Efficiënter werken met SharePoint Fluency

Oplossingen die écht integreren

Waldek Mastykarz en Wouter van Vugt

In Office 2007 is door Microsoft de Fluent Interface geïntroduceerd: een revolutionair interface concept dat het werken met Office eenvoudiger maakt. Na het succes van deze interface in Office wordt nu ook in SharePoint Fluency geintroduceerd. Het werken met SharePoint was nog nooit zo intuïtief! Dit betekent een nieuwe frisse blik en oplossingen die écht integreren. Hou je hart vast voor een hoge dosis JavaScript.

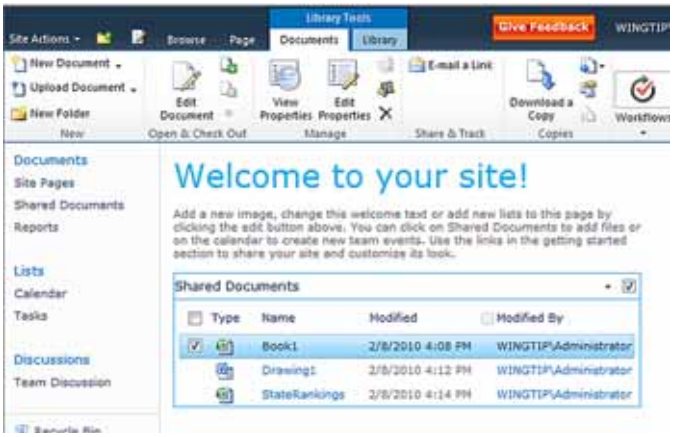
SharePoint 2010 betekent een hoop. Composite applicaties met de BCS Runtime, een enterprise search platform met FAST, een BI portal met Performance Point. Voor de kern van SharePoint is vanuit een ontwikkelaarsperspectief misschien niet zo veel anders. Lijsten en bibliotheken, site columns en content types. Toch is ook hier een belangrijke verandering in de berichtgeving en uitleg. Terwijl SharePoint 2007 toch vooral als lijsten bibliotheken werd gezien, is het nieuwe 2010 mantra: "Een SharePoint site is een verzameling van pagina's"

Herhaal deze mantra maar een keer. Waar SharePoint 2007 toch nog vooral als een verzameling van lijsten en libraries werd gezien, is SharePoint 2010 een collectie van pagina's die best wel wat lijsten mogen laten zien. Een goede verandering, want lijsten en libraries: laat dit nu eens niet overeenkomen met de werkelijkheid van de gebruiker. Immers, niet iedereen denkt in tabellen en foreign keys.

Een verzameling van pagina's

Dus, pagina's is wat de klok slaat in SharePoint 2010. En om dit concept te ondersteunen zijn er verschillende features aan het platfom toegevoegd. Natuurlijk valt qua Fluent Interface de Ribbon op. Het is veel meer dan een simpele verplaatsing van commando's van een toolbar naar wat knoppen bovenin. Omdat de Ribbon op context gebaseerd is, kunnen gebruikers verschillende opties gemakkelijker vinden. Zo is het invoeren van een plaatje op een pagina één enkele actie terwijl er in de vorige versie nog het tienvoudige voor nodig was. Daar moeten je polsen wel blij van worden!

Naast het aanbieden van extra klik-kracht door middel van de Ribbon is er ook gedacht aan het verminderen van navigaties. Er is een rijke dosis JavaScript over SharePoint uitgesprinkeld die ervoor zorgt, dat je tijdens het werken met gegevens vooral binnen de context van die gegevens kan blijven. Zo zijn lijstgegevens aanpasbaar via dialogen die met behulp van AJAX automatisch de pagina's verversen. Helemaal aangenaam is dat een Web Part binnen een pagina op een willekeurige plaats kan worden ingevoegd. Hierbij hoeft er geen ontwerp met Web Part zones te worden gemaakt, maar indien rigide structuur wenselijk is, mag dit natuurlijk nog wel. De Ribbon is daarbij volledig op de hoogte van Web Part selecties en biedt alle beheer en bewerkingsopties die je vroeger alleen toegankelijk kon maken door naar de lijst zelf te navigeren. Aardig cool, zo'n stukje screen real-estate.



