Is there an event that triggers if the number of ListViewItems in a ListView changes? (Windows Forms)

Asked 16 years, 3 months ago Modified 16 years, 3 months ago

Viewed 7k times

3

I'd like to enable/disable some other controls based on how many items are in my `ListView` control. I can't find any event that would do this, either on the `ListView` itself or on the `ListViewItemCollection`. Maybe there's a way to generically watch any collection in C# for changes?

I'd be happy with other events too, even ones that sometimes fire when the items don't change, but for example the `ControlAdded` and `Layout` events didn't work :(.

winforms **events** **listview**

Share

Improve this question

Follow

edited Sep 25, 2008 at 11:16

Matt Hamilton

204k ● 61 ● 392 ● 321

asked Sep 6, 2008 at 1:37

Domenic

113k ● 42 ● 226 ● 273

3 Answers

Sorted by: Highest score (default)

2

@Domenic

Not too sure, Never quite got that far in the thought process.

Another solution might be to extend ListView, and when adding and removing stuff, instead of calling .items.add, and items.remove, you call your other functions. It would still be possible to add and remove without events being raised, but with a little code review to make sure .items.add and .items.remove weren't called directly, it could work out quite well. Here's a little example. I only showed 1 Add function, but there are 6 you would have to implement, if you wanted to have use of all the available add functions. There's also .AddRange, and .Clear that you might want to take a look at.

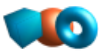
```
Public Class MonitoredListView
    Inherits ListView

    Public Event ItemAdded()
    Public Event ItemRemoved()

    Public Sub New()
        MyBase.New()
    End Sub

    Public Function AddItem(ByVal Text As String) As L
        RaiseEvent ItemAdded()

        MyBase.Items.Add(Text)
    End Function
```

```
    Public Sub RemoveItem(ByVal Item As ListViewItem)
        RaiseEvent ItemRemoved()

        MyBase.Items.Remove(Item)
    End Sub

End Class
```

Share Improve this answer

Follow

answered Sep 6, 2008 at 2:26



Kibbee

66.1k • 28 • 144 • 184

I would think it would be better to raise the event *after* making the change, instead of before. Or, you could have two events. `RaiseEvent ItemAdding();` for before and `RaiseEvent ItemAdded();` for after. – Jesse Chisholm Feb 5, 2015 at 18:40

0

I can't find any events that you could use. Perhaps you could subclass ListViewItemCollection, and raise your own event when something is added, with code similar to this.

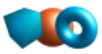
```
Public Class MyListViewItemCollection
    Inherits ListView.ListViewItemCollection

    Public Event ItemAdded(ByVal Item As ListViewItem)

    Sub New(ByVal owner As ListView)
        MyBase.New(owner)
    End Sub

    Public Overrides Function Add(ByVal value As
System.Windows.Forms.ListViewItem) As System.Windows.F
        Dim Item As ListViewItem
```

```
        Item = MyBase.Add(value)

        RaiseEvent ItemAdded(Item)

        Return Item
    End Function
End Class
```

Share Improve this answer

Follow

answered Sep 6, 2008 at 1:47



Kibbee

66.1k ● 28 ● 144 ● 184

0

I think the best thing that you can do here is to subclass ListView and provide the events that you want.

Share Improve this answer

Follow

answered Sep 6, 2008 at 2:17

jfs

16.7k ● 13 ● 63 ● 90