Peter Windemuller is werkzaam als ontwikkelaar bij DiVa-Digital Value

'Pimp your site'

HOE EENVOUDIG EN DOELTREFFEND AJAX KAN ZIJN

Iedere eindgebruiker wil zo snel mogelijk zijn relevante informatie op het scherm en heeft een hekel aan lang wachten na elke muisklik. In het modemtijdperk werd dit probleem nog opgelost door met behulp van frames het dataverkeer zoveel mogelijk te beperken. Maar met de huidige generatie browsers is er een nieuwe technologie beschikbaar die nog veel meer te bieden heeft. In dit artikel bespreken we een case waarin we Ajax-technologie hebben toegepast in een bestaande .NET 1.1-webapplicatie.

Omdat we het wiel niet opnieuw willen uitvinden, zijn we op zoek gegaan naar een framework dat ons op weg hielp in de wereld van Ajax. Aan de hand van een aantal criteria zijn verschillende Ajax-frameworks beoordeeld voor het gebruik in een .NET-applicatie. Vervolgens is het gekozen framework toegepast in een bestaande applicatie en laten we daar enkele voorbeelden van zien. Voor een opdrachtgever heeft DiVa een applicatie ontwikkeld die de bedrijfsprocessen van de organisatie volledig ondersteunt door een volledige web-based omgeving. De applicatie wordt gebruikt voor het onderhouden van woninggegevens, relaties en dossiers. De omgeving is ontwikkeld in ASP.NET 1.1, C# en Microsoft SharePoint. De applicatie is gebouwd op een eigen business- en datalaag. Om van het framework gebruik te maken, worden in de applicatie standaard custom-controls gebruikt die van de standaard ASP.NET-controls zijn afgeleid. De applicatie bestaat uit ongeveer 800 schermen. De applicatie bevat veel formulieren, waarbij verschillende onderdelen postbacks veroorzaken om extra gegevens op te halen of om de pagina aan te passen aan eerder gekozen waarden. Hierdoor moet de pagina vaak herladen worden, wat door gebruikers als vertragend dan wel als irritant werd bevonden. Het doel was, met zo min mogelijk programmeerwerk, het aantal postbacks te reduceren, waardoor de productiviteit en de gebruikerservaring werd verhoogd. Als oplossing hiervoor is gekeken of de bestaande applicatie met behulp van Ajax-technologie kon worden verbeterd.

Ajax en .NET Framework 1.1

Om te bepalen of een Ajax-framework geschikt was, zijn verschillende bestaande frameworks op een aantal criteria beoordeeld. Dit waren onder meer:

- compatibiliteit met ASP.NET 1.1
- het in gebruik kunnen nemen met minimale aanpassing aan bestaande configuratie en programmacode
- toekomstvastheid en ondersteuning vanuit de markt

Bij deze evaluatie zijn de volgende frameworks onderzocht:

1. **Atlas** - Dit is het standaard framework van Microsoft voor ASP.NET 2.0. Hoewel Atlas ook werkt in ASP.NET 1.1, is de functionaliteit hiervan beperkt tot het gebruik van de client-script libraries. Voor het gebruik van servercontrols is ASP.NET 2.0 vereist. Na de bètaperiode is de officiële naam veranderd in ASP.NET Ajax.
2. **AJAX.NET** - Dit is één van de bekendste Ajax-frameworks voor ASP.NET 1.1. Om AJAX.NET werkend te krijgen, dient de web.config te worden aangepast. De serverside ASP.NET-methode die door het Ajax-framework wordt aangeroepen, moet voorzien worden met het Ajax Method-attribuut en de class dient in het AJAX.NET-framework geregistreerd te worden. Ook dient er clientside-JavaScript gemaakt te worden om de serverside-methoden aan te roepen.