FIGUUR 1: EEN LIST VIEW WEB PART GESELECTEERD OP EEN WIKI PAGINA

Ook binnen de context van een lijst zijn er wat opmerkelijke dingen aan de hand. Iets waar je vroeger toch echt de ontwikkelaarssloffen moest aantrekken, kan je nu als gebruiker de interactie met SharePoint lijsten veel rijker maken. De List Form Web Parts die worden gebruikt om de Display/New/Edit formulieren te maken kan worden vervangen! Het InfoPath team heeft hier fijntjes op ingespeeld door het InfoPath Form Web Part te gebruiken als mogelijke vervanger. In SharePoint Designer is het een fluitje van een cent om een InfoPath formulier te koppelen. Hierbij mag je nu ook meerdere formulieren van een bepaald type maken, dus twee soorten Edit forms is geen probleem. Het mooie aan InfoPath is dat het nu maar een paar klikken kost om bijvoorbeeld ex-

tra validatie toe te voegen, of om extra gegevens op te halen en te presenteren in een drop-down. Power to the power-user!
Een laatste toch wel noemenswaardige feature is In-Place Editing. Hierbij kan je direct in de List View pagina een item bewerken. Het is weer net ASP.NET met een editbaar grid.

FIGUUR 2: IN-PLACE EDITING

Onder de motorkap

Bij het herstructureren van de SharePoint User Experience is bij de basis begonnen. SharePoint is immers ook maar een gewone web applicatie die wat HTML oplepelt. De HTML van SharePoint pagina's is een stuk verbeterd. Hierdoor laden de pagina's sneller en worden zij door meerdere browsers correct getoond. Terwijl SharePoint 2007 alleen Internet Explorer 6, 7 en later 8 goed ondersteunde, is deze lijst door SharePoint 2010 met Mozilla Firefox en Safari uitgebreid. Internet Explorer 6 wordt gelukkig achter gelaten. Iets tè legacy. Het SharePoint team mag op het resultaat best trots zijn. Het is vast niet voor niets dat veel SharePoint demo's met Mozilla Firefox worden gegeven!

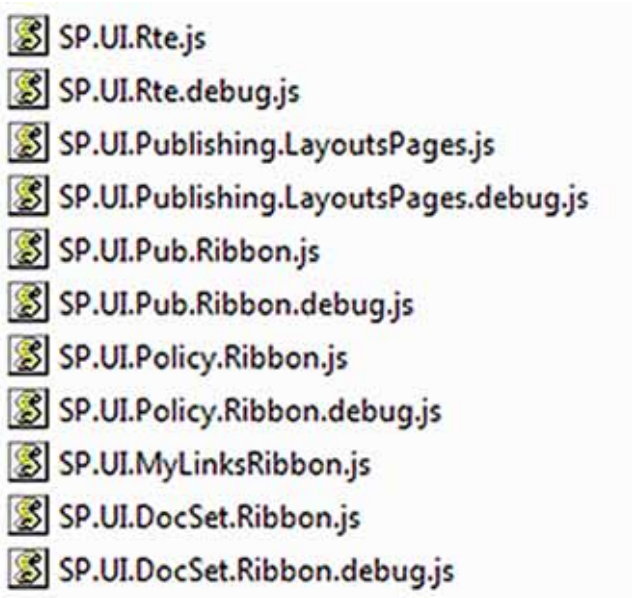
FIGUUR 3: STANDAARD SHAREPOINT 2010 TEAM SITE IN MOZILLA FIREFOX 3.5.

Ontwikkelaars zullen blij verrast zijn om te zien, dat SharePoint niet meer tussen HTML 4, en HTML ongedefinieerd zweeft, maar kucheurige XHTML genereert. De beta geeft overigens nog wel wat validatie foutjes. Nu we het toch over opmaak hebben, helemaal onverwacht is dat de opmaak van SharePoint WCAG 2.0 AA compliant is. SharePoint 2010 is ook stuk slimmer geworden ten aanzien van CSS gebruik. De oude CSS BLOB is in de nieuwste versie over meerdere bestanden verdeeld. Afhankelijk van de gebruikte functionaliteiten laadt SharePoint alleen die CSS definities die op dat moment nodig zijn.JavaScript, de verloren gave van 2007 ontwikkelaars
Om UI ontwikkeling op het platform te standaardiseren heeft SharePoint het Common UI Control Platform geintroduceerd. Dit platform bevat een standaard componenten zoals Dialog Windows en Thema's maar ook een aantal richtlijnen die in maatwerk applicaties kunnen worden toegepast. Een belangrijk onderdeel binnen dit client-side platform is Script On Demand. Met behulp van dit mechanisme is het mogelijk het laden van JavaScript uit te stellen totdat het wordt aangeroepen.

```
ScriptLink.BuildDelayedExecutionScript("MyScriptFile.js", "MyInit-
Method", "arg1", "arg2");
```

