

The results are in! See what nearly 90,000 developers picked as their most loved, dreaded, and desired coding languages and more in the 2019 Developer Survey.

How can I add a css class to an updatepanel in ASP.Net?

[Ask Question](#)

▲ How can I add a css class to an updatepanel in the c# code behind file of asp.net

30



2

c#

asp.net

css

edited Jul 29 '09 at 19:17

[Gavin Miller](#)**34.3k** 18 106 169

asked Jul 29 '09 at 19:05

[ErnieStings](#)**2,803** 15 40 52

maybe you need to clarify your question a bit.... a (pure) class cannot be added to an updatepanel (control) you add controls to the updatepanel – [Jaime](#) Jul 29 '09 at 19:09

What do you mean by "class"? Css class? class that inherits from System.Web.UI.Control? type with some data you want to keep track of? – [Joel Coehoorn](#) Jul 29 '09 at 19:10

5 a css class. the update panel renders as a div so it should be able to be assigned a css class – [ErnieStings](#) Jul 29 '09 at 19:16

6 Answers

[Home](#)[PUBLIC](#)[Stack Overflow](#)[Tags](#)[Users](#)[Jobs](#)**Teams**

Q&A for work

[Learn More](#)

20



As you've seen the update panel doesn't have a css class property. So since it can't be done directly you need a work around; there are two (Grabbed from [UpdatePanel and CSS](#)) that can get the behavior you desire.

One is to surround the update panel with a div:

```
<div id="foo" style="visibility: hidden; position: absolute">
  <asp:UpdatePanel ID="UpdatePanel1" runat="server">
    </asp:UpdatePanel>
</div>
```

The other is to apply a css selector based on the update panel's id:

```
<style type="text/css">
#<%=UpdatePanel1.ClientID%> {
  visibility: hidden;
  position: absolute;
}
</style>
```

Yet another way not mentioned in the article is surround the panel in a div and style the update panel based on it rendering as a div:

```
<style type="text/css">
#foo div {
  visibility: hidden;
  position: absolute;
}
</style>

<div id="foo">
  <asp:UpdatePanel ID="UpdatePanel1" runat="server">
    </asp:UpdatePanel>
</div>
```

edited Jul 29 '09 at 19:23

answered Jul 29 '09 at 19:10



Gavin Miller

34.3k 18 106 169

CodeBehind:

```
2 UpdatePanel panel = new UpdatePanel();  
panel.Attributes.Add("class", "your-css-class");
```

HTML Result:

```
<div id="..." class="your-css-class"></div>
```

answered Aug 21 '17 at 14:21



Tristan

21 3

HTML

```
0 <asp:UpdatePanel ID="UpdatePanel1" runat="server"></asp:UpdatePanel>
```

CSS

```
<style type="text/css">  
    #UpdatePanel1 { position: relative; }  
</style>
```

answered Nov 13 '14 at 11:03



krizlx

4,669

14

34

45

Please explain the downvote. – [krizlx](#) Nov 11 '16 at 15:34

OP is asking for adding a class. Not for targeting by ID. – [Sebazzz](#) Mar 15 at 6:23

4

You could also do as I have and just create a new class that inherits the UpdatePanel. I got the basis for this somewhere else but I don't remember where so I can't credit fully but I only tweaked it for this idea. I'm about to do the same for HTML Attributes (since the .attributes() collection is for css on the UpdatePanel instead of raw HTML attributes as with most other web.ui.controls).