Framework	ATLAS	AJAX.Net	MagixAJAX.NET	Anthem.Net
Maker	Microsoft	Michael Schwarz	Open Source	Jason Diamond
ASP. Net 1.0	Nee, alleen client javascript	Ja	Ja	Ja
ASP. Net 2.0	Ja	Ja	Ja	Ja
Basisconcept	Verscheidene concepten beschikbaar. Men kan gebruik maken van directe Ajax-aanroepen, het declareren van Ajax-componenten en er zijn extra servercontrols	Framework voor de communicatie tussen client en serverobjecten.	Gebruik van een 'MagixPanel' voor Ajax-updates	Vervanging van de standaard webcontrols door Ajax-enabled controls
Servercontrols	Update panel, Autocomplete, DragOverlay, UpdateProgress, ControlEventTrigger en TimerControl	Geen		Vervanging voor de meeste standaard webcontrols
Configuratie	Geen, wanneer de VS template gebruikt wordt.	Web.config	Web.config	Geen

Tabel 1. Vergelijkingsmatrix

3. **MagicAJAX.NET** - Dit is een open source framework dat zowel ASP.NET 1.1 als ASP.NET 2.0 ondersteunt. MagicAJAX.NET is een panel dat postbacks van de controls op het panel afvangt en omzet in Ajax-callbacks. Voor MagicAJAX.NET moet de web.config geconfigureerd worden. Alle controls die 'Ajax enabled' dienen te worden, moeten in een MagicAJAX.NET-panel geplaatst worden. Het idee van MagicAJAX.NET is ook in Anthem.NET en Atlas terug te vinden.
4. **Anthem.NET** - Dit is een cross-browser Ajax-framework, gebaseerd op open source, dat zowel ASP.NET 1.1 als ASP.NET 2.0 ondersteunt. Voor Anthem.NET hoeft de web.config niet geconfigureerd te worden. Het idee van Anthem.NET is een set van controls die de bestaande ASP.NET-controls uitbreidt met Ajax-functionaliteit. Bijna alle ASP.NET-controls hebben een Anthem-equivalent. De bedoeling is dat de programmeur verder op dezelfde wijze met de Anthem-controls kan werken als met de ASP.NET-controls.

Tabel 1 bevat de geëvalueerde frameworks met daarbij een aantal specifieke eigenschappen. Op grond van de bevindingen is in dit specifieke geval gekozen voor Anthem.NET, omdat dit framework het beste aan de gestelde criteria voldeed. Het werkt in ASP.Net 1.1 en kan eenvoudig worden geïmplementeerd, doordat onze webapplicatie al custom controls bevat die de standaard webcontrolset van Microsoft vervangt. Daarom hoeven we alleen onze custom controls te laten overerven van de Anthem.Net-controls.

Anthem.NET in de praktijk

Om het Anthem.NET-framework te kunnen gebruiken, moet de Anthem.NET-dll in de betreffende VS.NET-projecten gereferenced worden. Hierna kunnen de Anthem-controls direct worden gebruikt. In codevoorbeeld 1 hebben we een eenvoudige applicatie geschreven die na het intypen van tekst in de TextBox, de tekst naar het Label kopieert met behulp van een Ajax-callback. Om dit te kunnen realiseren hoef je niet meer code schrijven dan bij een normale PostBack! Zoals is te zien, is de Anthem-functionaliteit al beschikbaar zonder daarvoor enige code te hoeven schrijven. De controls die in de applicatie zijn gebruikt, zijn custom controls die van de standaard ASP.NET-controls zijn afgeleid. Om het Anthem.NET-framework in de applicatie te gebruiken, zijn de meeste custom controls afgeleid van de Anthem-controls in plaats van de standaard webcontrols.

