Microsoft Ajax (MS Ajax), voorheen bekend als Atlas, is eind januari gereleased. Met Ajax (Asynchronous Javascript en XML) is een ontwikkelaar in staat om data te versturen van de client naar de server zonder dat daarbij een webpagina opnieuw opgebouwd hoeft te worden. MS Ajax stelt je in staat om snel en eenvoudig rijke client-side webapplicaties te ontwikkelen. Het is volledig gratis in gebruik en zal in de toekomst een standaardonderdeel worden van het .NET framework.

De ScriptManager centraal

Microsoft Ajax voor client-side webapplicaties

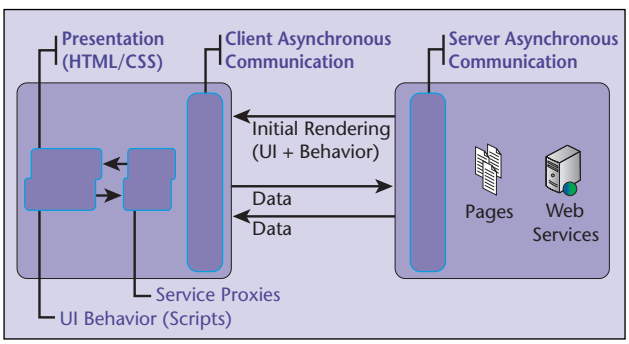
De volgende browsers ondersteunen momenteel MS Ajax: Internet Explorer, Firefox en Safari. Opera ondersteunt de meeste Ajax-scenario's helaas nog niet helemaal. In dit artikel bespreek ik de basisprincipes om met Microsoft Ajax te kunnen werken. Voorbeelden van gebruiksvriendelijke toepassing van Ajax zijn bijvoorbeeld het aanroepen van een webservice vanuit JavaScript en delen van de webpagina updaten zonder dat de gebruiker lang moet wachten. Dit kan je bereiken door gebruik te maken van:

- MS ASP.NET 2.0 AJAX Extensions
- MS AJAX Library
- MS AJAX Toolkit

MS ASP.NET 2.0 Ajax Extensions is de categorie waarin je gebruik kunt maken van speciale server controls. De server controls gebruik je net als de standaardcontrols uit het .NET Framework, zoals bijvoorbeeld het DataGrid. Je kan ze direct vanuit de toolbox in Visual Studio op je webpagina slepen. Controls die bij de Ajax Extensions horen zijn: ScriptManager, ScriptManagerProxy, UpdatePanel, UpdateProgress en Timer.

De MS Ajax library is minder bekend, maar minstens zo belangrijk en biedt de ontwikkelaar krachtige mogelijkheden. De MS Ajax library is een JavaScript bibliotheek welke gebruikt wordt aan de client-side (browser). De bibliotheek biedt meer vrijheid en mogelijkheden dan de Ajax Extensions, maar betekent meer codeerwerk. Hierbij kun je denken aan het aanroepen van webservices en objectgeoriënteerd werken vanuit JavaScript.

De derde en laatste categorie is de MS Ajax Toolkit. Wanneer je deze installeert, krijg je de beschikking over een aantal innovatieve controls. Deze controls zijn voor ontwikkelaars bedoeld en kunnen (net als de controls uit de Ajax Extensions) direct vanuit de toolbox op de webpagina gesleept worden. Het verschil ten opzichte van controls die MS Ajax Extensions bieden is dat die controls worden gezien als de kern van MS Ajax. De controls uit Toolkit zijn een uitbreiding van nieuwe functionaliteiten. Wat deze controls gemeen hebben, is dat zij met een paar configuratiehandelingen je webapplicatie een stuk gebruiksvriendelijker en aantrekkelijker maken. Een goed voorbeeld hiervan is DragPanelExtender control. Deze control biedt de mogelijkheid om objecten te kunnen verslepen. Sleep dit control op de webpagina. Met de property DragHandleID geef je aan op welk object de gebruiker moet klikken om de sleepactie uit te voeren. Met



Asynchrone client/server communicatie

TargetControlID geef je aan welk object er daadwerkelijk versleept moet worden.

```
<cc1:DragPanelExtender ID="DragPanelExtender1"
runat="server" DragHandleID="DragArea"
TargetControlID="PanelToDrag"></cc1:
DragPanelExtender>
```

Door deze simpele stappen te doorlopen heb je snel een webpagina gemaakt waar drag and drop toegepast wordt. De Toolkit zal niet verder in het artikel besproken worden. Zie aan het eind van dit artikel voor de url naar de website voor meer informatie. Op de website staan presentaties waarin je kunt zien wat je met deze controls kunt bereiken.

