Ben Bastiaensen is leerkracht informatica aan KTA de Merodelei Turnhout

SiteMapProvider uitbreiden

BREID DE NAVIGATION CONTROLS UIT

De SiteMapProvider in ASP.NET levert een krachtige manier om navigatie in een website te bouwen. Maar voor een rijkere en uitgebreidere navigatie is de provider net iets te statisch. De auteur laat in dit artikel zien hoe je zelf deze SiteMapProvider kunt uitbreiden en aan de navigation controls kunt koppelen.

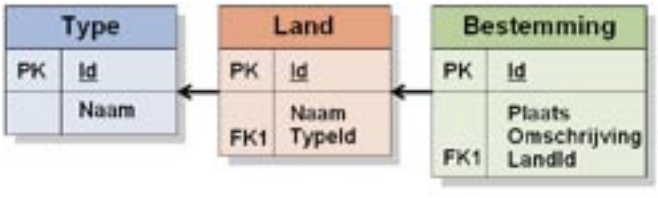
Navigatie is een belangrijk onderdeel van elke website. Er wordt dan ook veel aandacht aan besteed. Met de release van ASP.NET 2.0 werd het leven van de ontwikkelaar heel wat aangenamer met de navigation controls. Mooie, vlotte navigatie out-of-the-box. En voor een website waar de navigatie nooit of zelden wijzigt, zijn deze controls perfect. Maar zodra navigatie afhankelijk wordt van keuzes van een gebruiker is het een ander verhaal. Je bent dan genoodzaakt om toch je custom navigatie te ontwikkelen en af te stappen van de navigation controls. Maar het hoeft niet. Je kunt perfect een eigen SiteMapProvider bouwen die jouw navigatie gaat bepalen en deze dan koppelen aan de Menu-, TreeView- en SiteMapPath-control. Om dit te illustreren doen we het volgende: we ontwikkelen een fictieve website voor een reisbureau. Dit reisbureau biedt verschillende typen reizen aan. Afhankelijk van het type zal de inhoud van de TreeView-navigatielinks op de site verschillend zijn.

Databank

De gegevens die in de navigatie getoond worden, zijn opgeslagen in een databank. Er wordt een SQL Server-databank gebruikt. De structuur van de databank kun je in afbeelding 2 terugvinden.

Om met deze databank te kunnen werken, gebruiken we de web.config-connectionstring uit codevoorbeeld 1.
Deze connectionstring ziet er uit als iedere andere default connectionstring, met uitzondering van de optie MultipleActiveResultSets. Deze waarde moet op True gezet worden aangezien we, zoals je in codevoorbeeld 5 kunt zien, verscheidene geneste resultsets van een DataReader gaan gebruiken. Bij SQL Server 2005 staat deze mogelijkheid default uit. In eerdere versies of in MS Access staat deze mogelijkheid standaard aan en is de expliciete vermelding niet nodig.

Afbeelding 1. Overzicht van de webtoepassing



Afbeelding 2. Structuur van de 'Loco'-databank

SiteMapProvider aanmaken

Een eerste stap die we in code gaan uitvoeren, is het aanmaken van de custom SiteMapProvider; zie codevoorbeeld 2. Hiervoor hebben we een class nodig. Die laten we overerven van StaticSiteMapProvider. Als je dit uitvoert, merk je onmiddellijk dat er twee verplichte overloads in deze klasse zitten, GetRootNodeCore() en BuildSiteMap(). Uiteraard gaan we dadelijk aan deze methoden een custom invulling geven.

Uitbouw van de SiteMapProvider

Om deze klasse verder uit te bouwen, moeten we een aantal namespaces importeren:

- Imports Microsoft.VisualBasic
- Imports System.Web
- Imports System.Data
- Imports System.Data.SqlClient
- Imports System.Configuration
- Imports System.Collections.Specialized

Een aantal van deze namespaces is misschien niet expliciet nodig, omdat ze bij de declaratie van de methode eveneens opgenomen kunnen worden. Toch is het een goede gewoonte om de gebruikte namespaces te verzamelen aan het begin van de code. In dit voorbeeld hebben we bij de uitbouw van de SiteMapProvider één root SiteMapNode nodig. Binnen deze root node worden alle verdere nodes gekoppeld. De functies GetRootNodeCore() en BuildSite-

```
<connectionStrings>
  <add name="loco" connectionString="Data Source=localhost;
       Initial Catalog=Loco;
       Integrated Security=True;
       MultipleActiveResultSets=True"
    providerName="System.Data.SqlClient" />
</connectionStrings>
```

