

UpdatePanel Control Overview :

Canceling an Asynchronous Postback

Web Developer :

Visual Basic & C#

Contact :

Par3eh@yahoo.com

Par3eh91@gmail.com

BY : K1

© All Right Reserved For Par3eh 2009-2010

Creating Script That Cancels Postbacks

You will start by creating ECMAScript (JavaScript) code that manages the asynchronous postback in the browser.

To create JavaScript code to cancel a postback

1. In the ASP.NET Web site, add a new JScript file and name it CancelPostback.js.
2. Add the following script to the file:

Visual Basic

```
var divElem = 'AlertDiv';
var messageElem = 'AlertMessage';

Sys.Application.add_load(ApplicationLoadHandler)
function ApplicationLoadHandler(sender, args)
{
    Sys.WebForms.PageRequestManager.getInstance().add_initializeRequest(CheckStatus);
}
function CheckStatus(sender, args)
{
    var prm = Sys.WebForms.PageRequestManager.getInstance();
    if (prm.get_isInAsyncPostBack() & args.get_postBackElement().id ==
'CancelRefresh') {
        prm.abortPostBack();
    }
    else if (prm.get_isInAsyncPostBack() & args.get_postBackElement().id ==
'RefreshButton') {
        args.set_cancel(true);
        ActivateAlertDiv('visible', 'Still working on previous request.');
```

C#

```
var divElem = 'AlertDiv';
var messageElem = 'AlertMessage';

Sys.Application.add_load(ApplicationLoadHandler)
function ApplicationLoadHandler(sender, args)
{
    Sys.WebForms.PageRequestManager.getInstance().add_initializeRequest(CheckStatus);
}
function CheckStatus(sender, args)
{
    var prm = Sys.WebForms.PageRequestManager.getInstance();
    if (prm.get_isInAsyncPostBack() & args.get_postBackElement().id ==
'CancelRefresh') {
        prm.abortPostBack();
```

```

    }
    else if (prm.get_isInAsyncPostBack() & args.get_postBackElement().id ==
'RefreshButton') {
        args.set_cancel(true);
        ActivateAlertDiv('visible', 'Still working on previous request.');
```

```

    }
    else if (!prm.get_isInAsyncPostBack() & args.get_postBackElement().id ==
'RefreshButton') {
        ActivateAlertDiv('visible', 'Retrieving headlines.');
```

```

    }
}
function ActivateAlertDiv(visString, msg)
{
    var adiv = $get(divElem);
    var aspan = $get(messageElem);
    adiv.style.visibility = visString;
    aspan.innerHTML = msg;
}
if(typeof(Sys) !== "undefined") Sys.Application.notifyScriptLoaded();

```

3. The script performs the following tasks:

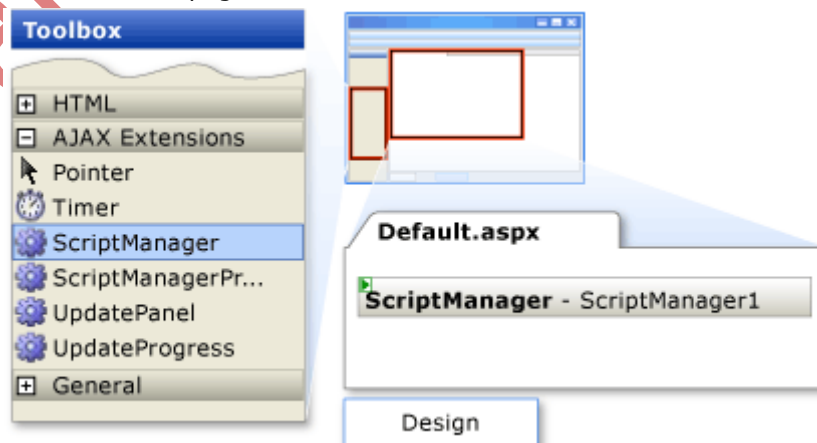
- Defines a handler for the load event of the Sys.Application class. This handler in turn registers a handler named `CheckStatus` for the `initializeRequest` event of the **PageRequestManager** class.
- Defines the `CheckStatus` handler to check whether an asynchronous postback is currently executing, and to determine the name of the element that caused the postback.
- Defines an `ActivateAlertDiv` function that toggles the visibility of a **<div>** element that is used to display messages.