XmlHttpRequest MS Ajax wordt mogelijk gemaakt door het XmlHttpRequest object. Dit object is beschikbaar vanaf Internet Explorer 5, Mozilla 1.0, Opera 9 en Safari 1.2. Met dit object is het mogelijk om over HTTP asynchrone aanroepen naar een webserver te doen. MS Ajax is dus eigenlijk geen nieuwe technologie, Microsoft heeft echter het gebruik ervan enorm vereenvoudigd. Ontwikkelaars hoeven niet meer te programmeren tegen het complexe XmlHttpRequest object, maar tegen een objectgeoriënteerde API. Dit maakt ons leven als ontwikkelaars een stuk eenvoudiger.

De ScriptManager Een van de belangrijkste controls binnen het MS Ajax framework is de ScriptManager. Wanneer je MS Ajax installeert krijg je de beschikking over deze control. De naam van de control geeft aan waar hij voor dient, het managen van scripts (script library's). Script library's bieden op hun beurt weer functionaliteiten aan. De script library's die bij de installatie van MS Ajax meegeleverd worden, staan omschreven in tabel 1.

MicrosoftAjax	De basis van MS Ajax. Type system extensions wordt hiermede mogelijk gemaakt.
MicrosoftAjaxTimer	Bevat de TimerControl klasse
MicrosoftAjaxWebForms	Deze library is belangrijk voor partial rendering mogelijkheden. Hierin zit de PageRequestManager class.
PreviewDragDrop	Deze library biedt de mogelijkheid om gebruik te maken van drag and drop
PreviewGlitz	Bevat een aantal animations om je webpagina wat mooie effecten te geven. Bijvoorbeeld de mogelijkheid om de transparantie van een control te wijzigen.
PreviewScript	Deze library managed de Http Requests (webrequestmanager)

Tabel 1

Naast de scriptlibrary's die worden meegeleverd met MS Ajax is de ontwikkelaar in staat om zelf scripts toe te voegen aan het ScriptManager control. Wanneer je scripts toevoegt aan de collection van de ScriptManager, wordt er voor gezorgd dat de desbetreffende scripts naar de browser wordt gestuurd. De scriptlibrary's die in tabel 1 staan komen niet voor in de scripts collection. Als ontwikkelaar hoef je geen rekening te houden met deze scriptlibrary's, dit wordt allemaal voor je gedaan door de ScriptManager. De ScriptManager bevat nog een belangrijke collection genaamd Services. Hier kan je referenties naar webservices opgeven.

Je hebt het ScriptManager control nodig om de volgende MS Ajax functionaliteiten mogelijk te maken:

- Partial rendering
- Type system extensions
- Webservice calls vanuit JavaScript
- Custom Authentication Service en profielinformatie

De eerste drie functionaliteiten worden hierna nader beschreven.

Partial rendering Partial rendering houdt in dat een webpagina niet geheel opnieuw hoeft te laden wanneer een postback plaatsvindt. Voorheen resulteerde ieder event tot het herladen van de gehele webpagina (bijvoorbeeld bij het selecteren van een item in een dropdown-menu). Dit scenario behoort tot het verleden wanneer je gebruik maakt van partial rendering. Om hier gebruik van te maken plaats je een ScriptManager op je webform. De property EnablePartialRendering van de ScriptManager staat standaard op true, wat noodzakelijk is voor partial rendering. Doordat de property EnablePartialRendering van de Scriptmanager op true staat, komt het PageRequestManager object beschikbaar. De PageRequestManager bevindt zich in de MicrosoftAjaxWebForms script library en is verantwoordelijk voor het beïnvloeden van de page-lifecycle. De PageRequestManager stelt property's, methods en events beschikbaar waardoor een ontwikkelaar de partial rendering kan beïnvloeden. Naast de ScriptManager heb je een UpdatePanel nodig. Een UpdatePanel kan je zien als een frame waarin je controls kunt stoppen die gebruik maken van partial rendering. Deze controls plaats je in het element <contenttemplate> van het UpdatePanel. Om deze controls te updaten moet er een event plaatsvinden, bijvoorbeeld wanneer een gebruiker op een knop klikt of een dropdown selectie wijzigt. Wanneer de control die het event afvuurt zelf buiten het update-panel staat moet