CODEVOORBEELD 1: REGISTRATIE VAN EEN ON-DEMAND SCRIPT.

Omdat hierdoor niet alle JavaScript tegelijkertijd met de pagina hoeft te worden geladen, zullen de pagina sneller door de browser worden verwerkt en getoond. Ook aan de grootte van JavaScript bestanden is gedacht. Eerder was één van deze bestanden zelfs 250KB! Grote JavaScript en CSS bestanden zijn geen SharePoint-specifieke uitdaging. Alle webapplicaties, die rijke functionaliteiten aanbieden, hebben ermee te maken. Vooral de nieuwste van deze applicaties maken gebruiken van het zogenaamde minification proces. In dat proces worden alle optionele tekens, zoals spaties, tabs en enter verwijderd waardoor de grootte van het bestand afneemt. SharePoint 2007 gebruikte veel JavaScript die niet volgens stricte regels was opgebouwd. Daardoor kon het niet met behulp van minification worden verkleind en was IIS compressie de enige optie voor optimalisatie.Ten behoeve van SharePoint 2010 heeft het SharePoint team alle JavaScript bestanden geoptimaliseerd. Niet alleen zijn de scripts gemoderniseerd maar het wordt ook standaard geminified opgeleverd!



FIGUUR 4: MINIFIED EN NORMALE SCRIPT FILES

Minification van script belemmert de leesbaarheid. Het is daarom gebruikelijk om tijdens het ontwikkelproces standaard JavaScript bestanden te gebruiken en minification als een onderdeel van een release in te bouwen. Op deze manier kunnen ontwikkelaars eenvoudig code onderhouden en debuggen en worden er in de productieomgeving de verkleinde versies van de bestanden gebruikt. SharePoint 2010 levert zowel de debug- als de productieversies van alle JavaScript bestanden. De debugversies bevatten het

SharePoint 2010 levert zowel de debug- als de productieversies van alle JavaScript bestanden.

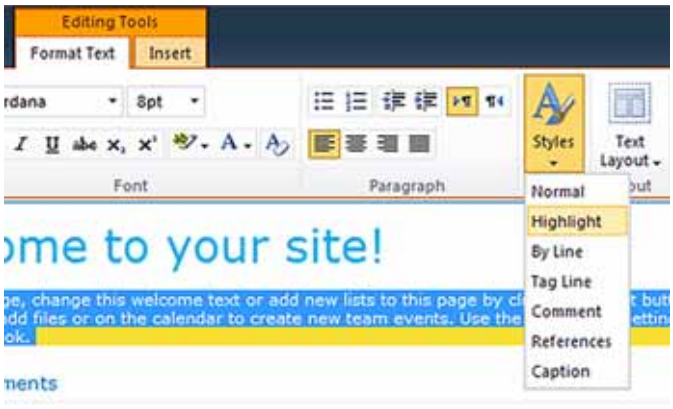
Het SharePoint team heeft een API beschikbaar gesteld waarmee berichten in de Status Bar kunnen worden geplaatst. Je mag met een vaste set kleuren iets meer informatie geven over de status van de status!

woord 'debug' in de naam, bijvoorbeeld SP.Runtime.debug.js. Standaard gebruikt SharePoint de productieversies van JavaScript bestanden maar ontwikkelaars kunnen dit aanpassen door de website in debug mode te plaatsen. In SharePoint 2010 heeft het SharePoint team ook een scheiding tussen 'interne' en 'publieke' JavaScript gemaakt. Alle JavaScript bibliotheken, die door ontwikkelaars kunnen worden gebruikt, zijn met de SP namespace gemarkeerd, bijvoorbeeld: SP.Runtime.js en SP.UI.Dialog.js. Alle andere bestanden zijn voor interne doeleinden bedoeld. Het 'openen' van een gedeelte van het client object model moet ontwikkelaars helpen om eenvoudiger maatwerkapplicaties met een uniforme gebruikerservaring te ontwikkelen.

