How to create a resizable TPanel with a size grip

> ***QUESTION***
>
> How can I create a *TPanel* that can be resized by grip in the lower right corner (just like the grip, the *TStatusBar* has)?

Answer 1

Try this one. It may need some refinement in painting the grip.

```
unit SizeablePanel;

interface

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

type
  TSizeablePanel = class(TPanel)
  private
    FDragging: Boolean;
    FLastPos: TPoint;
  protected
    procedure Paint; override;
    procedure MouseDown(Button: TMouseButton; Shift:
      TShiftState; X, Y: Integer); override;
    procedure MouseMove(Shift: TShiftState; X, Y: Integer); override;
    procedure MouseUp(Button: TMouseButton; Shift: TShiftState;
      X, Y: Integer); override;
  end;

procedure Register;

implementation

procedure Register;
begin
  RegisterComponents('PBGoodies', [TSizeablePanel]);
end;

procedure TSizeablePanel.MouseDown(Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if (Button = mbLeft) and ((Width - x ) < 10) and
    ((Height - y ) < 10) then
  begin
    FDragging := TRue;
    FLastPos := Point(x, y);
    MouseCapture := true;
    Screen.cursor := crSizeNWSE;
  end
  else
    inherited;
end;

procedure TSizeablePanel.MouseMove(Shift: TShiftState; X, Y: Integer);
var
  r: TRect;
begin
  if FDragging then
  begin
    r := BoundsRect;
    SetBounds( r.left, r.top, r.right - r.left + X - FlastPos.X,
    r.bottom - r.top + Y - Flastpos.Y );
    FLastPos := Point( x, y );
  end
```

```
  else
  begin
    inherited;
    if ((Width - x ) < 10) and ((Height - y ) < 10) then
      Cursor := crSizeNWSE
    else
      Cursor := crDefault;
  end;
end;

procedure TSizeablePanel.MouseUp(Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if FDragging then
  begin
    FDragging := False;
    MouseCapture := false;
    Screen.Cursor := crDefault;
  end
  else
    inherited;
end;

procedure TSizeablePanel.Paint;
var
  x, y: Integer;
begin
  inherited;
  Canvas.Font.Name := 'Marlett';
  Canvas.Font.Size := 10;
  Canvas.Brush.Style := bsClear;
  x := clientwidth - canvas.textwidth('o');
  y := clientheight - canvas.textheight('o');
  canvas.textout( x, y, 'o' );
end;

end.
```