je deze in het UpdatePanel definiëren. Hierdoor weet de ScriptManager dat wanneer de control een event afvuurt er een partial rendering moet plaatsvinden. Deze controls worden gezien als de triggers en worden in het element <triggers> geplaatst. Als de control binnen het UpdatePanel staat hoef je geen trigger te definiëren. Als je een control als trigger wilt opgeven, dan heb je de mogelijkheid om aan te geven of je wil dat hij voor een synchrone of een asynchrone postback zorgt.

```
<asp:AsyncPostBackTrigger ControlID="Button1"
EventName="Click" />
<asp:PostBackTrigger ControlID="Button1" />
```

Wanneer je voor een postbacktrigger kiest, zal je zien dat de rest van de pagina ook wordt geupdate. Een PostBackTrigger is nuttig voor controls die in zich in het updatepanel bevinden en een full postback nodig hebben om het gewenste resultaat te bereiken. Standaard maken controls die binnen het UpdatePanel staan gebruik van asynchrone postback calls. Dit is ook noodzakelijk om gebruik te kunnen maken van partial rendering. Wanneer je ervoor kiest om een asyncpostback te doen en je zet een break point in je codebehind, zie je dat er op de server niets verandert. Hij doorloopt hetzelfde proces. Het verschil zit hem in het renderen van de pagina. Partial rendering zorgt er voor dat alleen delen van de pagina opnieuw worden getekend. Kortom, we hebben gezien dat je gebruik kunt maken van partial rendering zonder JavaScript te schrijven. Dit alleen door gebruik te maken van een ScriptManager control en een UpdatePanel.

Type system extensions Een andere mogelijkheid die de ScriptManager biedt is 'Type System Extensions'. Deze techniek maakt het mogelijk dat een ontwikkelaar objectgeoriënteerd kan ontwikkelen in JavaScript. Hierdoor kan je sneller JavaScript ontwikkelen door het vermijden van dubbele code en is de code die je schrijft beter beheerbaar. Daarnaast kan je net als in het .NET Framework met een lagenmodel werken en nieuwe functionaliteit kan snel en gemakkelijk worden geïntroduceerd. Wanneer je objectgeoriënteerd programmeerd in Javascript kan je gemakkelijk en snel een nieuwe GUI voor je objectmodel ontwikkelen. Het enige wat je moet veranderen, is de koppeling tussen je GUI-controls en de property's van bijvoorbeeld je User object. Net als in het .NET Framework zijn al deze objecten gebaseerd op het type Object. Met de MS Ajax library kun je het volgende creëren:

- Namespaces
- Classes
- Interfaces
- Enumerations

Voor alle bovengenoemde onderdelen geldt, dat wanneer je ze wilt aanmaken ze geregistreerd moeten worden. Dat kun je doen door register<NameSpace | Class | Interface | Enum> (params); aan te roepen. Voordat je een klasse kunt maken en registreren, dien je eerst een namespace te registreren. Om een klasse te registreren moet je namelijk opgeven tot welke namespace deze behoort.

Zoals je natuurlijk van objectgeoriënteerd programmeren verwacht, behoort overerving ook tot de mogelijkheden. Het is mogelijk een Employee-klasse te maken die zijn eigenschappen overerft van de Person-klasse. Bij de registratie van de Employee-klasse moet je opgeven van welke klasse hij overerft, bijvoorbeeld:

```
<namespace>.Employee.registerClass('<namespace
>.Employee', <namespace>.Person);
```

