How to update page data after event handling?

Asked 15 years, 8 months ago Modified 15 years, 8 months ago

Viewed 5k times

3

On Page_Init I create a table of dynamically created controls based on a couple of database tables. One of the controls is an ImageButton for moving an list item up the list. What this event handler does is to update the SortOrder column in the database for the affected items.

Now the problem is that since the controls are created in the Page_Init event and the SortOrder is updated later on when the ImageButton command event is fired. What's the best procedure for updating the table with the correct SortOrder. If I recreate the table after the event has fired the ImageButton command event does not work any more.

- Should I implement a method for updating the data in the table without recreating it?
- Should I reload the page in code after the event has fired?

What's your preferred way for solving this problem?

asp.net event-handling page-lifecycle

Share

Improve this question

Follow

asked Apr 23, 2009 at 7:38

Joakim Karlsson

1,112 ● 1 ● 15 ● 28

3 Answers

Sorted by: Highest score (default)

8

Page events such as `Init` and `Load` will always fire before the event handler that raised the postback. This is the basis of the Page lifecycle (For a visual representation by Peter Bromberg, see here). Most developers new to ASP.NET have a major problem understanding and appropriately handling this "quandary".

The ideal way to do this is:

a. Your `Page_Init` should call a procedure (let's call it `BindData()` for illustration) that handles the creation of the table based on database data. This method would be similar to a binding method that binds to the database data and renders UI elements on the basis of that binding. IOW, you should remove the table creation code from the `Page_Init` method and put it in a separate method so that it can be called when needed.

Important note: This `BindData()` method also handles the attaching of the eventhandler for the dynamically created ImageButton control to the control. We'll call this

`ImageButton_Click`. This is crucial for the control to the event to fire on subsequent postback.

b. When your `ImageButton_Click` method executes, it calls the `BindData()` method to recreate the table and it's bindings but with new sort order rules.

So, the order of execution on first load is:

1. `Page_Init`
2. `BindData()`

The order of execution on subsequent loads (on postback) is:

1. `Page_Init`
2. `BindData()` - Eventhandler for ImageButton attached.
3. `ImageButton_Click`
4. `BindData()`

Share Improve this answer

Follow

answered Apr 23, 2009 at 8:09

Cerebrus

25.8k ● 8 ● 57 ● 71

4

You'll need something like this...

- OnInit (IsPostBack = false)
  - Dynamically create ImageButton

- Wireup ImageButton Event Handler
- Load Table - Check for a sort-order in Session/Variable. If none; use the default

Click the button

- OnInit (IsPostBack = true / 1st Postback)
  - Dynamically re-create ImageButton
  - Wireup ImageButton Event Handler
  - Load Table - with default sort order
- ImageButton_OnClick (Still the same 1st postback)
  - Reload Table - with specific sort order
  - Save this sort-order variable in Viewstate/Session variable

Cause some other Postback

- OnInit (IsPostBack = true / 2nd & Subsequent Postbacks)
  - Dynamically create ImageButton
  - Wireup ImageButton Event Handler
  - Load Table - Check for a sort-order in Session/Variable. If FOUND, use that.

Share Improve this answer Follow

answered Apr 23, 2009 at 8:08

Eoin Campbell
44.2k 18 105 160

What is it they say about "Great Minds" ;-) – Eoin Campbell Apr 23, 2009 at 8:19

1

Firstly, you seem to be binding your data manually to UI controls. In Asp.Net there and many ways to avoid this using built-in data binding techniques. Many controls like the GridView allow automatic creation of Html tables from a given data source. There are many other options including Repeaters.

However you do choose to bind your data, the technique is to rebind at some point every time through the page lifecycle.

You need to...

1. Bind you data on first page load with the default sort order
2. Rebind the data in the image button's event handler after the sort order has been changed.

The code would look something like this...

```
private void Page_Load (...)
{
    if (!IsPostBack)
        //On First Load
        BindData(defaultSoortOrder);
    else
        BindData(currentSortOrder);
}
```

```
private void ImageButton_Click (...)
{
    currentSortOrder = newSortOrder;
    BindData(currentSortOrder);
}
```

If the Image button is clicked, you will end up calling BindData twice. But this is necessary since a page postback could be initiated from any control, you need to always ensure you bind the data when the page loads.

Share Improve this answer Follow

edited Apr 23, 2009 at 8:55

answered Apr 23, 2009 at 8:49

Andy McCluggage
38.7k ● 18 ● 61 ● 71