Custom Buttons in Delphi Cheat Sheet

Initial Setup Guide

Getting Started

Step-by-Step Instructions:

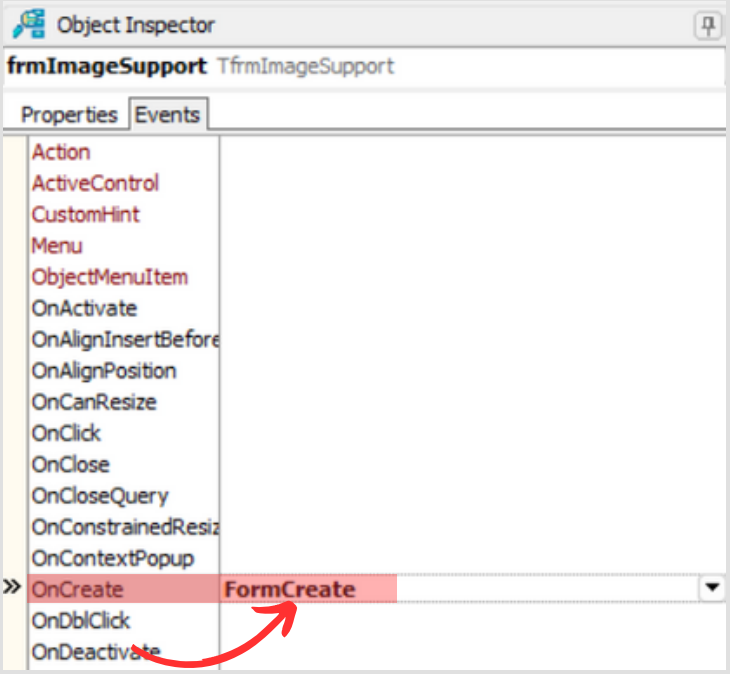
Step 1: Ensure 'CustomButtons.dcu' is in your program folder.

CBD_C.pas	2024/01/14 19:35	Delphi Source File	1 KB
CBD_P.dpr	2024/02/17 14:39	Delphi Project File	1 KB
CBD_P.dproj	2024/03/01 20:50	Delphi Project File	6 KB
CBD_P.dproj.local	2024/03/01 20:50	LOCAL File	2 KB
CBD_P.exe	2024/03/30 00:54	Application	1 101 KB
CBD_P.identcache	2024/03/29 14:23	IDENTCACHE File	1 KB
CBD_P.res	2024/01/14 17:41	Compiled Resourc...	6 KB
CBD_U.dcu	2024/03/30 00:54	DCU File	7 KB
CBD_U.dfm	2024/03/29 18:31	Delphi Form	2 KB
CBD_U.pas	2024/03/29 20:31	Delphi Source File	3 KB
CustomButtons.dcu	2024/03/29 14:18	DCU File	45 KB

Step 2: Reference 'CustomButtons' in the 'uses' section of your unit.

```
uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, CustomButtons, pngimage, jpeg;
```

Step 3: Navigate to the Form's OnCreate event.



Step 4: Declare your custom panel's variable as TCustomButtons locally or globally

```
  private
    { Private declarations }
    pnlCustom : TCustomButtons;
  public
    { Public declarations }
  end;

var
  frmBasic: TfrmBasic;

implementation

{$R *.dfm}

procedure TfrmBasic.FormCreate(Sender: TObject);
var
  pnlCustom : TCustomButtons;
begin
  pnlCustom := TCustomButtons.Create(Panel1);

end;
```

OR

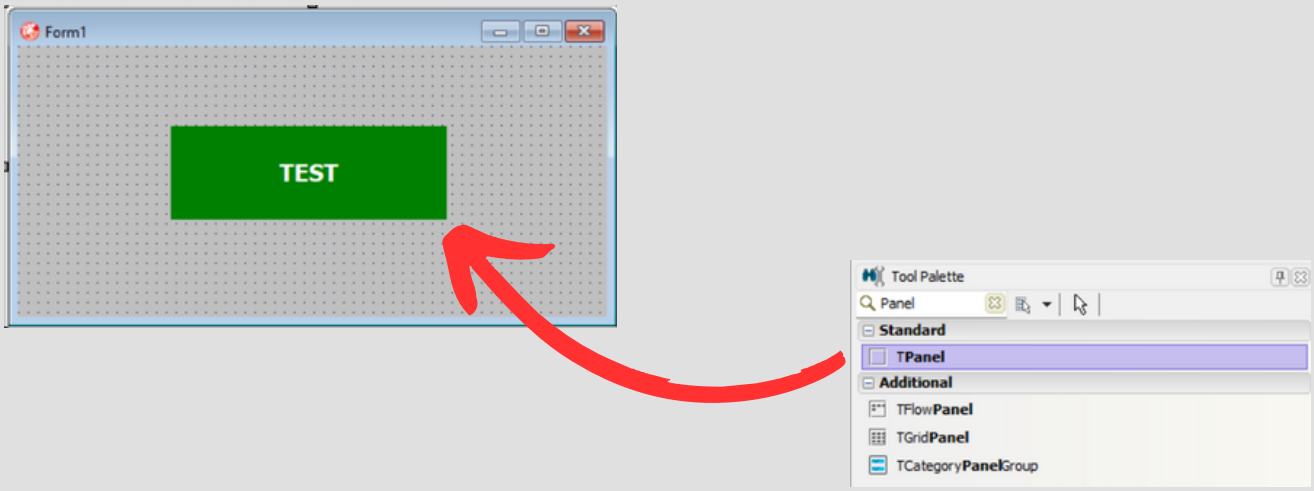
Step 5: Initialize your panel with TCustomButtons.Create in the FormCreate procedure:
pnlCustom := TCustomButtons.Create(YourPanelName);

```
procedure TfrmBasic.FormCreate(Sender: TObject);
var
  pnlCustom : TCustomButtons;
begin
  pnlCustom := TCustomButtons.Create(Panel1);

end;
```

