How to implement drag and drop at design time

> ***QUESTION***
>
> I'm writing a component (descendant of *TCustomControl*) that owns a button and a panel. When the button is clicked, the panel will either show or hide. But now I want to be able to put other controls on the panel (in design-mode). Somehow this does not seem to work. I can put i.e. a button on the panel, and it seems that the button is owned by the panel, because I can no longer drag the button off the panel. But when I then view the form as text (ALT F12), there's no button. Somehow the dropped control is not added to the panel.

I misread your post first, not seeing that you want to put controls on the panel rather than the *CustomControl*. The component below allows this, but you have to be aware that this way you hand the panel over to the IDE, more or less, and your component user can do whatever with it, including removing the panel at design time without removing your control. I've tried to get it to be recreated in this case, but it looks a bit kludgy. This is the 2nd or 3rd time I'm posting a compound component like this, but I feel the solution isn't optimal.

```
unit CustomControl2;

interface

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

type
  TCustomControl2 = class(TCustomControl)
  private
    fbutton: TButton;
    fPanel: TPanel;
    procedure TogglePanel(sender: TObject);
    procedure CreatePanel(AOwner: TComponent);
  protected
    procedure Loaded; override;
    procedure Notification(AComponent: TComponent; Operation: TOperation);
      override;
  public
    constructor Create(AOwner: TComponent); override;
  end;

procedure Register;

implementation

procedure Register;
begin
  RegisterComponents('Samples', [TCustomControl2]);
end;

{ TCustomControl2 }

constructor TCustomControl2.Create(AOwner: TComponent);
begin
  inherited;
  fPanel := nil;
  if (csDesigning in ComponentState) and not
    (csReading in AOwner.ComponentState) then
    {this is true if the component is dropped on the form}
    CreatePanel(AOwner);
    {now the panel is part of the streaming system and will be created from
    the dfm file of the form in the future}
  fbutton := TButton.Create(self);
  {the button is owned by self, not part of streaming}
  fbutton.Parent := self;
  fbutton.SetBounds(0, 0, 100, 25);
  fbutton.Caption := 'Toggle Panel';
  fbutton.OnClick:=TogglePanel;
end;
```

```
procedure TCustomControl2.CreatePanel(AOwner: TComponent);
begin
  fPanel := TPanel.Create(AOwner);
  fPanel.Parent := self;
  fPanel.height := 200;
  fPanel.align := alBottom;
  fPanel.Name := 'MyPanel';
  {giving the panel a name and creating it with the form/ frame as owner
  makes it streamable to dfm file}
end;

procedure TCustomControl2.Loaded;
var
  i: Integer;
begin
  inherited;
  for i := 0 to ControlCount - 1 do
    if Controls[i].Name = 'MyPanel' then
    begin
      fPanel := TPanel(Controls[i]);
      {make fPanel point to the right control}
      break;
    end;
  {this is a kludgy try to recreate the panel in case the user removes it
  at design time. But it only kicks in when the dfm file is being loaded
  again, so e.g. by viewing the form as text, then as form again}
  if not assigned(fPanel) then
  if csDesigning in ComponentState then
  begin
    CreatePanel(Owner);
    {recreate it}
    ShowMessage('Control ''MyPanel'' has been recreated');
  end;
end;

procedure TCustomControl2.Notification(AComponent: TComponent;
  Operation: TOperation);
begin
  inherited;
  if not (csDestroying in ComponentState) then
  if operation = opRemove then
  if AComponent.Name = 'MyPanel' then
  {someone removes the panel without removing our component}
  begin
    ShowMessage('Control ''MyPanel'' should not be removed');
    {we can't recreate it here right now...}
    fPanel := nil;
    {but we can prevent AVs}
    {Loaded recreates it the next time the dfm is being loaded}
  end;
end;

procedure TCustomControl2.TogglePanel(sender: TObject);
begin
  if assigned(fPanel) then
    fPanel.visible := not fPanel.visible;
end;

initialization

RegisterClass(TPanel);

end.
```

Original resource:	The Delphi Pool
Author:	Renate Schaaf
Added:	2009-08-12
Last updated:	2009-08-12