

Re: Updating verbs in the PropertyGrid

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.windowsforms.designtime/2004-10/>

From: Jer_m (jmghost_at_msn.com)

Date: 10/26/04

Date: 25 Oct 2004 19:51:50 -0700

Hi Jacob,

I had the same problem, maybe someone else can shed some light on this....

in the meantime, I have been able to get ISelectionService to work. Is this how your doing it?

```
DesignerVerb verbTestOn;  
DesignerVerb verbTestOff;
```

```
public override initialize(IComponent component)  
{  
    verbTestOn = new DesignerVerb("Test On",new EventHandler(TestOn));  
    verbTestOff = new DesignerVerb("Test Off",new  
EventHandler(TestOff));
```

```
    Verbs.Add(verbTestOn);  
}
```

```
private void TestOn(object sender, EventArgs e)
```

```
{  
    Verbs.Remove(verbTestOn);  
    Verbs.Add(verbTestOff);
```

```
    ISelectionService ss =  
(ISelectionService).GetService(typeof(ISelectionService));
```

```
    ss.SetSelectedComponents(new IComponent[] { this.Control.Parent }  
);  
    ss.SetSelectedComponents(new IComponent[] { this.Control } );  
}
```

```
private void TestOff(object sender, EventArgs e)
```

```
{  
    Verbs.Add(verbTestOn);  
    Verbs.Remove(verbTestOff);
```

```
    ISelectionService ss =
(ISelectionService).GetService(typeof(ISelectionService));

    ss.SetSelectedComponents(new IComponent[] { this.Control.Parent }
);
    ss.SetSelectedComponents(new IComponent[] { this.Control } );
}
```

kind regards
Jerron

"Jacob" <js@hotmail.com> wrote in message news:<edmWiq\$tEHA.220@TK2MSFTNGP15.phx.gbl>...

> Thank you for your great answer, but it's a solution to a little bit
> different problem. I've had no problems, enabling/disabling and setting the
> verbs to visible/invisible. The problem is **ADDING** verbs and **REMOVING** verbs
> from the collection. The changes made to the collection don't seem to get
> updated until after the control is deselected and reselected again. I poked
> around some things with Reflector and found that the PropertyGrid keeps its
> own array of verbs, different from the collection that is edited in the
> ControlDesigner. That seems to be the problem. The PropertyGrid's array
> doesn't get updated when a property is changed on the control and the
> control is still selected until after you deselect it and reselect it again.
> Do you have an answer for this one?

>
> Jacob

>
> P.S. – I've tried using the ISelectionService to deselect all controls and
> reselect my control after the control's property is changed but it still
> won't update the PropertyGrids verbs unless you deselect and reselect
> yourself.

>
>
> "Joey Calisay" <hcalisay@codex-systems.com> wrote in message
> news:eE7M4QxtEHA.2804@TK2MSFTNGP14.phx.gbl...

> > I forgot to include the key thing: utilize the ISelectionService and
> > provide
> > an eventhandler for its SelectionChanged. When your control is selected,
> > refresh the designerverbs:

> >
> > private void OnSelectionServiceSelectionChanged(object sender, EventArgs
> > e)

> >
> > {
> >
> > ISelectionService ss = GetService(typeof(ISelectionService)) as
> > ISelectionService;

