Thursday, May 29, 1997

TSplitterWnd Component:

The files:

SplitterWnd.pas
SplitterWnd.dcr

You will need to rename these files to be as above if they are unzipped with truncated names, otherwise Delphi will tend to choke on them.

To install, make sure that both of the files are in the same directory.

NOTE FOR DELPHI 3.0:

You will need to make a couple of small changes to the unit before it will work properly. Some things changed in the way Delphi handles object streaming, but they are really very simple:

Go to the interface section. Find the object definition for TSplitterWnd. Find the GetChildren procedure.
Change the procedure declaration to read like this:

```
procedure GetChildren( proc : TGetChildProc; root : TComponent );override;
```

Now go to the implementation section and change the getChildren procedure header to read the same:

```
procedure TSplitterWnd.GetChildren( proc: TGetChildProc; root : TComponent );
```

This will make the unit compile and work under Delphi 3.0. Thanks for your patience.

The objects:

TSplitterWnd
TPane
TSplitterWndEditor

TSplitterWnd:

The TSplitterWnd object is the object that actually gets put into the component palette. Make sure the dcr file is in the same directory so that the bitmap gets used, as well.

TSplitterWnd is a descendant of TCustomPanel. I have seen many splitter bar controls out there, and either they really suck or they are not released with source, which is not very useful since nobody can make them do what they want. I decided that the paradigm was all wrong. Instead of making a Splitter BAR, I thought, why don't I make a Splitter WINDOW? That way you can put anything you like into it and it can be any size, etc. This facilitated a lot of things that were hard to do previously, like having multiple splitter bars on one form, all going different directions. So, the idea behind this is that the splitter control is entirely self-contained and you can drop stuff into any one of its several panes.

<u>Properties to be aware of:</u>

- **Orientation**: swHorizontal or swVertical. Horizontal will make horizontal splitter bars in between the panes, and Vertical will make vertical ones.
- **Thickness**: Changes the thickness of the splitter bars. Default is 3.
- **CursorHorz, CursorVert**: Change the drag cursor depending on the orientation.
- **ProportionalResize**: Default is true. When the splitter window is resized, if ProportionalResize is true, the panes will all be reproportioned according to their current relative sizes. If set to false, only the last pane will change size. Beware of making windows too small when this property is false.
- **BarStyle**: Set to either sbSolid or sbCheckered. Default is Checkered. Should be self-explanatory. It changes the behavior of the bar drawing routine when you are dragging a splitter bar.
- **AllowSizing**: Default is true. Turn this off, and you can't drag the bar around during run-time, at least, not visually. PaneSize properties will still change the pane sizes and move the bars around, but those are only accessible programmatically. NOTE: If you turn this off, the cursor will still change shape. If you want your SplitterWnd to be a fixed divider, you should also set the CursorHorz and CursorVert properties to be crDefault.

Events to be aware of:

- **OnDrawDragRect**: ***procedure*** *(Sender : TSplitterWnd;* ***var*** *DrawRect : TRect;* ***var*** *OwnerDraw : Boolean*); This event is fired just before the rectangle is drawn when dragging a splitter bar. The splitter window that fired the event is sent in the Sender parameter. the DrawRect parameter is the rectangle that is about to be drawn. You may change it to whatever you like. Owner draw is a var parameter that can be changed. If it is set to false, the splitter window will not draw its default rectangle. If set to true, it will draw the default rectangle. You can set it to false and do whatever sort of drawing you like using device contexts.
- **OnEraseDragRect**: ***procedure*** *(Sender : TSplitterWnd;* ***var*** *DrawRect : TRect;* ***var*** *OwnerDraw : Boolean*); This event is very similar to the OnDrawDragRect event and is fired just before the drag rectangle is erased. You can change the rectangle that will be erased. If you set OwnerDraw to false, the default drag rectangle erasing procedure won't be called and you can do whatever drawing you like. A note on drawing the rectangle. I suggest using GetDCEx with the DCX_PARENTCLIP flag set so that the rectangle will be drawn on all of the window's children. Any questions about that, just look at the source code.

