

jQuery: meer invloed op dynamische websites

EEN DOMAIN SPECIFIC LANGUAGE VOOR HET WEB

Eelco Gelton en Korijn van Golen

Door de komst van AJAX worden websites alsmaar interactiever. De grens tussen client-side en server-side verwerking vervaagt steeds verder en gebruikers krijgen meer invloed in bijvoorbeeld het organiseren van de gebruikers interface. Microsoft levert het JavaScript framework jQuery bij Visual Studio en ondersteunt de open-source technologie in het ASP.NET AJAX framework. jQuery heeft een steeds grotere invloed op dynamische website.

In dit artikel lopen wij langs jQuery's belangrijkste sterke punten en zullen wij in een eenvoudig voorbeeld de kracht ervan laten zien. Aansluitend demonstrenen we hoe jQuery's plugin functionaliteit werkt en sommen we een paar bijzondere plugins op. Vervolgens laten we zien hoe men de brug tussen jQuery en ASP.NET legt, met of zonder Ajax.NET. Tot slot is er nog een Case Study waarbij een personaliseerbare interface wordt gebouwd.

Waarom zou ik jQuery gebruiken?

Abstractie. jQuery is in essentie een Domain Specific Language voor het web. Allerlei functionaliteit ligt binnen handbereik en is al een lange tijd in ontwikkeling. Er zijn veel JavaScript experts constant bezig om de code zo goed mogelijk te optimaliseren. Je kunt bij een nieuwe web applicatie alle JavaScript code zelf gaan schrijven en opbouwen, maar jQuery levert je waar-schijnlijk alles wat je nodig hebt in één keer aan met een geoptimaliseerde data flow en cross-browser compatibility. Dit brengt ons bij het volgende punt:

Algemene taal. jQuery heeft een zeer eenvoudige syntax en er is duidelijke documentatie van de API beschikbaar. Er is onlangs ook support voor Visual Studio's Intellisense toegevoegd. Met je collega's gebruik maken van één framework is gemakkelijker dan een zelfgebouwde library te leren gebruiken.

Efficiency. Met een leercurve van ongeveer een halfuur, DOM selectie via CSS3 & XPath (!), chaining en alle DOM manipulatie die beschikbaar is, is jQuery met gemak het efficiëntste JS Framework op het web. Omdat de library cross-browser compatible is zult u ook niet langer aan browser-hacks en/of -sniffing te hoeven doen[3].

Impact op uw applicatie. jQuery is lichtgewicht: het kost in totaal 19KB om een gecompriëerde versie van de library in een project te gebruiken. Daarnaast is het geschikt voor cross-browser verwerking.

Een voorbeeld

Met jQuery kan veel worden bewerkstelligd in weinig regels code[4]. Elementen worden geselecteerd door middel van een syntax met het \$ symbool. Met de volgende regels code[5], selecteren we de 'p' tag met de CSS class 'neat', voegen we de CSS class 'ohmy' toe en activeren we een animatie waarbij het element langzaam verschijnt. Dat alles gekoppeld op een cross-browser manier aan het document.ready event:

```
$(document).ready(function() {
    $("p.neat").addClass("ohmy").
    show("slow");
});
```

Het voorbeeld illustreert hoe een nieuwe CSS klasse kan worden toegepast op een bestaand element. Tussen de haken achter het \$ symbool kan een CSS selector wor-

den gebruikt of een XPath expressie. jQuery werkt door middel van een set functies die altijd het jQuery object weer returnen. Op die manier kunnen ook eendeloos veel functies aan elkaar gechained worden. In het volgende voorbeeld laten we zien hoe gemakkelijk de rijen van een tabel om en om gekleurd kunnen worden middels een CSS klasse. Verder ook een voorbeeld om :hover in IE6 gemakkelijk te simuleren voor bijv. uitklappende pure CSS navigatie menu's.

```
// zebra striping
$("table tr:nth-child(odd)").
addClass("highlight");
// mouse hover
$("table tr").hover(function() {
    $(this).addClass("hover");
}, function() {
    $(this).removeClass("hover");
});
```

Met de selector ":nth-child(odd)" wordt per parent (=tabel) de filter ':odd' toegepast. Doordat we de klasse 'hover' aan de <tr> tags toevoegen hebben we nu ook een bug in IE6 omzeild (deze ondersteunt geen :hover op elementen die geen <a> tag zijn). Dit alles kan ook in één keer met jQuery's chaining:

```
$("table tr").hover(function() {
    $(this).addClass("hover");
}, function() {
    $(this).removeClass("hover");
}).filter(":nth-child(odd)").
addClass("highlight");
```