In Visual Studio kun je de toolbox gebruiken om Anthem.NET-controls op de webpagina's en user-controls te slepen naar waar je ze wilt gebruiken. Nadat een radiobuttonlist op een webpagina wordt gesleept, is te zien dat de control er ten opzichte van een gewone ASP.NET-control enkele eigenschappen bij heeft gekregen onder de categorie Misc; zie afbeelding 1. Deze properties kunnen natuurlijk ook in de code gezet worden. Een aantal belangrijke properties hierbij zijn:

1. **AutoCallBack** - Werkt synoniem aan AutoPostback. Er wordt alleen een Ajax-callback in plaats van een postback gedaan.
2. **AutoUpdateAfterCallBack** - Deze Boolean geeft aan of de html van de control bij een callback moet worden ververst.
3. **EnableCallBack** - Of de control een callback kan maken. Indien deze op 'false' staat, wordt er een postback gemaakt.
4. **EnableDuringCallBack** - Deze property geeft aan of de control enabled is tijdens de callback. Dit kan handig zijn als na de callback de control gewijzigd wordt of om dubbelclicks en dergelijke te voorkomen.
5. **TextDuringCallBack** - Hier kan een tekst ingevuld worden (bijvoorbeeld loading…) die tijdens de callback wordt getoond. De control bepaalt zelf waar de melding getoond wordt, wat voor bijvoorbeeld de datagrid voor een verrassing kan zorgen.

Voor en na de Ajax-callback kun je Javascript-functies aanroepen. Deze worden gedefinieerd bij PreCallBackFunction en PostCallBackFunction. Als door de PreCallBackFunction een false wordt teruggegeven, wordt de callback niet uitgevoerd. Wel wordt de Javascript-functie uitgevoerd die bij de CallBackCancelFunction staat gedefinieerd (indien ingevuld). Met behulp van de PreCallBackFunction kan bijvoorbeeld eenvoudig een JavaScript-bevestigingscherm worden getoond. Bij het gebruik van een Anthem-control hoef je niet alles met behulp van Ajax-callbacks te doen. Wanneer je bij een bepaalde Anthem.button-control bijvoorbeeld toch een postback wilt doen, zet je de enablecallback property op false. Hierdoor voert de control een gewone postback uit. Een groot voordeel van Anthem.Net is dat het mogelijk is om per control aan te geven of deze moet worden vernieuwd na een callback. Dit kan van tevoren worden ingesteld in de designer, maar je kunt ook in de code de property UpdateAfterCallback op 'true' te zetten. Op deze manier kan de verbruikte bandbreedte worden geoptimaliseerd.

AnthemPanel en AnthemPlaceHolder

Van de panel-control en placeholder-control zijn ook Anthem.NET-versies beschikbaar. Deze hebben als bijzondere eigenschap dat deze ook niet-Anthem-controls die zich op het panel of placeholder bevinden, kunnen updaten. Ook kan er met behulp van de property AddCallBacks, de postback van bijvoorbeeld een ASP.NET-buttoncontrol omgevormd worden tot een callback. Het updaten van controls in een speciaal panel vind je ook in andere Ajax-frameworks terug (Atlas, magicajax).

TimerControl

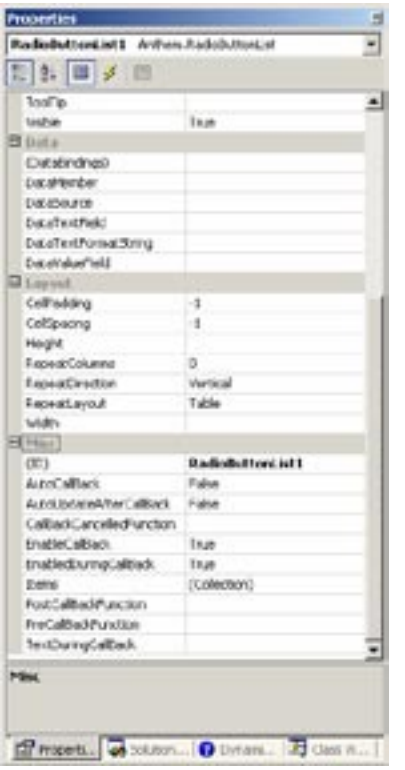
Met de TimerControl kan op een regelmatige interval een callback worden gemaakt naar de methode die in het tick-event van het TimerControl is gedeclareerd. Met behulp van deze control kan een andere control regelmatig ververst worden en bijvoorbeeld een progressbar worden ontwikkeld.