De Ribbon in SharePoint

Eén van de meest opvallende aanpassingen binnen het Fluency concept is de implementatie van de Ribbon. Toen de eerste geruchten over het gebruiken van de Ribbon in SharePoint bekend werden, hadden velen daar twijfels over. Elk internet browser heeft een menubalk met opties. Het opnemen van 'nog een balk' zou niet alleen verwarrend zijn maar ook kostbare ruimte op het scherm wegnemen. Daarnaast is ribbon een complexe control, die uit veel HTML, CSS bestaat en veel afbeeldingen gebruikt. Sceptici beweerden dat de Ribbon de omvang van de pagina zou verhogen, waardoor de gehele pagina langzamer zou laden. Een alternatief hiervoor zou een Silverlight implementatie kunnen zijn, maar die zou dan niet aan de toegankelijkheidseisen voldoen. De implementatie van de HTML Ribbon leek een enorme uitdaging te zijn...

Tijdens de ontwikkeling van SharePoint 2010 is er veel tijd aan het ontwerpen van de Ribbon voor het web besteed. Enerzijds moest de gebruikerservarng vergelijkbaar zijn met de Office applicaties. De ribbon moest ook aanpasbaar zijn, aan webstandaarden voldoen en natuurlijk geen performance bottleneck worden.Zeker in het licht van geavanceerde functies als in-place galleries en live preview was het geen eenvoudige klus.



FIGUUR 5: LIVE PREVIEW

Voor ontwikkelaars betekent de introductie van de Ribbon een aantal dingen. Allereerst een gemeenschappelijke plek om je commando's te tonen, met zodoende één ontwikkelmodel. Dit ontwikkelmodel bestaat net als bij het oudere broertje uit twee delen. Ten eerste maak je in XML definities aan: structuren die tabs, groepen en controls beschrijven. Natuurlijk localizeerbaar. Daarnaast kan je zowel in de XML als daarbuiten JavaScript gebruiken voor het dynamisch maken van de content. Met de XML kan je niet alleen nieuwe controls toevoegen maar ook bestaande items aanpassen of verwijderen. Genoeg flexibiliteit om een rijke gebruikerservaring te bieden. Qua geavanceerde features is er ook meer dan genoeg te doen. Met behulp van JavaScript kunnen ontwikkelaars zelfs nieuwe typen controls aan de Ribbon toevoegen en het is mogelijk om zowel server-als client-side de Ribbon te renderen.

Sticky berichten

Zoals bij iedere applicatie is het ook voor apps die in SharePoint hun werk doen belangrijk om de gebruiker op de hoogte te houden van de voortgang van het werk. Voor diegene die met SharePoint 2007 hebben gewerkt zal de Status Bar een bekende zijn. Het is een control uit SharePoint 2007 dat binnen Publishing Sites werd gebruikt. Het concept van de Status Bar is niet gewijzigd, maar nu kan er binnen maatwerk applicaties eenvoudig gebruik van worden gemaakt. Het SharePoint team heeft daartoe een API beschikbaar gesteld waarmee berichten in de Status Bar kunnen worden geplaatst. Eenvoudig genoeg, en daarbij mag je met een vaste set kleuren iets meer informatie geven over de status van de status! Groen is goed en rood laat zich natuurlijk wel raden. Daarnaast ondersteunt de Status Bar HTML opmaak voor het tonen van berichten.

```
<script type="text/javascript">
var msgId;
function changeStatus() {
  msgId = SP.UI.Status.addStatus('My Status:', 'Hello World from
  changeStatus function', true);
     SP.UI.Status.setStatusPriColor(msgId, 'red');
}
</script>
<a href="#" onclick="changeStatus(); return false;">Change
Status</a>
```