Plugins

Behalve de standaard jQuery library, is er ook een enorme hoeveelheid plugins beschikbaar. Zo bestaan er eenvoudige plugins om PNG support toe te voegen aan oudere browsers, het opbouwen van menu's dynamisch te maken, validatie, paginering, table sorting enzovoorts, maar ook geavanceerde plugins die je een image laten croppen of een geanimeerde fotogallerij via AJAX bewerkstelligen. Ook is er een officiële uitbreiding genaamd UI, waarin een flink aantal grafische toepassingen van jQuery zijn uitgewerkt.

Plugin authoring

De reden dat er zoveel plugins beschikbaar zijn is dat het zo eenvoudig is om er zelf één te maken. Het voorgaande voorbeeld waarbij we een table zebra-stripping en hover highlighting gaven zouden we eenvoudig in een plugin kunnen veranderen op de volgende manier:

```
jQuery.fn.zebra = function(options) {
  // we overschrijven de standaard waarden
  // enkel als ze zijn ingegeven
  // door jQuery.extend te gebruiken
  settings = jQuery.extend({
    selector: ":nth-child(odd)",
    hoverClass: "hover",
    highlightClass: "highlight"
  }, options);
  // do the plugin
  $(this).hover(function() {
    $(this).addClass(settings.hoverClass);
  }, function() {
    $(this).removeClass(settings.hoverClass);
  }).filter(settings.selector).
  addClass(settings.highlightClass);
  // always return jQuery object for chaining
  return this;
}
```

De plugin gebruiken we nu bijvoorbeeld als volgt:

```
$("#table tr").zebra();
```

We kunnen nu ook bepaalde functies ingeven of een andere lijst van objecten zebra-stripping meegeven, zoals bijvoorbeeld een of een andere willekeurige lijst van elementen. In het volgende voorbeeld stellen we de zebra-stripping in op alle tags en een aantal <div>. Tevens geven we wat opties mee om aan te sluiten op de CSS.

```
$("#ul li, div.zebraContainer > div").zebra({
  hoverClass: "over",
  highlightClass: "oddRow"
});
```

	2000	2001	2002	2003	2004	2005	2006
Mary	150	160	40	120	30	70	70
Tom	3	40	30	45	35	48	70
Brad	10	00	10	85	25	79	70
Kate	40	80	90	25	15	119	200

Enkele populaire plugins

Middels een korte toelichting, een URL en een stukje code zullen we een aantal van de populaire plugins bespreken.

hoverIntent

Het gebeurt u waarschijnlijk weleens dat u een stukje zit te lezen, met uw muis even een ander venster wil openen en per ongeluk over het menu van de website zweeft: weg is uw stuk tekst. Deze plugin voorkomt dit door een speciaal mouseHover event te koppelen aan jQuery dat alleen triggert wanneer iemand's muis vertraagt tot onder een bepaalde snelheid terwijl het boven een element hangt, gedurende een minimale delay.

<http://plugins.jquery.com/project/hover>

```
$("#ul.nav li").hover(hoverFn);
```

iFixpng

Deze plugin voegt PNG compatibility toe aan browsers die het niet ondersteunen. De functie .ifixpng() kan ook dynamisch gebruikt worden wanneer bijv. een nieuwe PNG wordt toegevoegd aan de pagina.

<http://plugins.jquery.com/project/iFixPng>

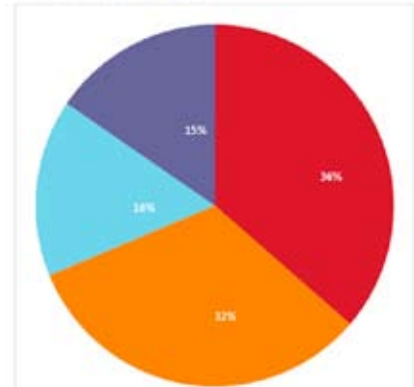
```
$.ifixpng("/images/pixel.gif");
$("#img[src$=png], .bgPng").ifixpng();
```

fgCharting

Deze indrukwekkende plugin kan de data uit een HTML tabel visualiseren. Dit gaat met behulp van Canvas, een onderdeel van JavaScript dat in staat is vectoren op het scherm te tekenen. Het gaat hier om o.a. lijngrafieken, cirkeldiagrammen, histogrammen en nog enkele andere varianten. Deze grafiek kan ook weer dynamisch worden geupdate (wat bijv. inhoud dat u grafieken kunt animeren of data dynamisch kunt inladen).