Tip by Peter Below

Answer 2

Here's a component that will do that and also looks like it has a statusbar at the bottom:

```
unit SizeGripPanel;

interface

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

type
  TSizeGripPanel = class(TPanel)
  private
    FAllowMove, FAllowSize, FShowSizeGrip: Boolean;
    procedure SetAllowMove(Value: Boolean);
    procedure SetAllowSize(Value: Boolean);
    procedure SetShowSizeGrip(Value: Boolean);
  protected
    procedure WMNCHitTest(var Msg: TWMNCHitTest); message WM_NCHITTEST;
    procedure Paint; override;
  published
    property ShowSizeGrip: Boolean
      read FShowSizeGrip write SetShowSizeGrip;
    property AllowMove: Boolean
      read FAllowMove write SetAllowMove;
    property AllowSize: Boolean
      read FAllowSize write SetAllowSize;
  end;

procedure Register;
```

```
implementation

procedure Register;
begin
  RegisterComponents('Samples', [TSizeGripPanel]);
end;

procedure TSizeGripPanel.WMNCHitTest(var Msg: TWMNCHitTest);
var
  ScreenPt: TPoint;
  MoveArea: TRect;
  HANDLE_WIDTH: Integer;
  SIZEGRIP: Integer;
begin
  {This code came from Lou's Tip of the Day web site ... with changes}
  HANDLE_WIDTH := BevelWidth;
  Sizegrip := 19;
  inherited;
  if not (csDesigning in  ComponentState) then
  begin
    ScreenPt := ScreenToClient(Point(Msg.Xpos, Msg.Ypos));
    MoveArea := Rect(
      HANDLE_WIDTH,
      HANDLE_WIDTH,
      Width - HANDLE_WIDTH,
      Height - HANDLE_WIDTH
    );
    if FAllowSize then
    begin
      {left side}
      if (ScreenPt.x < HANDLE_WIDTH) then
        Msg.Result := HTLEFT
      {top side}
      else
      if (ScreenPt.y < HANDLE_WIDTH) then
        Msg.Result := HTTOP
      {right side}
      else
      if (ScreenPt.x >= Width - HANDLE_WIDTH) then
        Msg.Result := HTRIGHT
      {bottom side}
      else
      if (ScreenPt.y >= Height - HANDLE_WIDTH) then
        Msg.Result := HTBOTTOM
      {top left corner}
      else
      if (ScreenPt.x < Sizegrip) and (ScreenPt.y < Sizegrip) then
        Msg.Result := HTTOPLEFT
      {bottom left corner}
      else
      if (ScreenPt.x < Sizegrip) and
        (ScreenPt.y >= Height - Sizegrip) then
        Msg.Result := HTBOTTOMLEFT
      {top right corner}
      else
      if (ScreenPt.x >= Width - Sizegrip) and
        (ScreenPt.y < Sizegrip) then
        Msg.Result := HTTOPRIGHT
      {bottom right corner}
      else
      if (ScreenPt.x >= Width - Sizegrip) and
        (ScreenPt.y >= Height - Sizegrip) then
        Msg.Result := HTBOTTOMRIGHT;
    end;
    {no sides or corners, this will do the dragging}
// !! PJ ->
//    else
//    if PtInRect(MoveArea, ScreenPt) and FAllowMove then
//      Msg.Result := HTCAPTION;
    if (Msg.Result = HTCLIENT) and
      PtInRect(MoveArea, ScreenPt) and FAllowMove then
      Msg.Result := HTCAPTION;
// !! PJ <-
```

```
  end;
end;

procedure TSizeGripPanel.Paint;
const
  Alignments: array[TAlignment] of Longint
    = (DT_LEFT, DT_RIGHT, DT_CENTER);
var
  Rect: TRect;
  TopColor, BottomColor: TColor;
  FontHeight: Integer;
  LineBeg, LineEnd: TPoint;
  Flags: Longint;
  R: TRect;

  procedure AdjustColors(Bevel: TPanelBevel);
  begin
    TopColor := clBtnHighlight;
    if Bevel = bvLowered then
      TopColor := clBtnShadow;
    BottomColor := clBtnShadow;
    if Bevel = bvLowered then
      BottomColor := clBtnHighlight;
  end;

  procedure DrawCorner(pane: TRect);
  begin
    {Got this code from a Codeguru post. It was a CStatusBar
    descendant and written in C}
    OffsetRect(pane, - 1, - 1);
    with Canvas do
    begin;
      Canvas.Pen.Color := clBtnHighlight;
      MoveTo(pane.right - 15, pane.bottom);
      LineTo(pane.right, pane.bottom - 15);
      MoveTo(pane.right - 11, pane.bottom);
      LineTo(pane.right, pane.bottom - 11);
      MoveTo(pane.right - 7, pane.bottom);
      LineTo(pane.right, pane.bottom - 7);
      MoveTo(pane.right - 3, pane.bottom);
      LineTo(pane.right, pane.bottom - 3);
      Canvas.Pen.Color := clBtnShadow;
      MoveTo(pane.right - 14, pane.bottom);
      LineTo(pane.right, pane.bottom - 14);
      MoveTo(pane.right - 10, pane.bottom);
      LineTo(pane.right, pane.bottom - 10);
      MoveTo(pane.right - 6, pane.bottom);
      LineTo(pane.right, pane.bottom - 6);
      MoveTo(pane.right - 2, pane.bottom);
      LineTo(pane.right, pane.bottom - 2);
      MoveTo(pane.right - 13, pane.bottom);
      LineTo(pane.right, pane.bottom - 13);
      MoveTo(pane.right - 9, pane.bottom);
      LineTo(pane.right, pane.bottom - 9);
      MoveTo(pane.right - 5, pane.bottom);
      LineTo(pane.right, pane.bottom - 5);
      MoveTo(pane.right - 1, pane.bottom);
      LineTo(pane.right, pane.bottom);
    end;
  end;

begin
  Rect := GetClientRect;
  if BevelOuter <> bvNone then
  begin
    AdjustColors(BevelOuter);
    Frame3D(Canvas, Rect, TopColor, BottomColor, BevelWidth);
  end;
  Frame3D(Canvas, Rect, Color, Color, BorderWidth);
  if BevelInner <> bvNone then
  begin
    AdjustColors(BevelInner);
    Frame3D(Canvas, Rect, TopColor, BottomColor, BevelWidth);
  end;
```

```
  with Canvas do
  begin
    Brush.Color := Color;
    FillRect(Rect);
    Brush.Style := bsClear;
    Font := Self.Font;
    FontHeight := TextHeight('W');
    with Rect do
    begin
      Top := ((Bottom + Top) - FontHeight) div 2;
      Bottom := Top + FontHeight;
    end;
    Flags := DT_EXPANDTABS or DT_VCENTER or Alignments[Alignment];
    Flags := DrawTextBiDiModeFlags(Flags);
    DrawText(Handle, PChar(Caption), - 1, Rect, Flags);
    Rect := GetClientRect;
    if FShowSizeGrip then
    begin
      R := Rect;
      R.Top := Height - 19;
      R.Left := Rect.Left + BevelWidth;
      R.Bottom := Rect.Bottom - BevelWidth;
      R.Right := Rect.Right - BevelWidth;
      AdjustColors(BevelOuter);
      {Always have sunken statusbar! If you want a bar that is
      raised when your panel is sunken, use this line, instead:
      Frame3D(Canvas, R, BottomColor, TopColor, 1);}
      Frame3D(Canvas, R, clBtnShadow, clBtnHighlight, 1);
      DrawCorner(R);
    end;
  end;
end;

procedure TSizeGripPanel.SetAllowMove(Value: Boolean);
begin
  if Value <> FAllowMove then
  begin
    FAllowMove := Value;
    Invalidate;
  end;
end;

procedure TSizeGripPanel.SetAllowSize(Value: Boolean);
begin
  if Value <> FAllowSize then
  begin
    FAllowSize := Value;
//    FShowSizeGrip := Value;   !! PJ
    Invalidate;
  end;
end;

procedure TSizeGripPanel.SetShowSizeGrip(Value: Boolean);
begin
  if Value <> FShowSizeGrip then
  begin
    FShowSizeGrip := Value;
    Invalidate;
  end;
end;

end.
```

Tip by Eddie Shipman (with modifications by Peter Johnson)

See *Tip #93* for a different approach to making a resizeable panel that changes the panel's window style.

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