Using the Script with an UpdatePanel Control

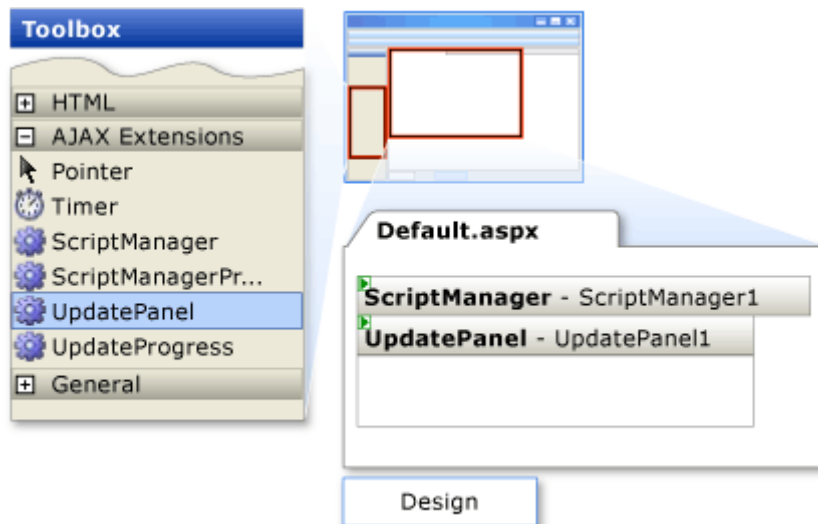
In this procedure you will use the script you created in a page that contains an UpdatePanel control. The script cancels the postback if the user clicks a link while the asynchronous postback is underway.

To create a page where users can cancel a postback

1. Create a new single-file ASP.NET Web page named `Default.aspx` and switch to Design view.
2. In the **AJAX Extensions** tab of the toolbox, double-click the `ScriptManager` control to add it to the page.



3. In the toolbox, double-click the `UpdatePanel` control to add it to the page.



4. Switch to Source view and add the following style rules to a **<style>** block in the **<head>** element of the page:

Visual Basic

```
<style type="text/css">
body {
    font-family: Tahoma;
}
#UpdatePanel1{
    width: 400px;
    height: 200px;
    border: solid 1px gray;
}
div.AlertStyle {
    font-size: smaller;
    background-color: #FFC080;
    width: 400px;
    height: 20px;
    visibility: hidden;
}
</style>
```

C#

```
<style type="text/css">
body {
    font-family: Tahoma;
}
#UpdatePanel1{
    width: 400px;
    height: 200px;
    border: solid 1px gray;
}
div.AlertStyle {
    font-size: smaller;
    background-color: #FFC080;
    width: 400px;
    height: 20px;
    visibility: hidden;
}
</style>
```


- Defines a **<div>** element that will be used to display a message during an asynchronous postback. The **<div>** element also contains a **LinkButton** control that enables the postback to be canceled.
8. In the **<script runat="server">** element, add the following server code as the Click event handler, which is for the **Refresh** button in the **UpdatePanel** control.

Visual Basic

```
Protected Sub NewsClick_Handler(ByVal sender As Object, ByVal e As EventArgs)
    System.Threading.Thread.Sleep(2000)
    HeadlineList.DataSource = GetHeadlines()
    HeadlineList.DataBind()
End Sub
```

C#

```
protected void NewsClick_Handler(object sender, EventArgs e)
{
    System.Threading.Thread.Sleep(2000);
    HeadlineList.DataSource = GetHeadlines();
    HeadlineList.DataBind();
}
```

9. The code uses data binding to read and display a list of headlines in the **DataList** control.

Note:

The handler for the **Click** event intentionally introduces a delay for this tutorial. In practice, you would not introduce a delay. Instead, the delay would be the result of server traffic or of server code that takes a long time to process, such as a long-running database query.

10. Inside the **<script>** element, add the following code for the page's **Load** event:

Visual Basic

```
Protected Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs)
    If Not (IsPostBack) Then
        HeadlineList.DataSource = GetHeadlines()
        HeadlineList.DataBind()
    End If
End Sub
```

C#

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        HeadlineList.DataSource = GetHeadlines();
        HeadlineList.DataBind();
    }
}
```

11. The code checks whether the current request is a postback. If the request is not a postback, the **DataList** control is bound to a list of headlines. (During asynchronous postbacks, data binding occurs in the **NewClick_Handler** method that you created in the previous step.)

12. Inside the **<script>** element, add the following code to generate headlines:

Visual Basic

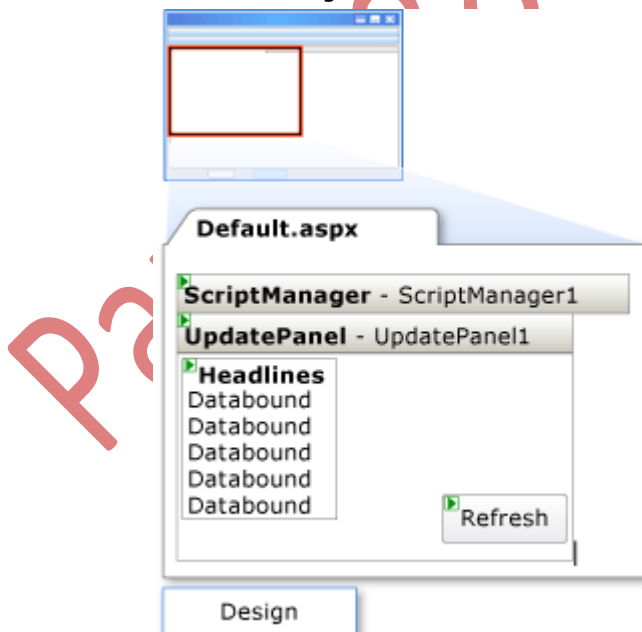
```
' Helper method to simulate news headline fetch.
Private Function GetHeadlines() As SortedList
    Dim headlines As New SortedList()
    headlines.Add(1, "This is headline 1.")
    headlines.Add(2, "This is headline 2.")
    headlines.Add(3, "This is headline 3.")
    headlines.Add(4, "This is headline 4.")
    headlines.Add(5, "This is headline 5.")
    headlines.Add(6, "(Last updated on " & DateTime.Now.ToString() & ")")
    Return headlines
End Function
```

C#