http://www.filmgroup.com/lab/creating_accessible_charts_using_canvas_and_jquery

Pie Chart Generated from HTML table:



```
$.fgCharting();
<table id="dataTable"><!-- data --></table>
<canvas id="canvas1" class="fgCharting_src-dataTable_type-pie"></canvas>
```

Toelichting: de plugin leest alle 'canvas' tags in en verwerkt de class. Hierin staat eerst 'fgCharting' wat aangeeft dat de plugin het element moet verwerken en daarna staan key/value pairs gescheiden door underscores (_) en gekoppeld door dashes (-). Dit is slechts een mogelijkheid, de instellingen per grafiek kunnen ook in het JavaScript worden ingegeven.

jQuery en ASP.NET

jQuery kan op ASP.NET aansluiten op twee manieren: met of zonder AJAX.NET. In onze Case Study kiezen we ervoor geen gebruik te maken van AJAX.NET, om te illustreren hoe effectief jQuery kan zijn met enkel ASP.NET. In de tweede manier illustreren wij kort en bondig hoe AJAX.NET en jQuery samen kunnen gaan om optimaal gebruik te kunnen maken van beide libraries.

1. Zonder gebruik te maken van AJAX.NET

Allereerst includen we de jQuery library in het project en verwijzen we er naar in de HTML.

```
<script type="text/javascript"
charset="utf-8"
src="js/jquery-1.3.2.js"></script>
```

Vervolgens includen we ook het script in het project waarmee we IntelliSense inschakelen voor Visual Studio voor jQuery[6]. Let op: dit bestand moet dezelfde bestandsnaam hebben als het bestand waar het naar verwijst en eindigen op "-vsdoc.js". In ons geval dus "jquery-1.3.2-vsdoc.js". Bij onze Case Study hebben wij ook de jQuery UI library nodig:

```
<script type="text/javascript"
charset="utf-8"
src="js/jquery-ui-1.7.1.custom.min.js"></script>
```

Intellisense is helaas nog niet volledig compatibel met jQuery plugins en daarom zal er nu een Warning verschijnen in de Error List. Intellisense verwacht dat er per library een vsdoc file beschikbaar is. Deze is nog niet gemaakt voor jQuery UI en daarom maken we zelf een blanco vsdoc file aan en includen we die in het project. Deze heet in dit geval "jquery-ui-1.7.1.custom.min-vsdoc.js". De error is nu verdwenen, IntelliSense werkt weer en het project compileert. Nuttige info indien je ook een plugin gebruikt waarvoor nog geen vsdoc beschikbaar is.

2. Gebruikmakend van AJAX.NET

Om gebruik te kunnen maken van AJAX.NET libraries én jQuery zullen we jQuery moeten registreren bij de ScriptManager in plaats van in de HTML. Dat kan door de ScriptManager in de Toolbox te dubbelklikken of door hem zelf te schrijven. Daarna registreren we jQuery.

```
<asp:ScriptManager ID="ScriptManager1"
runat="server">
  <Scripts>
    <asp:ScriptReference Path="~/js/
jquery-1.3.2.js" />
  </Scripts>
</asp:ScriptManager>
```

Verder voegen we een UpdatePanel toe aan de pagina. Ter illustratie voegen wij een TextBox Control en een Button Control toe aan de ContentTemplate:

```
<asp:UpdatePanel ID="UpdatePanel1"
runat="server">
  <ContentTemplate>
    <asp:TextBox ID="textBox"
runat="server"></asp:TextBox>

    <asp:Button ID="button" runat="server"
Text="AJAX Request" onclick="button_Click"
/>
  </ContentTemplate>
</asp:UpdatePanel>
```

In de code behind van de pagina zetten we

nu de volgende code zodat we kunnen zien wanneer een AJAX Request is geslaagd:

```
protected void button_Click(object sender,
EventArgs e)
{
    textBox.Text += "*";
}
```