UPDATED: I've updated to include some other customization I've made that allow for ANY attribute to be added. Really this duplicates the first customization except that the 1st creates a fairly standard approach where this is completely flexible (thus not standard).

```
Imports System.ComponentModel
Imports System.Collections
Imports System.Web.UI
```

```
Namespace Controls
```

```
Public Class UpdatePanelCss
    Inherits UpdatePanel
    Private _cssClass As String
    Private _tag As HtmlTextWriterTag = HtmlTextWriterTag.Div
    Public HtmlAttributes As New HtmlAttributes

    <DefaultValue("")> _
    <Description("Applies a CSS style to the panel.")> _
    Public Property CssClass() As String
        Get
            Return If(_cssClass, [String].Empty)
        End Get
    End Property
End Class
```

```

        Set(ByVal value As String)
            _cssClass = value
        End Set
    End Property

    ' Hide the base class's RenderMode property since we don't u
    <Browsable(False)> _
    <EditorBrowsable(EditorBrowsableState.Never)> _
    <DesignerSerializationVisibility(DesignerSerializationVisibi
    Public Shadows Property RenderMode() As UpdatePanelRenderMod
    Get
        Return MyBase.RenderMode
    End Get
    Set(ByVal value As UpdatePanelRenderMode)
        MyBase.RenderMode = value
    End Set
    End Property

    <DefaultValue(HtmlTextWriterTag.Div)> _
    <Description("The tag to render for the panel.")> _
    Public Property Tag() As HtmlTextWriterTag
    Get
        Return _tag
    End Get
    Set(ByVal value As HtmlTextWriterTag)
        _tag = value
    End Set
    End Property

    Protected Overrides Sub RenderChildren(ByVal writer As HtmlT
    If IsInPartialRendering Then
        ' If the UpdatePanel is rendering in "partial" mode
        ' it's the top-level UpdatePanel in this part of the
        ' it doesn't render its outer tag. We just delegate
        ' class to do all the work.
        MyBase.RenderChildren(writer)
    Else
        ' If we're rendering in normal HTML mode we do all tl
        ' rendering. We then go render our children, which i
        ' normal control's behavior is.
        writer.AddAttribute(HtmlTextWriterAttribute.Id, Clie
        If CssClass.Length > 0 Then
            writer.AddAttribute(HtmlTextWriterAttribute.[Cla
        End If
        If HtmlAttributes.Count > 0 Then
            For Each ha As HtmlAttribute In HtmlAttributes
                writer.AddAttribute(ha.AttrName, ha.AttrVal)
            
```

```

        Next
    End If
    writer.RenderBeginTag(Tag)
    For Each child As Control In Controls
        child.RenderControl(writer)
    Next
    writer.RenderEndTag()
End If
End Sub

End Class

Public Class HtmlAttribute
    Private PAttrName As String
    Private PAttrVal As String

    Public Sub New(AttrName As String, AttrVal As String)
        PAttrName = AttrName
        PAttrVal = AttrVal
    End Sub

    Public Property AttrName() As String
        Get
            Return PAttrName
        End Get
        Set(value As String)
            PAttrName = value
        End Set
    End Property

    Public Property AttrVal() As String
        Get
            Return PAttrVal
        End Get
        Set(value As String)
            PAttrVal = value
        End Set
    End Property

End Class

Public Class HtmlAttributes
    Inherits CollectionBase

    Public ReadOnly Property Count() As Integer
        Get

```

```

        Return List.Count
    End Get
End Property

Default Public Property Item(ByVal index As Integer) As HtmlAttribute
    Get
        Return CType(List.Item(index), HtmlAttribute)
    End Get
    Set(ByVal Value As HtmlAttribute)
        List.Item(index) = Value
    End Set
End Property

Public Function Add(ByVal item As HtmlAttribute) As Integer
    Return List.Add(item)
End Function

Public Sub Remove(ByVal index As Integer)
    If index < List.Count AndAlso List.Item(index) IsNot Nothing Then
        List.RemoveAt(index)
    Else
        Throw New Exception(String.Concat("Index(", index.ToString(), ") is not a valid. List only has ", List.Count.ToString, " items."))
    End If
End Sub

Public Sub Remove(ByRef hAttribute As HtmlAttribute)
    If List.Count > 0 AndAlso List.IndexOf(hAttribute) >= 0 Then
        List.Remove(hAttribute)
    Else
        Throw New Exception("Object does not exist in collection")
    End If
End Sub

End Class

End Namespace

```

C# conversion via <http://www.developerfusion.com/>:

```

using Microsoft.VisualBasic;
using System;
using System.Collections;
using System.Collections.Generic;
using System.Data;

```

```

using System.Diagnostics;
using System.ComponentModel;
using System.Web.UI;

namespace Controls
{
    public class UpdatePanelCss : UpdatePanel
    {
        private string _cssClass;
        private HtmlTextWriterTag _tag = HtmlTextWriterTag.Div;

        public HtmlAttributes HtmlAttributes = new HtmlAttributes();
        [DefaultValue("")]
        [Description("Applies a CSS style to the panel.")]
        public string CssClass {
            get { return _cssClass ?? String.Empty; }
            set { _cssClass = value; }
        }

        // Hide the base class's RenderMode property since we don't want it
        [Browsable(false)]
        [EditorBrowsable(EditorBrowsableState.Never)]
        [DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)]
        public new UpdatePanelRenderMode RenderMode {
            get { return base.RenderMode; }
            set { base.RenderMode = value; }
        }

        [DefaultValue(HtmlTextWriterTag.Div)]
        [Description("The tag to render for the panel.")]
        public HtmlTextWriterTag Tag {
            get { return _tag; }
            set { _tag = value; }
        }

        protected override void RenderChildren(HtmlTextWriter writer)
        {
            if (IsInPartialRendering) {
                // If the UpdatePanel is rendering in "partial" mode
                // it's the top-level UpdatePanel in this part of the
                // it doesn't render its outer tag. We just delegate
                // class to do all the work.
                base.RenderChildren(writer);
            } else {
                // If we're rendering in normal HTML mode we do all the
                // rendering. We then go render our children, which
            }
        }
    }
}