Codevoorbeeld 1. MultipleActiveResultSets is mogelijk

```
Public Class LocoSiteMapProvider
  Inherits StaticSiteMapProvider
  ' Code implementatie hier.
End Class
```

Codevoorbeeld 2. Lege Custom SiteMapProvider-klasse

```
Shared ReadOnly foutboodschap As String = "Probleem met connecti
onstring"
Dim Connection As String
Dim Root As SiteMapNode
Dim TypeId As Integer

Public Overrides Sub Initialize(ByVal name As String, _
       ByVal attributes As NameValueCollection)
   MyBase.Initialize(name, attributes)

   Connection = attributes("connectionStringName")

   If String.IsNullOrEmpty(Connection) Then
      Throw New ConfigurationErrorsException(foutboodschap)
   End If
End Sub
```

Codevoorbeeld 3. De methode Initialize

Map() zijn functies die als waarde een SiteMapNode hebben, namelijk de root node Maar voordat we deze uitbouwen, werken we eerst de procedure Initialize uit. Deze procedure hanteren we om de attributen die we via web.config doorgeven (zie codevoorbeeld 6) te gebruiken. In dit voorbeeld is dat er maar één, namelijk de connectionString. Codevoorbeeld 3 laat zien hoe we dit doen. Er worden hier ook de variabelen gedeclareerd die nodig zijn voor de uitwerking.

De belangrijkste methode in onze custom SiteMapProvider is de methode BuildSiteMap(). Deze gaat uiteindelijk de volledige sitemapstructuur genereren. Dat betekent dus dat we op dit moment ook moeten bepalen voor welk reistype we een sitemapstructuur willen bouwen. Dit zal in twee stappen verlopen. Ten eerste bepalen we in de code van de Masterpage welk type er geselecteerd is. Dit kunnen we terugvinden in de url. Het type wordt dan in een sessievariabele bewaard. Dit is noodzakelijk omdat we vanuit de klasse LocoSiteMapProvider geen notie hebben van het Requestobject. We kunnen met andere woorden op dat moment niets uit de querystring halen en zijn dus genoodzaakt dit op een ander moment te doen. Codevoorbeeld 4 illustreert dit.

De tweede stap doen we in de SiteMapProvider zelf. Zoals je in codevoorbeeld 5 ziet, kunnen we vanuit een klasse een Sessievariabele aanspreken via de HttpContext-klasse. Op die manier halen we de waarde van 'Type' reis op en kunnen we onze SiteMap verder opbouwen. Zoals je ziet, starten we met een dubbele controle. We controleren of de root node al bestaat en of het reeds gekende type reis (TypeId) hetzelfde is als het gevraagde type. Als dat het geval is, hoeft de SiteMap niet meer gegenereerd te worden, omdat we ze al hebben. In de andere gevallen wordt de SiteMap leeg gemaakt en wordt een nieuwe gebouwd.

Eerst wordt de root node opgebouwd. Daarna worden alle landen bij het gezochte type opgehaald. Binnen elk land worden dan de bestemmingen opgehaald. Voor elke entry wordt een SiteMapNode-object gebouwd dat via AddNode() wordt toegevoegd aan de juiste hoger gelegen node, zodat we een nette hiërarchische structuur krijgen. De methode GetRootNodeCore() moet ook geïmplementeerd worden. Maar deze doet in dit voorbeeld niet meer dan BuildSiteMap() oproepen en vervolgens de SiteMapNode 'Root' teruggeven; zie codevoorbeeld 6.

Toon de gegenereerde SiteMap

Nu moeten we er nog voor zorgen dat we de net ontwikkelde SiteMapProvider toevoegen aan de providerscollectie. We stellen deze eveneens in als de defaultprovider voor sitemaps, zodat de Menu-,

```
Protected Sub Page_Init(ByVal sender As Object, _
       ByVal e As System.EventArgs) Handles Me.Init
   Session("Type") = Request.QueryString("type")
End Sub
```