De eerste parameter geeft aan welke klasse je wilt registreren. De tweede parameter geeft aan van welke klasse er eventueel wordt overgeërfd. Je kunt nog een derde parameter meegeven waarin je een interface opgeeft, die de klasse implementeert. Hieronder zie je een voorbeeld van de definitie en registratie van de Person-klasse.

```
Type.registerNamespace("Avanade");

Avanade.Person = function(firstName, lastNa-
me, emailAddress) {
    var firstName = firstName;
    var lastName = lastName;
    var emailAddress = emailAddress;

    this.getFirstName = function() {
        return this.firstName;
    }

    this.setFirstName = function(firstName) {
        this.firstName = firstName;
    }

    this.getLastName = function() {
        return this.lastName;
    }

    this.setLastName = function(lastName) {
        this.lastName = lastName;
    }

    this.getName = function() {
        return this.firstName + ' ' + this.
lastName;
```

```
    }

    this.dispose = function() {
        alert('bye ' + this.getName());
    }
}
Avanade.Person.registerClass(Avanade.Person',
null, Sys.IDisposable);
```

Je kunt de registratie van klassen en dergelijke direct in je ASP.NET (aspx) pagina doen. Je kunt er ook voor kiezen om het in een aparte JavaScript-file te doen. Wanneer je daarvoor kiest moet je de volgende regel toevoegen onder aan het bestand: `Sys.Application.notifyScriptLoaded();`. Deze methode staat los van de klasse. Door middel van deze methode wordt de ScriptManager op de hoogte gebracht wanneer het script is geladen. Tot slot geven we de referentie naar het JavaScript-file op in de ScriptManager.

```
<Scripts>
     <asp:ScriptReference Path="~/JavaScript/
Person.js" />
</Scripts>
```

Kortom, voor het creëren van nieuwe klasses in JavaScript is de ScriptManager van belang. De ScriptManager stelt de scriptlibrary's beschikbaar waarmee je objecten kunt registreren.

Webservices aanroepen vanuit JavaScript

De laatste mogelijkheid van de ScriptManager die we hier zullen bespreken is de mogelijkheid om vanuit JavaScript gebruik te kunnen maken van webservices. MS Ajax biedt de ontwikkelaar op een eenvoudige wijze de mogelijkheid om gebruik te maken van webservices door middel van JavaScript proxy's. Wanneer je voorheen vanuit JavaScript een webservice wilde benaderen moest je tegen het XmlHttpRequest object programmeren. Dit is vele malen lastiger dan de mogelijkheid die MS Ajax biedt. Om een webservice te benaderen moet je het volgende element toevoegen aan de web.config van je webproject:

```
<remove verb="*" path="*.asmx"/>
       <add verb="*" path="*.asmx"
validate="false" type="Microsoft.Web.Script.
Services.ScriptHandlerFactory, Microsoft.
Web.Extensions, Version=1.0.61025.0,
Culture=neutral, PublicKeyToken=31bf3856ad364
e35"/>
```

De ScriptHandlerFactory die je hiermee registreert zorgt ervoor dat wanneer je een webservice call doet vanuit server-side code hij gebruik maakt van de oude handler:

```
System.Web.Services.Protocols.WebServiceHandl
erFactory
```

Wanneer je een webservice wilt benaderen vanuit JavaScript zorgt deze handler ervoor dat er een proxy wordt gegenereerd en dat er wordt gecommuniceerd door middel van JSON. JSON (JavaScript Object Notation)) is een text format voor het serializeren van gestructureerde data. JSON wordt standaard gebruikt wanneer je calls vanuit JavaScript doet zowel voor calls direct naar een webservice of via bridges. Vervolgens geef je aan van welke webservice(s) je gebruik wilt maken. Wanneer de webservice zich op hetzelfde domein als de webapplicatie bevindt, volsta je met de volgende toevoeging aan de ScriptManager.

```
<Services>
       <asp:ServiceReference Path="~/
Webservice/Ajaxwebservice.asmx" />
</Services>
```

Doordat je een referentie hebt opgegeven, wordt er automatisch een JavaScript-proxy gegenereerd voor de communicatie met de webservice. Dit lijkt sterk op het toevoegen van een webreference in een ASP.NET of WinForms project zoals we dat gewend zijn.