Het moge duidelijk zijn dat nu AJAX.NET functioneert en een asterisk toevoegt aan de textBox.Text wanneer er op de knop gedrukt wordt. jQuery is geïncludeerd in de code en men zou denken dat men er gebruik van kan maken. Dat is niet helemaal waar. Wanneer we de volgende code implementeren in de <head> van onze pagina wordt al gauw duidelijk wat er mis is.

```
<script type="text/javascript">
//
$(document).ready(function() {

    alert('$(document).ready() has been
called');
});
//]]&gt;
&lt;/script&gt;</pre>
</div>
<div data-bbox="383 415 658 586" data-label="Text">
<p>Het blijkt dat de code alleen wordt aangeroepen bij de eerste DOM Ready event en niet wanneer AJAX requests aflopen. Het probleem is dat hier ook geen globale events voor bestaan en dat AJAX.NET de DOM van het UpdatePanel volledig opnieuw inlaadt. jQuery raakt zijn DOM nodes dan kwijt en functioneert niet goed meer. Gelukkig is er wel een functie die AJAX.NET aanroept zodra de DOM Ready is; óók wanneer een AJAX Request afloopt. De oplossing is dus:</p>
</div>
<div data-bbox="393 600 600 632" data-label="Text">
<pre>&lt;script type="text/javascript"&gt;
//<![CDATA[
// binds main() to 'DOM Ready'</pre>
</div>
<div data-bbox="682 59 938 186" data-label="Text">
<pre>// and 'ASP.NET Ajax Complete' events
function pageLoad() {
    $(document).ready(main);
}
// main javascript code
function main() {

    alert('$(document).ready() has been
called');
}
//]]&gt;
&lt;/script&gt;</pre>
</div>
<div data-bbox="672 187 954 229" data-label="Text">
<p>Nu is de verbinding tussen AJAX.NET en jQuery klaar en kan er vrij gewerkt worden met beide libraries.</p>
</div>
<div data-bbox="672 239 949 271" data-label="Section-Header">
<h2>Case Study: Personaliseerbare interface</h2>
</div>
<div data-bbox="672 272 954 415" data-label="Text">
<p>In ons voorbeeld willen we de gebruiker de vrijheid geven om zelf de user interface vorm te laten geven. We maken een webapplicatie in Visual Studio 2008. We gebruiken drag-and-drop functionaliteit uit de UI library en illustreren ook het gebruik van AJAX callbacks, zodat de instellingen van de gebruiker kunnen worden opgeslagen. Het eindresultaat zal de volgende functionaliteit hebben: zie figuur 1.</p>
</div>
<div data-bbox="672 424 773 442" data-label="Section-Header">
<h2>Client-side</h2>
</div>
<div data-bbox="672 443 950 600" data-label="Text">
<p>Bij dit voorbeeld gaan wij er van uit dat de pagina bestaat uit een of meerdere divs met de CSS klasse 'column' die vervolgens 0 of meerdere divs met de CSS klasse 'box' bevatten. Ze hebben een ID attribuut dat begint met de string 'box' gevolgd door het numerieke id van de box (bijv. afkomstig uit een database), bijv. 'box37'. We beginnen nu met het aanmaken van het script om onze HTML content versleepbaar te maken:</p>
</div>
<div data-bbox="682 613 870 625" data-label="Text">
<pre>$(document).ready(function() {</pre>
</div>
<div data-bbox="384 645 853 905" data-label="Image">
<img alt="Screenshot of a web browser showing a user interface with four test boxes labeled Test 1, Test 2, Test 3, and Test 4. Test 1 is a smaller box overlapping the bottom of Test 2 and Test 3. The boxes contain placeholder text like 'Your select text. Your select text. Your select text.'"/>
</div>
<div data-bbox="383 914 437 927" data-label="Caption">
<p>FIGUUR 1.</p>
</div>
<div data-bbox="529 965 750 981" data-label="Page-Footer">
<p>.NET magazine | september 2009</p>
</div>
<div data-bbox="933 965 957 980" data-label="Page-Footer">
<p>35</p>
</div>
```

Voor jQuery zijn genoeg plugins beschikbaar om aan alle functionele eisen te voldoen en kan veel voor webapplicaties betekenen.

```
// alle divs van klasse 'column' worden sortable
$(".column").sortable({
  // essentiële instellingen
  connectWith: '.column',
  update: saveBoxes,
  // optionele instellingen
  placeholder: 'ghost box',
  revert: true,
  opacity: 0.7,
  handle: 'h1',
  containment: '.container'
});
```

Met deze uitzonderlijk kleine hoeveelheid code zijn alle divs in de columns nu sorteerbaar en tussen de verschillende columns te verslepen. Let wel dat de optie 'connectWith' alleen van toepassing is wanneer meer dan 1 column wordt gebruikt in de interface. In principe zijn alleen de bovenste twee opties essentieel voor ons voorbeeld. De rest is slechts opmaak en daarvoor wijs ik u door naar de documentatie van jQuery UI.