Step 6: Start with the code to add custom designs to the panel on your form

```
procedure TfrmBasic.FormCreate(Sender: TObject);
var
  pnlCustom : TCustomButtons;
begin
  pnlCustom := TCustomButtons.Create(Panel1);

  with pnlCustom do
  begin
    Hover.Color := clBlue; // Colour when hovering over the panel

    Click.Color := clSkyBlue;  // Colour when clicking on the panel

    HorizontalGradient(clSkyBlue, clBlue); // Static panel colour (Gradient)
                                           // when program runs

    BorderRadius := 70;  // Initial border radius

    Click.BoxShadow(10, 10, clSilver);  // Boxshadow when clicking on the panel
  end;

end;
```

***NOTE:** Ensure that you have dragged a panel from the object inspector onto your form, enter this panel's name into **Step 5** "YourPanelName" NOT the variable "pnlCustom"



Example Code

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, CustomButtons; // Step 2: Reference the class here

type
  TForm1 = class(TForm)

  private
    { Private declarations }
    pnlCustom: TCustomButtons; // Step 4: Declare your custom panel variable locally or globally
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}

procedure TForm1.FormActivate(Sender: TObject);
begin
  // Step 5: Initialize your panel with TCustomButtons.Create
  pnlCustom := TCustomButtons.Create(YourPanelName); // Replace 'YourPanelName' with the actual panel
                                                     // name in your design
  // Step 6: After initializing the object (panel), you can start with the code  for example:
  with pnlCustom do
  begin
    Hover.Color := clBlue;
    VerticalGradient(clSkyBlue, clSilver);
    Click.Color := clWhite;
    Hover.BoxShadow(5, 5, clGray);
    // ....
  end;
end;
end.
```

Static Features

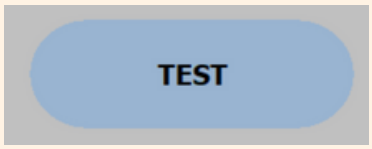
Color

```
pnlCustom.Color := clBlue;
pnlCustom.Color := RGB(55, 55, 55);
```

Sets the panel colour. Replace clBlue or RGB with any values

BorderRadius

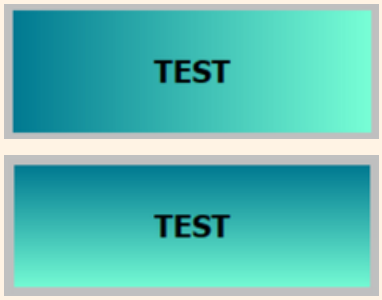
```
pnlCustom.BorderRadius := 50;
pnlCustom.BorderRadius := 20;
```



Adjusts the corner radius for rounded edges. Increase value for larger curvers.

Gradients

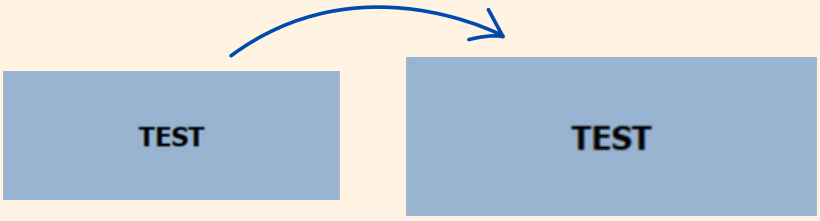
```
pnlCustom.HorizontalGradient(FromColor, ToColor);
pnlCustom.VerticalGradient(FromColor, ToColor);
//Can use any colour. Exp: RGB(0, 0, 0) or clBlue
```



Adds a gradient effect. Use two colour constants for the start and en colours

ScalyBy