Doordat de Path property van de ServiceReference alleen naar lokale webservices kan refereren (Dit komt door Same-Domain restrictions) is het niet mogelijk om een webservice op te geven die buiten het domein valt. Dit kun je oplossen door gebruik te maken van een bridge file (*.asbx). In deze file geef je op waar de externe webservice zich bevindt. Hieronder staat een voorbeeld van een bridge-bestand:

```
<?xml version="1.0" encoding="utf-8" ?>
  <bridge namespace="Avanade"
className="Proxy">
  <proxy type="Microsoft.Web.Preview.Services.
BridgeRestProxy" serviceUrl="http://local-
host/Ajaxwebservice/Service.asmx/HelloWorld"
/>
     <method name="HelloWorld" >
     <input>
        <parameter name="string" />
```

```
    </input>
    </method>
</bridge>
```

In dit bridge-bestand wordt geen referentie gemaakt naar de webservice maar naar een methode die de webservice bevat (Helloworld). Wanneer je de serviceUrl hebt opgegeven, is je bridge-bestand in staat de webservice method aan te roepen. Nu moet je een methode definiëren die de bridge file bevat. Oftewel het bridge-bestand moet aan de ScriptManager aangeven welke methode hij heeft. Wanneer de webservice methode ook parameters bevat dien je die ook in je bridge-bestand te definiëren. Tot slot zal je net als bij het vorige webservice-voorbeeld een referentie moeten opgeven in de ScriptManager. Dit maal geef je de url naar het bridge-bestand op. Om gebruik te kunnen maken van bridge-bestanden dien je eerst nog een aanpassing te doen in IIS. Je AJAX-project moet nog geconfigureerd worden in IIS. Dit doe je als volgt:

Ga naar de website waar je gebruik wilt maken van bridges.

- Klik met rechtermuisknop op de virtual folder.
- Klik op 'properties'.
- In het tabblad 'virtual directory' klik je op 'configuration'.
- Klik op Add.
- Je moet hier een extensie en een executable toevoegen. De extensie is .asbx en de executable moet dezelfde zijn als die van de .aspx extensie. c:\windows\microsoft.net\framework\v2.0.50727\aspnet_isapi.dll

Voeg vervolgens het volgende element toe aan de web.config van je webapplicatie:

```
<buildProviders>
  <add extension=".asbx" type="Microsoft.Web.
Preview.Services.BridgeBuildProvider" />
</buildProviders>
```

De buildprovider is noodzakelijk om de bridgebestanden te compileren. Om van deze BridgeBuildProvider gebruik te kunnen maken moet je aan je webproject een referentie naar de Microsoft.Web.Preview.dll toevoegen. Wanneer je dit niet doet, is je applicatie niet in staat de BridgeBuildProvider in te laden. Tot slot dien je het volgende element ook toe te voegen aan de web.config van je webapplicatie.

```
<httpHandlers>
<add verb="*" path="*.asbx" type="Microsoft.
Web.Script.Services.ScriptHandlerFactory"
validate="false"/>
</httpHandlers>
```

Deze handler moet je toevoegen om de request omtrent het bridge-bestand af te handelen. Wanneer je dit allemaal hebt gedaan is je Ajax-website in staat om door middel van een bridge te communiceren met een externe webservice vanuit JavaScript.

Conclusie

We hebben in een vogelvlucht een aantal aspecten van MS Ajax behandeld. Hierbij is duidelijk geworden dat de ScriptManager een centrale rol speelt. De ScriptManager stelt script-library's beschikbaar welke krachtige functionaliteit toevoegen. De rol van de ScriptManager kan worden gezien als een ondersteunende rol voor het genereren van proxy's, het laden van scriptlibrary's en nog veel meer. De definitieve release van MS Ajax is inmiddels beschikbaar. De informatie in dit artikel is echter gebaseerd op de bèta 2-versie.

Referenties

- De meest recente documentatie van MS Ajax: http://ajax.asp.net/docs
- Mocht je nog vragen hebben over MS Ajax kun je ze stellen op: http://forums.asp.net/default.aspx?GroupID=34
- Zie de demo 'Developing ASP.NET 2.0 applications using AJAX' op http://Ajax.asp.net voor een korte introductie over MS Ajax.

Dennis van de Laar is als Associate Consultant werkzaam bij Avanade (www.avanade.com), een samenwerkingsverband tussen Microsoft en Accenture. Voor vragen of opmerkingen is hij te bereiken op dennisv@avanade.com of via zijn blog http://dennisv.net.