> >
> > if (ss.SelectionCount == 1 && ss.PrimarySelection is
> > CdxTabControlAdvanced)
> >
> > {

```
>>
>> this.ModifyMenu();
>>
>> }
>
>> }
>>
>>
>>
>> Added wireups on the Initialize method of the designer:
>>
>>
>>
>> IDesignerHost host = GetService(typeof(IDesignerHost)) as IDesignerHost;
>>
>> ISelectionService ss = GetService(typeof(ISelectionService)) as
>> ISelectionService;
>>
>> if (ss != null)
>>
>> {
>>
>> ss.SelectionChanged += new
>> EventHandler(e_OnISelectionServiceSelectionChanged);
>>
>> }
>>
>>
>> sorry for the mess...
>> "Joey Calisay" <hcalisay@codex-systems.com> wrote in message
>> news:eW1qXJxtEHA.3200@TK2MSFTNGP09.phx.gbl...
>>> you just have to have a reference of the designerverbs on the designer
> class
>>> level to modify their properties on your OnComponentChanged eventhandler
>>>
>>> public class CdxTabControlAdvancedDesigner :
>>> System.Windows.Forms.Design.ParentControlDesigner
>>>
>>> {
>>>
>>> private DesignerVerb m_verbRemoveTab = null;
>>>
>>> private DesignerVerb m_verbAddTab = null;
>>>
>>> private void OnComponentChanged(object sender, ComponentChangedEventArgs
>> e)
>>>
>>> {
>>>
>>> //Modify the designerverbs here
>>>
```

```
> > if (e.Component = this.Component) this.ModifyMenu();
> >>
> >> }
> >>
> >> protected virtual void ModifyMenu()
> >>
> >> {
> >>
> >> m_verbRemoveTab.Enabled = false;
> >>
> >> if (this.CdxTabControlAdvanced.aaTabButtons.Count > 0)
> >>
> >> {
> >>
> >> m_verbRemoveTab.Enabled = true;
> >>
> >> }
>
> >> }
> >>
> >> public CdxTabControlAdvancedDesigner()
> >>
> >> {
> >>
> >> this.m_verbAddTab = new DesignerVerb("Add Tab", new
> >> EventHandler(this.e_OnAddTab));
> >>
> >> this.m_verbRemoveTab = new DesignerVerb("Remove Tab", new
> >> EventHandler(this.e_OnRemoveTab));
> >>
> >> base.Verbs.AddRange(new DesignerVerb[] {this.m_verbAddTab,
> >> this.m_verbRemoveTab});
> >>
> >> }
> >>
> >> protected override void Dispose(bool bIsDisposing)
> >>
> >> {
> >>
> >> IComponentChangeService cs = GetService(typeof(IComponentChangeService))
> > as
> >> IComponentChangeService;
> >>
> >> if (cs != null)
> >>
> >> {
> >>
> >> cs.ComponentChanged -= new
> > ComponentChangedEventHandler(OnComponentChanged);
> >>
> >> }
```

```
>>>
>>> base.Dispose(bIsDisposing);
>>>
>>> }
>>>
>>> public override void Initialize(IComponent Icomponent)
>>>
>>> {
>>>
>>> base.Initialize(Icomponent);
>>>
>>> IComponentChangeService cs = GetService(typeof(IComponentChangeService))
> as
>>> IComponentChangeService;
>>>
>>> if (cs != null)
>>>
>>> {
>>>
>>> cs.ComponentChanged +=new
> ComponentChangedEventHandler(OnComponentChanged);
>>>
>>> }
>>>
>>> ModifyMenu();
>>>
>>> }
>
>>> }
>>>
>>> "Jacob" <js@hotmail.com> wrote in message
>>> news:eA$aAwrtEHA.1336@TK2MSFTNGP10.phx.gbl...
>>> > I'm working on a component (with a custom designer) where the designer
> verbs
>>> > change based on what a property is set to. The problem is,... it
>>> > appears
>>> > that the designer keeps its own cache of the verbs such that when I
> change
>>> > the property of my component, I have to click off the component and
>>> > then
>>> > back onto it to update the verb links listed in the PropertyGrid. I
> can't
>>> > explain it any simpler than that. If you've tried to dynamically change
> the
>>> > verbs before, changing the context menu verbs is easy, but changing the
>>> > links in the property grid without reselecting the component is proving
> to
>>> > be very difficult. Anybody have the answer?
>>> >
>>> >
>>> > Thanks,
```

```
> > > Jacob
> > >
> > >
> > > P.S. – Some sample code...
> > >
> > > internal class MyCustomDesigner : ControlDesigner
> > >
> > > {
> > >
> > >
> > >
> > >
> > >
> > >
> > > public override DesignerVerbCollection Verbs
> > >
> > > {
> > >
> > >
> > >
> > > get
> > >
> > > {
> > >
> > > DesignerVerbCollection verbs = new DesignerVerbCollection();
> > >
> > > // Adding of custom verbs here....
> > >
> > > return verbs;
> > >
> > > }
>
> > > }
> > >
> > >
> > >
> > > public override void Initialize(IComponent component)
> > >
> > > {
> > >
> > >
> > >
> > > base.Initialize(component);
> > >
> > > // Be notified when properties are changed.
> > >
> > > IComponentChangeService service = GetService(
> > >
> > > typeof(IComponentChangeService)) as IComponentChangeService;
> > >
> > > if(service != null)
> > >
```

```
> >> > service.ComponentChanged +=
> >> >
> >> > new ComponentChangedEventHandler(OnComponentChanged);
> >> >
> >> > }
> >> >
> >> >
> >> >
> >> > #region Private Methods
> >> >
> >> > private void OnComponentChanged(object sender,
> >> > ComponentChangedEventArgs
> >> > e)
> >> >
> >> > {
> >> >
> >> > if(e.Component == this.Component && e.Member != null && e.Member.Name
> >> > ==
> >> > "Icon")
> >> >
> >> > {
> >> >
> >> > // This is where I want to change the verbs displayed.
> >> >
> >> >
> >> >
> >> > }
>
> >> > }
> >> >
> >> >
> >> >
> >> > }
> >> >
> >> >
> >> >
> >> >
> >> >
```