Silverlight: Binding a child controls property to a property in a user control

Asked 15 years, 11 months ago Modified 14 years, 5 months ago Viewed 10k times

8

If I have a user control defined:

```
public partial class MainFooter : UserControl
{
    public System.Windows.Media.Color BkColor;
}
```

and it's xaml:

```
<UserControl x:Class="Test.MainFooter">
    <Grid x:Name="LayoutRoot">
        <Rectangle x:Name="rctBottom_Background2"
                   HorizontalAlignment="Stretch"
                   Grid.Row="2">
            <Rectangle.Fill>
                <LinearGradientBrush EndPoint="0.82,0.895"
StartPoint="0.911,-0.442">
                    <GradientStop Color="{**How can I bind this to the BkColor
property?}"/**>
                    <GradientStop Color="#00FFFFFF" Offset="1"/>
                </LinearGradientBrush>
            </Rectangle.Fill>
        </Rectangle>
    </Grid>
</UserControl>
```

and used:

```
<MyControls:MainFooter x:Name="rcrMainFooter"
                       BkColor="#FFE2B42A">
</MyControls:MainFooter>
```

How would I go about binding the GradientStop Color in the Rectangle to the value of the it's user controls BkColor property?

.net silverlight

Share Improve this question Follow

asked Jan 7, 2009 at 18:57

Jeremy

46.3k ● 71 ● 211 ● 352

5 Answers

Sorted by: Highest score (default)

8

Often when I've seen this question posed the answer is 'you have to do it in code', which sounded to me like 'Silverlight binding doesn't support this' - so you have to do it '*completely manually*' by setting the property by hand. But thats not the case :

Silverlight binding does support this - its just Silverlight XAML that doesn't.

Here's an example of a `UserControl` that basically wraps a `DataForm`. In the constructor you run the binding which can bind to your 'user control property'. Hopefully if they change the XAML support for this in future then it'll be trivial to come back and fix.

App.xaml

```
<AddressControl MyHeader="Shipping Address"/>
```

AddressControl.xaml

```
<UserControl>
    <DataForm Name="dfAddress" Header="BOUND IN CODE"/>
</UserControl>
```

Optional: indidicate that you've bound the value in code with a comment

AddressControl.xaml.cs

```
publicAddressControl()
{
    InitializeComponent();

    // bind the HeaderProperty of 'dfAddress' to the 'MyHeader' dependency
    // property defined in this file
    dfAddress.SetBinding(DataForm.HeaderProperty,
    new System.Windows.Data.Binding { Source = this,
                                      Path = new PropertyPath("MyHeader") });
}

// standard string dependency property
public string MyHeader
{
    get { return (string)GetValue(MyHeaderProperty); }
    set { SetValue(MyHeaderProperty, value); }
}

public static readonly DependencyProperty MyHeaderProperty =
       DependencyProperty.Register("MyHeader", typeof(string),
       typeof(AddressControl), null);
```

This binds the `MyHeader` property on my `AddressControl` usercontrol to the `Header` property on the dataform. I made it 'My' solely for easy readability - but I'm actually using just 'Header' in my real code.

Real shame we still can't do this in XAML, but its better than what I first attempted which was to capture the `DataContextChanged` events and then manually set things.

Share
Improve this answer
Follow

edited Mar 27, 2010 at 3:26

answered Mar 27, 2010 at 3:21

Simon_Weaver
146k ● 90 ● 673 ● 713

> Many MANY thanks for this answer - it's helped us figure out how to do pass-thru binding on controls. – Quango Aug 12, 2011 at 10:07

1

The only way is to do it programically (e.g. in the change event for the BkColor (assuming its a DependencyProperty) change it in the other places on your control. Alternatively you could use a ControlTemplate and use TemplateBinding. If your UserControl is a workaround for this (e.g. no behavior/methods/events), then replace your user control with a ContentControl and use Template Bindng.

Share Improve this answer Follow

answered Jan 12, 2009 at 2:53

Shawn Wildermuth
7,438 ● 3 ● 25 ● 28

0

Take a look at this blog post: Workaround for missing ElementName in Silverlight 2.0 Binding. It sounds like its a little different from what you're trying to do, but you may be able to wrangle his code to do what you want. Good luck! :-)

Share Improve this answer Follow

answered Jan 7, 2009 at 19:07

Rob
26.3k ● 32 ● 113 ● 157

0

```
<LinearGradientBrush EndPoint="0.82,0.895" StartPoint="0.911,-0.442">
 <GradientStop Color="{TemplateBinding BackGround}" />
 <GradientStop Color="#00FFFFFF" Offset="1"/> </LinearGradientBrush>
```

answered Nov 25, 2009 at 16:47

community wiki
thisisjaiswal

Share

Improve this answer

Follow

0

There is one other option. While it appears that you cannot bind the color values themselves, you can bind the entire Background property, like so:

< Border BorderThickness="2" BorderBrush="Black" CornerRadius="20" Background="{Binding Path=Background}" />

...and then...

```
Public ReadOnly Property Background As Brush
    Get
        Dim lgb As New LinearGradientBrush
        lgb.GradientStops = New GradientStopCollection From {New GradientStop With {.Color = PrimaryColor, .Offset = 0.0}, New GradientStop With {.Color = SecondaryColor, .Offset = 1.0}}
        lgb.StartPoint = New Point(0, 0)
        lgb.EndPoint = New Point(1, 1)
        Return lgb
    End Get
End Property
```

Share Improve this answer Follow

answered Jun 30, 2010 at 14:14

plasne

1