De voordelen voor de eindgebruiker en ontwikkelaar

Voordelen van het gebruik van Ajax voor de gebruiker is dat de applicatie sneller reageert. Dit is vooral het geval als de tijd bij een

```
<%@ Page Language="C#" %>
<%@ Register TagPrefix="anthem" Namespace="Anthem"
  Assembly="Anthem" %>
<script runat="server">
  protected void textbox1_TextChanged(object sender, EventArgs e)
  {
    //Om verschil tussen postback en callback te laten zien
    System.Threading.Thread.Sleep(500);
    message.Text = textbox1.Text;
  }
</script>
<html xmlns="http://www.w3.org/1999/xhtml">
  <head id="Head1" runat="server">
    <title>Anthem test</title>
  </head>
  <body>
    <form id="form1" runat="server">
      <h1>Anthem TextBox voorbeeld</h1>
      <p>
        Type iets in de textbox ga er met tab uit.
        De ingevoerde tekst verschijnt in de message label.
      </p>
      <anthem:TextBox ID="textbox1"
        runat="server"
        AutoCallBack="True"
        OnTextChanged="textbox1_TextChanged"
        CausesValidation="false"
        TextDuringCallBack="working..." />
      <asp:Label  ID="textboxlabel"
        runat="server"
        AssociatedControlID="textbox1" />
      <br />
      <anthem:Label   ID="message"
      runat="server"
      AutoUpdateAfterCallBack="true" />
    </form>
  </body>
</html>
```

Codevoorbeeld 1



Afbeelding 1. Radiobuttonlist met extra categorie Misc

postback gebruikt wordt voor het sturen van data naar de browser en het opbouwen van het scherm en minder door de verwerking op de server. In dit geval zal de versnelling door het gebruik van Ajax maximaal zijn. Indien Ajax-callbacks door server-verwerking lang duren, kan bij de gebruiker verwarring ontstaan, omdat er geen feedback op het browserscherm plaatsvindt. In dit geval is het verstandig om een bericht, bijvoorbeeld 'een moment a.u.b…', op het scherm te tonen. In Anthem.NET is dit geïmplementeerd met behulp van de property TextDuringCallBack. Voor de developer kost het erg weinig moeite om Anthem.NET te implementeren. De web.config hoeft niet aangepast te worden en er hoeft geen JavaScript te worden gemaakt om de Anthem.NET-controls te laten werken. Voor de rest werken de Anthem.NET-controls gelijk aan de standaard ASP.NET-controls.

Samenvatting

Met Ajax kun je ASP.NET-webapplicaties een betere gebruikerservaring en productiviteit geven. Deze komen vooral tot hun recht als de tijd om de pagina's op te bouwen lang is ten opzichte van de verwerkingstijd op de server. Doordat slechts een deel van de data naar de server wordt gestuurd, kan het gebruik van Ajax ook snelheidswinst opleveren. Voor het genoemde project is voor het Anthem-framework gekozen vanwege de eenvoudige implementatie. Het Anthem-framework draait ook op ASP.NET 1.1. Het Anthem-framework richt zich op het implementeren van Ajax-functionaliteit van de bestaande ASP.NET-controls. Bij andere problemen, zoals bij maps.google.com, zijn deze controls wellicht minder bruikbaar. In dit geval zijn andere frameworks (zoals AJAX.NET of Atlas) wellicht een betere oplossing. Indien ASP.NET 2.0 gebruikt wordt, heeft Atlas als extra voordeel, dat ook de Atlas-servercontrols gebruikt kunnen worden.

Peter Windemuller is als ontwikkelaar werkzaam bij DiVa-Digital Value in Wageningen (www.diva.nl). Zijn e-mailadres is peterw@diva.nl

Referenties

http://anthemdotnet.com
http://atlas.asp.NET
http://www.magicajax.NET
http://www.ajaxpro.info