```
// Helper method to simulate news headline fetch.
private SortedList GetHeadlines()
{
    SortedList headlines = new SortedList();
    headlines.Add(1, "This is headline 1.");
    headlines.Add(2, "This is headline 2.");
    headlines.Add(3, "This is headline 3.");
    headlines.Add(4, "This is headline 4.");
    headlines.Add(5, "This is headline 5.");
    headlines.Add(6, "(Last updated on " + DateTime.Now.ToString() + ")");
    return headlines;
}
```

13. The headlines in this tutorial are created as a static list. In a real application they would be generated dynamically.

14. Switch to Design view and make sure that the page resembles the following figure.



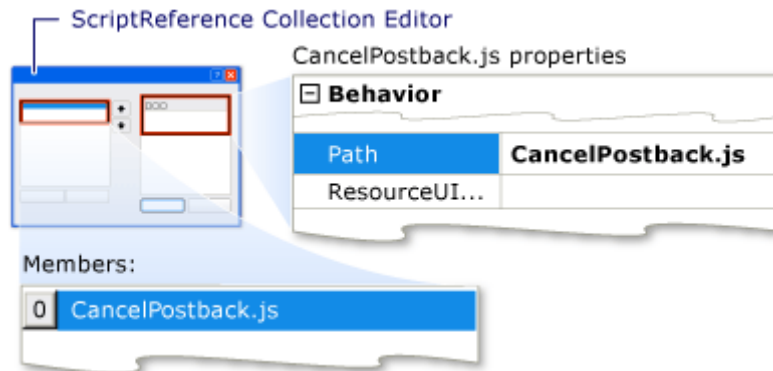
15. Select the ScriptManager control.

16. In the Properties window, select the **Scripts** property and click the ellipsis (...) button to display the **ScriptReference Collection Editor** dialog box.

17. Click **Add** to add a script reference.

- Set the **Path** property of the script reference to CancelPostBack.js, which is the JavaScript file that you created previously.

Adding a script reference by using the Scripts collection of the ScriptManager makes sure that the script is loaded after the Microsoft AJAX Library has loaded.



- Click **OK** to close the **ScriptReference Collection Editor** dialog box.
- Save your changes and press CTRL+F5 to view the page in a browser.
- Click the **Refresh** button and wait for the panel to refresh.

A message is displayed with an option to cancel the postback.
- Click the **Refresh** button again and after the message appears, click the **Refresh** button again and wait for the panel to refresh.

The text of the message changes to indicate that the previous refresh is still in progress. The second refresh is ignored.
- Click the **Refresh** button again, and when the message appears, click the **Cancel** link to cancel the postback.

This time, the time displayed in the UpdatePanel control does not change, because the asynchronous postback was canceled.

Visual Basic

```
<%@ Page Language="VB" %>
<%@ Import Namespace="System.Collections.Generic" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">
    Protected Sub NewsClick_Handler(ByVal sender As Object, ByVal e As EventArgs)
        System.Threading.Thread.Sleep(2000)
        HeadlineList.DataSource = GetHeadlines()
        HeadlineList.DataBind()
    End Sub
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As EventArgs)
        If Not (IsPostBack) Then
            HeadlineList.DataSource = GetHeadlines()
            HeadlineList.DataBind()
        End If
    End Sub
    ' Helper method to simulate news headline fetch.
    Private Function GetHeadlines() As SortedList
        Dim headlines As New SortedList()
        headlines.Add(1, "This is headline 1.")
        headlines.Add(2, "This is headline 2.")
        headlines.Add(3, "This is headline 3.")
        headlines.Add(4, "This is headline 4.")
        headlines.Add(5, "This is headline 5.")
        headlines.Add(6, "(Last updated on " & DateTime.Now.ToString() & ")")
        Return headlines
    End Function
</script>
```