CODE VOORBEELD: STATUS BAR

FIGUUR 6: STATUS BAR

Een ander component dat de communicatie met gebruikers ondersteunt is de Notification Area. Terwijl de Status Bar bedoeld is voor het tonen van permanente berichten is de Notification Area bedoeld voor het tonen van snelle feedback tijdens het uitvoeren van taken. Korte berichten tijdens het valideren van een pagina of tijdens het ophalen van data. Standaard wordt een bericht in de Notification Area drie seconden getoond. Daarnaast is het ook

Giraffe IT is een gespecialiseerde Microsoft dienstverlener met aantoonbare kennis en ervaring op het gebied van SharePoint en integratie technologie. Wij staan voor end-to-end oplossingen waarbij het Microsoft Platform optimaal wordt benut. Onze mensen zijn stuk voor stuk specialisten op hun vakgebied, weten waar ze over praten en doen wat ze zeggen. Resultaatverplichting is voor ons dus de gewoonste zaak van de wereld. **Giraffe, geen twijfel mogelijk!**

NIKKELSTRAAT 3H 2984 AM RIDDERKERK T: 0180-425700 F: 0180-423300 E: INFO@GIRAFFE.NL I: WWW.GIRAFFE.NL

GIRAFFE

MAATWERK OPLOSSINGEN VAN EENZELFDE RIJKE INTERACTIE ALS MET SHAREPOINT 2010 OUT OF THE BOX...

mogelijk om 'sticky' berichten te maken die worden getoond zolang een taak wordt uitgevoerd. Ontwikkelaars zijn dan zelf verantwoordelijk voor het verbergen van de Notification Area zodra de taak voltooid is.Net als de Status Bar, ondersteunt de Notification Area HTML opmaak voor berichten.

```
<script type="text/javascript">
var nid;
function notifyInProgress() {
  nid = SP.UI.Notify.addNotification("Action in progress...",
  true);
}

function notifyDone() {
  if (nid != null) {
    SP.UI.Notify.removeNotification(nid);
    nid = null;
  }

  SP.UI.Notify.addNotification("Done!", false);
}
</script>
<a href="#" onclick="notifyInProgress(); return false;">Run
Forrest!</a>
<a href="#" onclick="notifyDone(); return false;">Done!</a>
```

CODE VOORBEELD: DIALOGS

Modale vensters met het Dialog Platform

Een laatste stukje fraaiheid aan de client kant is het dialog platform dat het mogelijk maakt om cross-browser dialogen te tonen. Van oudsher had alleen Internet Explorer een showModalDialog functie, weinig portabel dus. Vanuit de markt is het concept geboren om met HTML overlays te werken. Geen echte dialoog in de vorm van een Windows Form dus, maar het geeft wel hetzelfde gevoel. Een dialoog kan zowel een bepaald HTML element uit de bron pagina tonen, of een complete aparte pagina die wordt opgegeven met behulp van een URL. Met behulp van een script worden acties gekoppeld aan de knoppen en is het mogelijk om data door te geven aan de code die het dialoog heeft getoond.

```
var options = SP.UI.$create_DialogOptions();
options.url = SP.Utilities.UrlBuilder.urlCombine(
  SP.PageContextInfo.get_webServerRelativeUrl(),
  '_layouts/ClientUIDemo/DialogPage.aspx');
options.width = 400;
options.height = 300;
options.dialogReturnValueCallback = Function.createDelegate(this,
...});
var dialog = SP.UI.ModalDialog.showModalDialog(options);
```

Conclusie

Een breed scala aan nieuwe features maken het werken met SharePoint eenvoudiger dan ooit te voren. Voor gebruikers een prettigere manier van werken, en voor de ontwikkelaar meer standaard componenten en niet langer de HTML-hell die SharePoint 2007 nog typeerde. Met de komst van een rijk programmeermodel aan de client kant is het ook prima mogelijk maatwerk oplossingen te voorzien van eenzelfde rijke interactie als wat SharePoint 2010 out of the box biedt.

Waldek Mastykarz, (SharePoint Server MVP) werkt bij Imtech ICT Integrated Solutions. Hij blogt op http://blog.mastykarz.nl

Wouter van Vugt, (VSTO MVP) maakt deel uit van een team dat door de Microsoft is gevraagd om SharePoint 2010 training te ontwikkelen en te doceren.