```



```

        // normal control's behavior is.
        writer.AddAttribute(HtmlTextWriterAttribute.Id, ClientID);
        if (CssClass.Length > 0) {
            writer.AddAttribute(HtmlTextWriterAttribute.Class, CssClass);
        }
        if (HtmlAttributes.Count > 0) {
            foreach (HtmlAttribute ha in HtmlAttributes) {
                writer.AddAttribute(ha.AttrName, ha.AttrVal);
            }
        }
        writer.RenderBeginTag(Tag);
        foreach (Control child in Controls) {
            child.RenderControl(writer);
        }
        writer.RenderEndTag();
    }
}

public class HtmlAttribute
{
    private string PAttrName;

    private string PAttrVal;
    public HtmlAttribute(string AttrName, string AttrVal)
    {
        PAttrName = AttrName;
        PAttrVal = AttrVal;
    }

    public string AttrName {
        get { return PAttrName; }
        set { PAttrName = value; }
    }

    public string AttrVal {
        get { return PAttrVal; }
        set { PAttrVal = value; }
    }
}

public class HtmlAttributes : CollectionBase
{

```

```
public int Count {
    get { return List.Count; }
}

public HtmlAttribute this[int index] {
    get { return (HtmlAttribute)List[index]; }
    set { List[index] = value; }
}

public int Add(HtmlAttribute item)
{
    return List.Add(item);
}

public void Remove(int index)
{
    if (index < List.Count && List[index] != null) {
        List.RemoveAt(index);
    } else {
        throw new Exception(string.Concat("Index(", index.ToS
valid. List only has ", List.Count.ToString(), " items."));
    }
}

public void Remove(ref HtmlAttribute hAttribute)
{
    if (List.Count > 0 && List.IndexOf(hAttribute) >= 0) {
        List.Remove(hAttribute);
    } else {
        throw new Exception("Object does not exist in collec
    }
}

}
```

edited Mar 22 '13 at 20:08

answered Mar 22 '13 at 19:05



rainabba

2,191 20 23

1 Though this answer is useful, the question was asked with C# tag, not VB.NET. – [Ed Schwehm](#) Mar 22 '13 at 19:27

It's all good. I upvoted your answer, no need to get upset. – [Ed Schwehm](#) Mar 25 '13 at 13:30

Thank you Ed. Deleting my last. – [rainabba](#) Apr 18 '13 at 20:19

No worries. We're all friends here. – [Ed Schwehm](#) Apr 19 '13 at 13:01

you can use single class html attribute

20

```
<asp:UpdatePanel ID="UpdatePanel1" runat="server" class="MyCssClass"
</asp:UpdatePanel>
```

answered Aug 16 '11 at 21:33



[Milox](#)

4,604 4 21 34

3 This works in .NET 4, but the earlier .NET parser blows up when you try to do this with `System.Web.HttpParseException: Type 'System.Web.UI.UpdatePanel' does not have a public property named 'class'` – [bdukes](#) Jan 30 '12 at 16:45

9 This is because `UpdatePanel` doesn't implement [IAttributeAccessor](#) in .NET 3.5, but does in .NET 4 (so you can't event programmatically add the `class` via the `Attributes` property, because that was added in .NET 4, as well). – [bdukes](#) Jan 30 '12 at 17:44

This should be the accepted answer. – [Sebazzz](#) Mar 15 at 6:23

An update panel can render as a div or span (depending on mode).

4

Easiest way to achieve what you want is to wrap the UpdatePanel in a standard Panel:

```
<asp:Panel ID="Panel1" runat="Server">
  <asp:UpdatePanel ID="UpdatePanel1" runat="server">
    </asp:UpdatePanel>
  </asp:Panel>
```

The you can just do this in codebehind:

```
Panel1.CssClass = "myCssClass";
```

You could also use a div, like LFSR Consulting said, and add `runat="server"` and then change the class attribute. But Panel is a bit easier to work with (a Panel just renders as a div).

answered Jul 29 '09 at 19:31



Dan Diplo

22.3k 4 52 83