De verdere functionaliteit aan de client-side is de JavaScript functie `saveBoxes` (waarnaar wordt verwezen bij de optie 'update' in voorgaande listing). Deze functie wordt aangeroepen zodra de gebruiker klaar is met slepen en alle elementen op hun nieuwe DOM plaats zijn terechtgekomen.

We genereren een JSON (JavaScript Object Notation)[7] object om mee te geven aan de AJAX call. Dit is efficiënt aangezien ASP.NET een JSON object dynamisch om kan zetten naar een List object. Dit kan handmatig of met de JSON library[8].

```
// process currently present boxes and submit them ajax style
function saveBoxes() {
  // genereer een JSON array met de ids van alle boxen in
  // de nieuwe volgorde
  var boxes = new Array();
  var i = 0;
  $(".column").each(function() {
    boxes[i] = $(this).sortable('toArray');
    i++;
  });
  var boxesJSON = JSON.stringify({'boxes':boxes});
  // AJAX verzoek insturen naar de WebMethod SaveBoxes in Default.aspx
  $.ajax({
    type: "POST",
    url: "Default.aspx/SaveBoxes",
```

```
data: boxes,
contentType: "application/json; charset=utf-8",
dataType: "json",
success: function(msg) {
  // doe iets met het antwoord
}
});
```

De AJAX call kan nog korter, door middel van de shorthand `$.post` of `$.get` maar ter illustratie geven wij hier een grotere selectie van de opties weer.

Server-side

Vervolgens gaan we ervoor zorgen dat er een AJAX postback plaatsvindt zodra de drag-and-drop is voltooid. Dit doen we door een WebMethod [9] aan te maken in de code behind van onze aspx pagina. Deze WebMethod geven we een parameter mee in de vorm van een `List<List<int>>` object dat verder verwerkt kan worden. De JSON array die we mee geven wordt direct ingelezen door ASP.NET.

```
using System.Web.Services;
[WebMethod]
public static string
SaveBoxes(List<List<string>> boxes)
{
  return "0"; // geef iets terug aan de GUI
}
```

Deze verbinding stelt ons in staat om de data op te slaan en zo de gebruikers interface dynamisch te maken.

Conclusie

jQuery is een cross browser JavaScript Framework die veel voor uw webapplicatie kan betekenen:

- De cross-browser compatibiliteit betekent dat u geen browser-hacks of -sniffing[3] meer nodig zult hebben;
- De optimalisatie en chaining zorgen ervoor dat uw scripts zo snel en efficiënt mogelijk werken;
- De library werkt goed samen met het ASP.NET AJAX Framework en Intellisense;
- En er zijn genoeg plugins beschikbaar om aan al uw functionele eisen te voldoen.

Links

jQuery + Intellisense file:

http://docs.jquery.com/Downloading_jQuery_VS2008_SP1:

<http://www.microsoft.com/downloads/details.aspx?FamilyId=FBEE1648-7106-44A7-9649-6D9F-6D58056E>

Hotfix KB958502:

<http://code.msdn.microsoft.com/KB958502>

Evt. plugins:

<http://plugins.jquery.com/>

<http://www.jquery.com>

<http://blog.jquery.com/2008/09/28/jquery-microsoft-nokia/>

http://docs.jquery.com/Sites_Using_jQuery

http://nl.wikipedia.org/wiki/Browser_sniffing

http://docs.jquery.com/How_jQuery_Works

http://docs.jquery.com/Downloading_jQuery

<http://www.json.org>

<http://www.json.org/json2.js>

<http://msdn.microsoft.com/en-us/library/byxd99hx.aspx>

.....
Eelco Gelton, is directeur-eigenaar van Sentinel IT en afgestudeerd in Computer Science aan de TU Delft. Voor vragen en opmerkingen is hij te bereiken op eelco@sentinel-it.nl.

.....
Korijn van Golen, is programmeur bij Sentinel IT en studeert Computer Science aan de TU Delft. Voor vragen en opmerkingen is hij te bereiken op korijn@sentinel-it.nl.