Using DynamicPopulate with a User Control And JavaScript

Christian Wenz

Overview

The **DynamicPopulate** control in the ASP.NET AJAX Control Toolkit calls a web service (or page method) and fills the resulting value into a target control on the page, without a page refresh. It is also possible to trigger the population using custom client-side JavaScript code. However special care has to be taken when the extender resides in a user control.

Steps

First of all, you need an ASP.NET Web Service which implements the method to be called by the **DynamicPopulateExtender** control. The web service implements the method **getDate()** that expects one argument of type string, called **contextKey**, since the **DynamicPopulate** control sends one piece of context information with each web service call. Here is the code (file **DynamicPopulate.vb.asmx**) which retrieves the current date in one of three formats:

```
<%@ WebService Language="VB" Class="DynamicPopulate" %>

Imports System.Web
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.Web.Script.Services

<ScriptService()> _
Public Class DynamicPopulate
  Inherits System.Web.Services.WebService

  <WebMethod()> _
    Public Function getDate(ByVal contextKey As String) As String
    Dim myDate As String = ""
    Select Case contextKey
      Case "format1"
        myDate = String.Format("{0:MM}-{0:dd}-{0:yyyy}",
   DateTime.Now)
      Case "format2"
        myDate = String.Format("{0:dd}.{0:MM}.{0:yyyy}",
   DateTime.Now)
      Case "format3"
        myDate = String.Format("{0:yyyy}/{0:MM}/{0:dd}",
   DateTime.Now)
    End Select
    Return myDate
  End Function

End Class
```

In the next step, create a new user control (**.ascx** file), denoted by the following declaration in its first line:

```
<%@ Control Language="VB" ClassName="DynamicPopulate2" %>
```

A **<label>** element will be used to display the data coming from the server.

```
<label id="myDate" runat="server" />
```

Also in the user control file, we will use three radio buttons, each one representing one of the three possible date formats supported by the web service. When the user clicks on one of the radio buttons, the browser will execute JavaScript code which looks like this:

```
$find("mcd1_dpe1").populate(this.value)
```

This code accesses the **DynamicPopulateExtender** (do not worry about the strange ID yet, this will be covered later on) and triggers the dynamic population with data. In the context of the current radio button, **this.value** refers to its value which is **format1**, **format2** or **format3**—exactly what the web method expects.

The only thing missing in the user control yet is the **DynamicPopulateExtender** control which links the radio buttons to the web service.

```
<ajaxToolkit:DynamicPopulateExtender ID="dpe1" runat="server"
   ClearContentsDuringUpdate="true"
  TargetControlID="mcd1$myDate"
  ServicePath="DynamicPopulate.vb.asmx" ServiceMethod="getDate" />
```

Again you may note the strange ID used in the control: **mcd1$myDate** instead of **myDate**. Previously, the JavaScript code used **mcd1_dpe1** to access the **DynamicPopulateExtender** instead of **dpe1**.
This naming strategy is a special requirement when using **DynamicPopulateExtender** within a user control. Furthermore, you have to embed the user contol in a specific way to make it all work. Create a new ASP.NET page and register a tag prefix for the user control you have just implemented:

```
<%@ Register TagPrefix="uc1" TagName="myCustomDate"
   Src="~/DynamicPopulate2.vb.ascx" %>
```

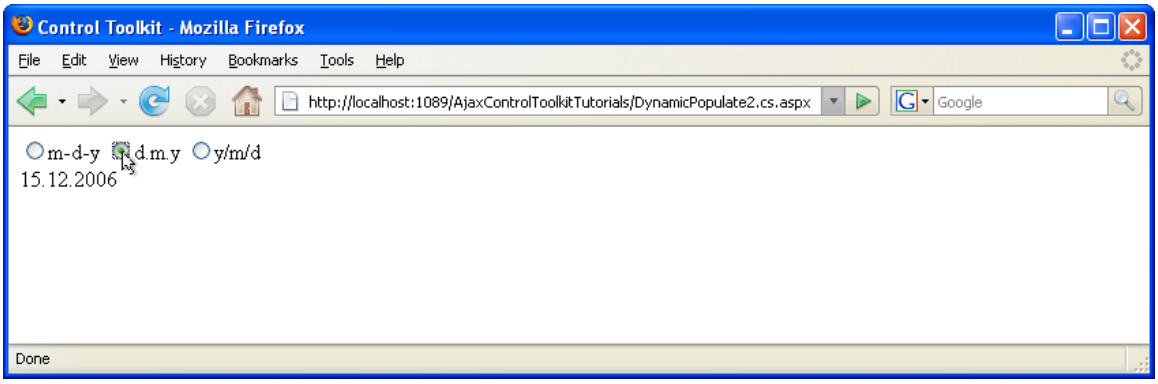
Then, include the ASP.NET AJAX **ScriptManager** control on the new page:

```
<asp:ScriptManager ID="asm" runat="server" />
```

Finally, add the user control to the page. You only have to set its **ID** attribute (and **runat="server"**, of course), but you also have to set it to a specific name: **mcd1**—since this is the prefix used within the user control to access it using JavaScript.

```
<div>
  <uc1:myCustomDate ID="mcd1" runat="server" />
</div>
```

And that's it! The page behaves as expected: A user clicks on on of the radio buttons, the control in the Toolkit calls the web service and displays the current date in the desired format.



The radio buttons reside in a user control