# Handling Postbacks from A Popup Control Without an UpdatePanel

Christian Wenz

## Overview

The PopupControl extender in the AJAX Control Toolkit offers an easy way to trigger a popup when any other control is activated. When a postback occurs in such a panel and there are several panels on the page it is hard to determine which panel has been clicked.

## Steps

When using a **PopupControl** with a postback, but without having an **UpdatePanel** on the page, the Control Toolkit does not offer a way to determine which client element has triggered the popup which in turn caused the postback. However a small trick provides a workaround for this scenario.
First of all, here is the basic setup: two text boxes which both trigger the same popup, a calendar. Two **PopupControlExtenders** bring text boxes and popup together.

```
<form id="form1" runat="server">
  <asp:ScriptManager ID="asm" runat="server" />
  <div>
    Departure date: <asp:TextBox ID="tbDeparture" runat="server"
   />
    Return date: <asp:TextBox ID="tbReturn" runat="server" />
  </div>
  <asp:Panel ID="pnlCalendar" runat="server">
     <asp:Calendar ID="c1" runat="server"
   OnSelectionChanged="c1_SelectionChanged" />
  </asp:Panel>

  <ajaxToolkit:PopupControlExtender ID="pce1" runat="server"
    TargetControlID="tbDeparture" PopupControlID="pnlCalendar"
   Position="Bottom" />
  <ajaxToolkit:PopupControlExtender ID="pce2" runat="server"
    TargetControlID="tbReturn" PopupControlID="pnlCalendar"
   Position="Bottom" />
</form>
```

The basic idea is to add a hidden form field in the **<form>** element that holds the text box which launched the popup:

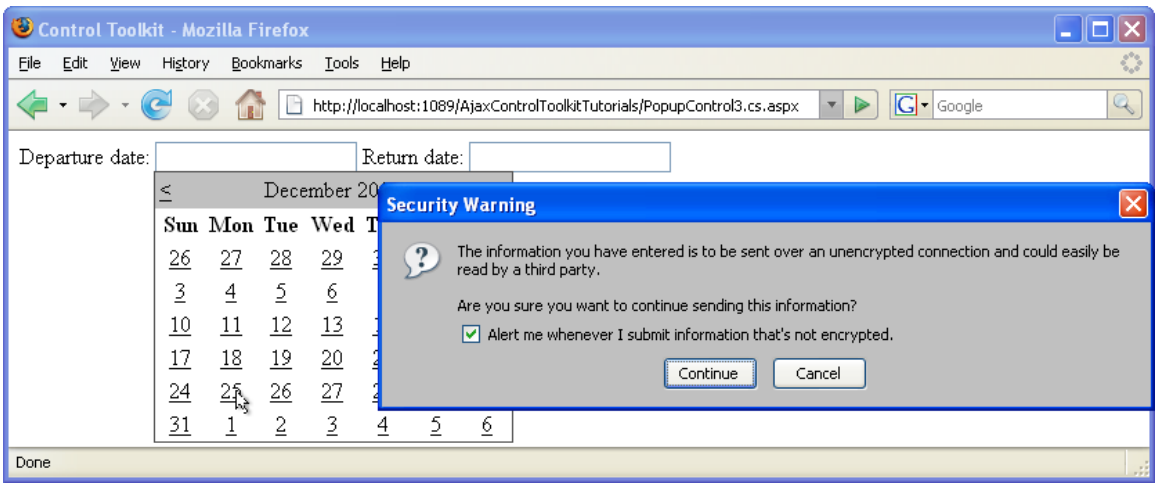
```
<input type="hidden" id="tbHidden" runat="server" />
```

When the page is loaded, JavaScript code adds an event handler to both text boxes: Whenever the user clicks on a text box, its name is written into the hidden form field:

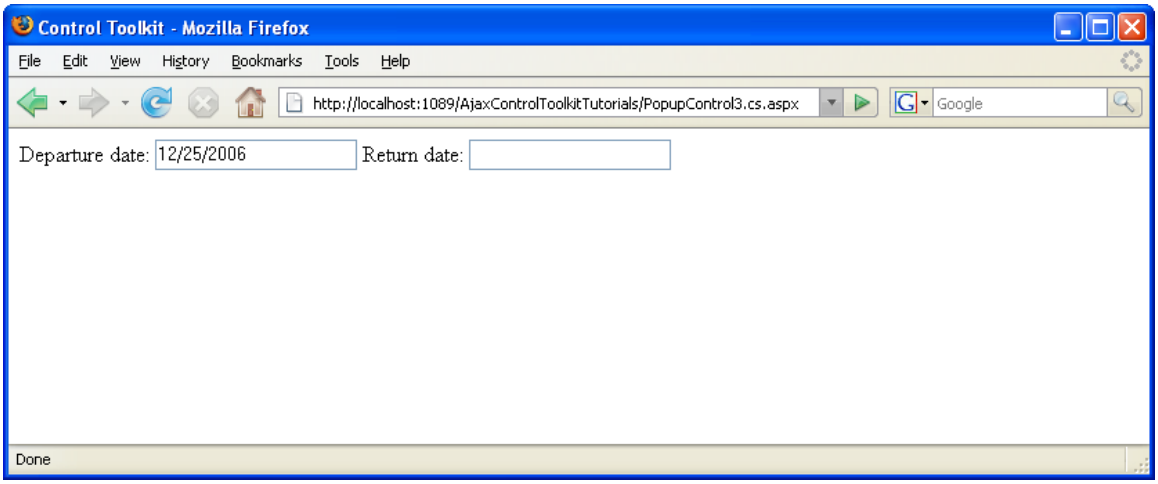
```
<script type="text/javascript">
  function pageLoad()
 {
    $get("tbDeparture").onclick = saveTextBox;
    $get("tbReturn").onclick = saveTextBox;
  }

  function saveTextBox()
 {
    $get("tbHidden").value = this.id;
  }
</script>
```

In the server-side code, the value of the hidden field must be read. Since hidden form fields are trivial to manipulate, a whitelist approach to validate the hidden value is required. Once the correct text box has been identified, the date from the calendar is written into it.

```
<script runat="server">
  Protected Sub c1_SelectionChanged(sender As object, e As
   EventArgs)
    Dim id As String = tbHidden.Value
    If (id = "tbDeparture" Or id = "tbReturn")
      Dim tb As TextBox = CType(FindControl(id), TextBox)
      tb.Text = CType(sender,
   Calendar).SelectedDate.ToShortDateString()
    End If
  End Sub
</script>
```




**The Calendar appears when the user clicks into the textbox**




**Clicking on a date puts it in the textbox**