That brings me to the next object in the unit: TPane.

TPane:

TPane is a descendant of TScrollBox. It is the actual container object for the stuff that goes in between the splitter bars. If you drop a SplitterWnd onto a form and don't actually put anything inside of it, but have some panes, you will still see the splitter bars because they are separating two scroll boxes. More will be said about this later.

Properties to be aware of:

- **PaneSize**: This controls the size of the pane. It doesn't matter whether you are vertical or horizontal. It will always indicate the number of pixels between splitter bars. Changing this at design or runtime will move the splitter bar to the right or underneath. The rightmost or bottommost pane will not let you change this property.
- **MinPaneSize**: This sets the minimum allowable pane size in pixels. If it is set to 0, the minimum pane size will be twice the width of a scroll bar. Default is zero.
- **SplitterWnd**: I have been debating as to whether or not I should allow this to be published, but it doesn't seem to present any real problems at design time. If you have multiple splitter windows on a form, you can move panes from one to the other (theoretically - I hope it doesn't crash any systems. I haven't really tested it) by changing the SplitterWnd property. Also, if you want to create a new pane at runtime and add it, you would do it like this:
```
newPane := TPane.Create(MainForm);
newPane.SplitterWnd := SplitterWnd1;
```
- **PaneIndex**: Did you just move all of your pages around and get them all set up only to realize that you got them in the wrong order? Just change the pane index and it will move it to where you tell it to go. Just like the TabIndex property of a TTabSheet.

TSplitterWndEditor:

This is the component that allows for design-time editing of the component. It's really very straightforward. If there are any questions, you can mail me and I will be happy to answer them.

Using the component:

Here is a short tutorial on how to make the thing work:

Example 1:

Drop a SplitterWnd onto an empty form. It will be named SplitterWnd1.
Align alClient
Right click.
Select New Pane.
Right click again (yes, on the pane will be fine)
Select New Pane.

There are now two panes on the control. You will notice that they are not the same size. There is a reason for that. Suppose you had added a bunch of panes and made them look just the way you wanted them to. Then you decide you need another one. You add a new pane, and OOPS. They all got their sizes changed. The way it works right now is it preserves the relative size percentages of all of the panes when you add a new one. If you delete that new pane, you are right back to where you were before you added it because your relative sizes are all preserved. If you don't like that behavior, you have the source, and you can change it at will.

Now, click on the splitter bar in between the panes. The whole SplitterWnd is selected. Go to the Object Inspector and find the Orientation property. Change it to swVertical. You will notice that the panes retained their relative size ratios.

Example 2:

Don't delete what you just did. Change the orientation back to swHorizontal.
Click on Pane1 (the topmost pane).
Change BorderStyle to bsNone.
Drop a SplitterWnd component onto Pane1.
Align SplitterWnd2 alClient.
Change the orientation to swVertical.
Right click and add two panes.
Right click. Select Equalize Panes.

Run the program. You can size the panes with the mouse. They do have a minimum limit that is at the moment set to twice the size of the system scroll bar. That's the behavior of Explorer.

You have just created a page with two splitter bars going different directions!

Congratulations!

Miscellaneous information:

I wrote this component so that people would learn how to use the Delphi environment to make REAL vcl controls. Many of the ones that I see are very rudimentary and don't work right, giving me access violations at design time, etc. I am not guaranteeing this component to be perfect or fool-proof in any way, but in many cases, it works WITH Delphi rather than AROUND Delphi, making it a little more robust. If there are ANY questions, I am more than happy to answer them by e-mail. I always get components with e-mail addresses in them, and often I don't use them, though it would be easier to simply ask the author what he was up to rather than trying to figure it out. I have tried to comment my source very well. Hopefully it is understandable enough. Please send any questions, no matter how dumb-sounding, to me, and I will do my best to give you a respectful and helpful answer.

Thanks to all the component writers out there who gave me this idea.

If that sounds limiting, I hope you see a psychiatrist.

Good luck.

Chris : ckmonson@burgoyne.com