Dynamically Populating a Control

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. This tutorial shows how to set this up.

Steps

First of all, you need an ASP.NET Web Service which implements the method to be called by **DynamicPopulate**. The web service class requires the **ScriptService** attribute which is defined within **Microsoft.Web.Script.Services**; otherwise ASP.NET AJAX cannot create the client-side JavaScript proxy for the web service which in turn is required by **DynamicPopulate**.

The web method must expect one argument of type string, called **contextKey**, since the **DynamicPopulate** control sends one piece of context information with each web service call. The following web service returns the current date in a format represented by the **contextKey** argument:

```
<%@ 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
```

The web service is then saved as **DynamicPopulate.vb.asmx**. Alternatively, you could implement the **getDate()** method as a page method within the actual ASP.NET page with the **DynamicPopulate** control.

In the next step, create a new ASP.NET file. As always, the first step is to include the **ScriptManager** in the current page to load the ASP.NET AJAX library and to make the Control Toolkit work:

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

Then, add a label control (for instance using the HTML control of the same name, or the **<asp:Label />** web control) which will later show the result of the web service call.

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

An HTML button (as an HTML control, since we do not require a postback to the server) will then be used to trigger the dynamic population:

```
<input type="button" id="Button1" runat="server" value="Load date
   (m-d-y)" />
```

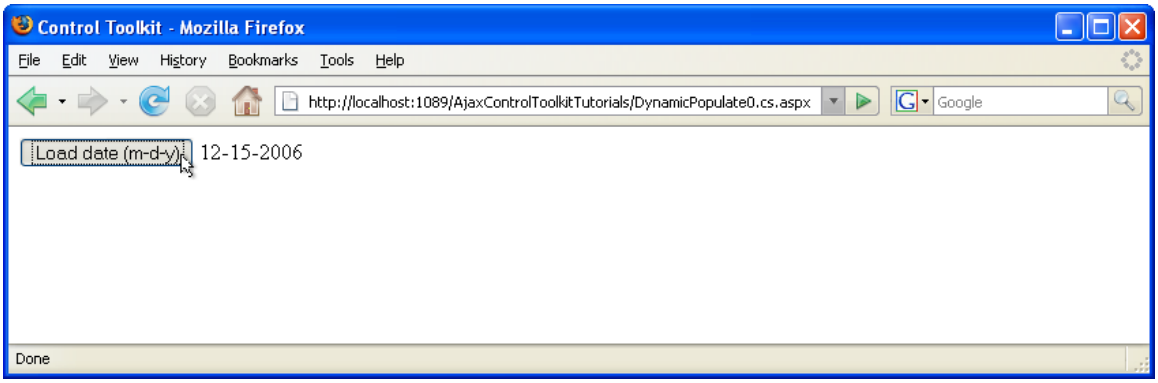
Finally, we need the **DynamicPopulateExtender** control to wire things up. The following attributes will be set (apart from the obvious ones, **ID** and **runat="server"**):

- **TargetControlID**—where to put the result from the web service call
- **ServicePath**—path to the web service (omit if you want to use a page method)
- **ServiceMethod**—name of the web method or page method
- **ContextKey**—context information to be sent to the web service
- **PopulateTriggerControlID**—element which triggers the web service call
- **ClearContentsDuringUpdate**—whether to empty the target element during the web service call

As you can see, the control requires some information but putting everything into place is quite straight-forward. Here is the markup for the **DynamicPopulateExtender** control in the current scenario:

```
<ajaxToolkit:DynamicPopulateExtender ID="dpe1" runat="server"
   ClearContentsDuringUpdate="true"
  TargetControlID="myDate" ServicePath="DynamicPopulate.vb.asmx"
   ServiceMethod="getDate"
  ContextKey="format1" PopulateTriggerControlID="Button1" />
```

Run the ASP.NET page in the browser and click on the button; you will receive the current date in month-day-year format.



A click on the button retrieves the date from the server