Codevoorbeeld 4. Bepalen 'Type' reis – deel 1

```
Public Overrides Function BuildSiteMap() As SiteMapNode
   ' Geen nodes bouwen indien deze al gemaakt zijn.
   If Root IsNot Nothing AndAlso _
      TypeId = HttpContext.Current.Session("Type") Then
      Return Root
   End If

   ' Verwijderen van de Nodes in het geval van een nieuw Type
   Me.Clear()

   ' Ontwikkelen van de root node
   Root = New SiteMapNode(Me, "Default.aspx", "Default.aspx", "Home")
   Me.AddNode(Root, Nothing)

   Dim objCn As New SqlConnection(ConfigurationManager. _
        ConnectionStrings(Connection).ConnectionString)
   Try
      'objcn
      objCn.Open()
      TypeId = HttpContext.Current.Session("Type")

      Dim strSqlLand As String = "SELECT * FROM Land " _
               & "WHERE TypeId = @TypeId " _
               & "ORDER BY Naam ASC;"

      Dim objCmdLand As New SqlCommand(strSqlLand, objCn)
      objCmdLand.Parameters.Clear()
      objCmdLand.Parameters.Add("@TypeId", SqlDbType.Int)
      objCmdLand.Parameters("@TypeId").Value = TypeId

      Dim objReaLand As SqlDataReader = objCmdLand.ExecuteReader

      Dim url As String

      While objReaLand.Read
         Dim LandId As Integer = objReaLand.Item("Id").ToString
         url = "Reizen.aspx?type=" & TypeId.ToString & "&land=" & LandId
         Dim LandNode As New SiteMapNode(Me, url, url, _
            objReaLand.Item("Naam").ToString)
         AddNode(LandNode, Root)

         ' Subtitels ophalen voor de huidige subcategorie
         Dim strSqlBestemming = "SELECT * FROM Bestemming " _
              & "WHERE LandId = @LandId " _
              & "ORDER BY Plaats ASC;"

         Dim objCmdBestemming As New SqlCommand(strSqlBestemming, objCn)
         objCmdBestemming.Parameters.Clear()
         objCmdBestemming.Parameters.Add("@LandId", SqlDbType.Int)
         objCmdBestemming.Parameters("@LandId").Value = LandId

         Dim objReaBestemming As SqlDataReader = _
                  objCmdBestemming.ExecuteReader

         While objReaBestemming.Read
            Dim BestemmingId As Integer = _
                     objReaBestemming.Item("Id").ToString
            url = "Bestemmingen.aspx?type=" & TypeId.ToString _
                     & "&bestemming=" & BestemmingId
            Dim BestemmingNode As New SiteMapNode(Me, url, url, _
                     objReaBestemming.Item("Plaats").ToString)
            AddNode(BestemmingNode, LandNode)
         End While
      End While
   Catch ex As Exception
      Return New SiteMapNode(Me, "Default.aspx", "Default.aspx", "Fout")
      Exit Function
   Finally
      objCn.Close()
   End Try
   Return Root
End Function
```

Codevoorbeeld 5. BuidSiteMap()

```
Protected Overrides Function GetRootNodeCore() As SiteMapNode
   BuildSiteMap()
   Return Root
End Function
```
Codevoorbeeld 6. GetRootNodeCore()

```
<siteMap defaultProvider="LocoSiteMapProvider" enabled="true">
   <providers>
      <add name="LocoSiteMapProvider"
           type="LocoSiteMapProvider"
           connectionStringName="loco"/>
      </providers>
</siteMap>
```
Codevoorbeeld 7. GetRootNodeCore()

TreeView- en SiteMapPath-controls deze kunnen gebruiken. Hiervoor moeten we in het System.Web-element van de web.config zijn; zie codevoorbeeld 7.

Ten slotte voegen we een SiteMapDataSource toe. Daarvan zetten we de eigenschap ShowStartingNode op False, zodat de root node niet getoond wordt. We plaatsen een TreeView-control en koppelen deze aan de SiteMapDataSource. En dan krijgen we uiteindelijk het resultaat zoals in afbeelding 1.

Tot slot

Zoals je uit dit simpele voorbeeld hebt gemerkt, zijn de verschillende providers die in ASP.NET 2.0 zijn ingebouwd niet alleen zeer krachtig op zich. Hun relatief eenvoudige uitbreidbaarheid maakt dat je ze in bijna alle mogelijke situaties kunt gebruiken. Hierdoor blijven ook de controls die met deze providers samenwerken in alle omstandigheden bruikbaar.

Ben Bastiaensen is leerkracht informatica aan KTA de Merodelei Turnhout en zelfstandig consultant webontwikkeling bij Edoceo. (www.edoceo.be) Je kunt hem bereiken op ben.bastiaensen@edoceo.be .

Referenties

ASP.NET 2.0 Providers: http://msdn2.microsoft.com/en-us/library/aa478948.aspx
StaticSiteMapProvider: http://msdn2.microsoft.com/en-us/library/system.web.staticsitemapprovider.aspx
Jeff Prosise over SiteMapProviders (in c#): http://msdn.microsoft.com/msdnmag/issues/05/06/WickedCode/