```
pnlCustom.ScaleBy(20);
pnlCustom.ScaleBy(40);
```

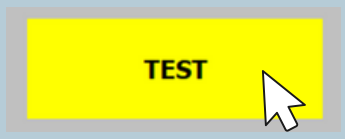


Enlarges the panel by the give value while keeping it's position

Hover Features

Color

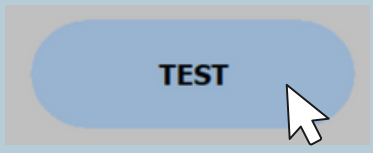
```
pnlCustom.Hover.Color := clBlue;
pnlCustom.Hover.Color := RGB(100, 100, 100);
```



Changes the panel color when the mouse hovers over it.

BorderRadius

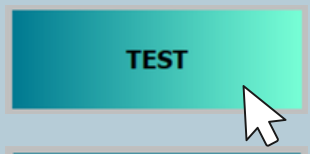
```
pnlCustom.Hover.BorderRadius := 50;
pnlCustom.Hover.BorderRadius := 100;
```

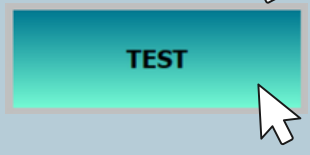


Adjusts the corner radius on hover.

Gradients

```
pnlCustom.Hover.HorizontalGradient(FromColor, ToColor);
pnlCustom.Hover.VerticalGradient(FromColor, Tocolor);
//Can use any colour for example: RGB(126,55,7) or clBlue
```

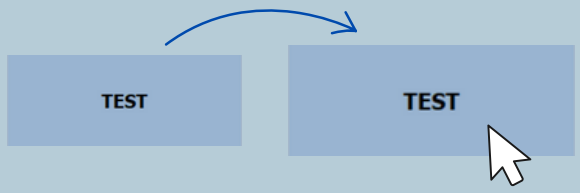




Applies a gradient effect on hover.

ScalyBy

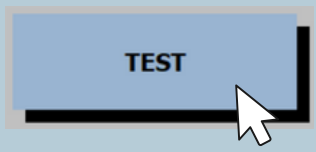
```
pnlCustom.Hover.ScaleBy(10);
pnlCustom.Hover.ScaleBy(50);
```



Scales the panel outwards on hover.

BoxShadow

```
pnlCustom.Hover.BoxShadow(10, 10, clSilver);
pnlCustom.Hover.BoxShadow(5, 5, RGB(244, 244, 244));
```

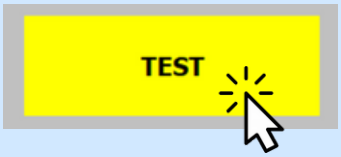


Creates a shawdow effect when the mouse hover over the panel.

Click Features

Color

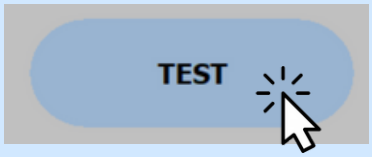
```
pnlCustom.Click.Color := clBlue;
pnlCustom.Click.Color := RGB(100, 234, 423);
```



Changes the panel color when the mouse clicks on it.

BorderRadius

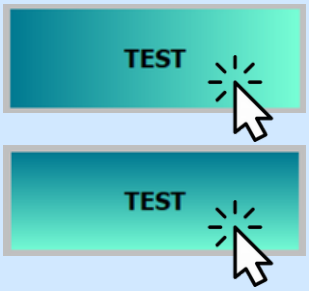
```
pnlCustom.Click.BorderRadius := 50;
pnlCustom.Click.BorderRadius := 35;
```



Adjusts the corner radius on click.

Gradients

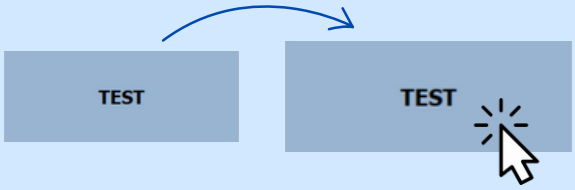
```
pnlCustom.Click.HorizontalGradient(clBlue, clGreen);
pnlCustom.Click.VerticalGradient(clYellow, clRed);
//Can use any colour for example: RGB(0, 0, 0) or clWhite
```



Applies a gradient effect on click.

ScalyBy

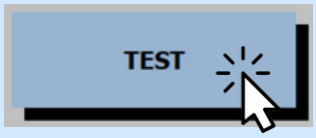
```
pnlCustom.Click.ScaleBy(35);
pnlCustom.Click.ScaleBy(20);
```



Scales the panel outwards on click.

BoxShadow

```
pnlCustom.Click.BoxShadow(5, 3, clSilver);
pnlCustom.Click.BoxShadow(10, 6, RGB(244, 244, 244));
```



Creates a shadow effect when the mouse clicks on the panel.

Additional Notes:

Image Support

The TCustomButtons class seamlessly integrates image support for your custom buttons. Here's how it works:

JPEG Images:

- If you add a JPEG image to your panel, the panel's caption will be automatically replicated as a label in front of the image.
- When you apply colour or gradient changes to the panel, the image will adjust accordingly.
- Other features like BoxShadow, BorderRadius, and ScaleBy won't affect the image.

PNG Images:

- PNG images will always remain visible on top of other effects.
- Hover and click events will work behind the transparent areas of the PNG.
- Gradients and colour changes will show in the blank spaces of the PNG image.