
    property BorderColor: TColor
      read FBorderColor write SetBorderColor default clBlack;
    property BorderWidth: Integer
      read FBorderWidth write FBorderWidth default 2;
    property FillColor: TColor
      read GetFillColor write SetFillColor;
    property Height default 200;
    property Width default 200;
    property OnClick;
    property OnContextPopup;
    property OnDblClick;
    property OnEndDock;
    property OnEndDrag;
    property OnEnter;
    property OnExit;
    property OnMouseDown;
    property OnMouseMove;
    property OnMouseUp;
    property OnResize;
    property OnStartDrag;
  end;

procedure Register;

implementation

procedure Register;
begin
  RegisterComponents('EXS', [TShapedPanel]);
end;

constructor TShapedPanel.Create(AOwner: TComponent);
begin
  inherited Create(AOwner);
  ControlStyle := [
    csCaptureMouse, csClickEvents, csOpaque, csDoubleClicks
  ];
  Width := 200;
  Height := 200;
  RgnBrush := TBrush.Create;
  RgnBrush.Color := clBlack;
  FFillColor := clWhite;
  IsLoaded := False;
  FBorderWidth := 2;
  FBorderColor := clBlack;
  FRgn := 0;
  FRgn2 := 0;
end;

destructor TShapedPanel.Destroy;
begin
  DeleteObject(FRgn);
  DeleteObject(FRgn2);
  inherited;
end;

procedure TShapedPanel.CreateWnd;
begin
  inherited;
  MakeRegion;
  IsLoaded := True;
  {IsLoaded is to make sure MakeRegion is not called before there
  is a Handle for this control, but it may not be nessary}
end;

procedure TShapedPanel.MakeRegion;
var
  x4, y2: Integer;
  FPoints: array[0..5] of TPoint;
begin
  {I moved the Region creation to this procedure so it can be called
  for WM_SIZE}
  SetWindowRgn(Handle, 0, False);
  {this clears the window region}
  if FRgn <> 0 then
  begin
    {Make sure to Always DeleteObject for a Region}
    DeleteObject(FRgn);
    DeleteObject(FRgn2);
    FRgn := 0;
    FRgn2 := 0;
   end;
   x4 := Width div 4;
   y2 := Height div 2;
   FPoints[0] := Point(x4, 0);
   FPoints[1] := Point(Width - x4, 0);
   FPoints[2] := Point(Width, y2);
   FPoints[3] := Point(Width - x4, Height);
   FPoints[4] := Point(x4, Height);
   FPoints[5] := Point(0, y2);
   FRgn := CreatePolygonRgn(FPoints, 6, WINDING);
   SetWindowRGN(Handle, FRgn, True);
   FRgn2 := CreatePolygonRgn(FPoints, 6, WINDING);
   {FRgn2 is used for FrameRgn in Paint}
end;

procedure TShapedPanel.WMSize(var Msg: TMessage);
var
  TmpClr: TColor;
begin
  inherited;
  if IsLoaded then
  begin
    TmpClr := Canvas.Brush.Color;
    Canvas.Brush.Color := FFillColor;
    MakeRegion;
    FillRgn(Canvas.Handle, FRgn2, Canvas.Brush.Handle);
    Paint;
    Canvas.Brush.Color := TmpClr;
  end;
end;

procedure TShapedPanel.Paint;
var
```

```
  TmpClr: TColor;
begin
  inherited;
  if IsLoaded then
  begin
    TmpClr := Canvas.Brush.Color;
    Canvas.Brush.Color := FFillColor;
    MakeRegion;
    FillRgn(Canvas.Handle, FRgn2, Canvas.Brush.Handle);
    FrameRgn(
      Canvas.Handle,
      FRgn2,
      RgnBrush.Handle,
      FBorderWidth,
      FBorderWidth
     );
    Canvas.Brush.Color := TmpClr;
  end;
end;

procedure TShapedPanel.SetBorderColor(Value: TColor);
begin
  if FBorderColor <> Value then
  begin
    FBorderColor := Value;
    RgnBrush.Color := FBorderColor;
    Paint;
  end;
end;

procedure TShapedPanel.SetFillColor(const Value: TColor);
begin
  if FFillColor <> Value then
  begin
    FFillColor := Value;
    Paint;
  end
end;

function TShapedPanel.GetFillColor: TColor;
begin
  Result := FFillColor;
end;

end.
```

This component creates a hexagonal panel that does not display its caption.

Original resource:	The Delphi Pool
Author:	Eddie Shipman
Added:	2009-09-07
Last updated:	2010-03-16