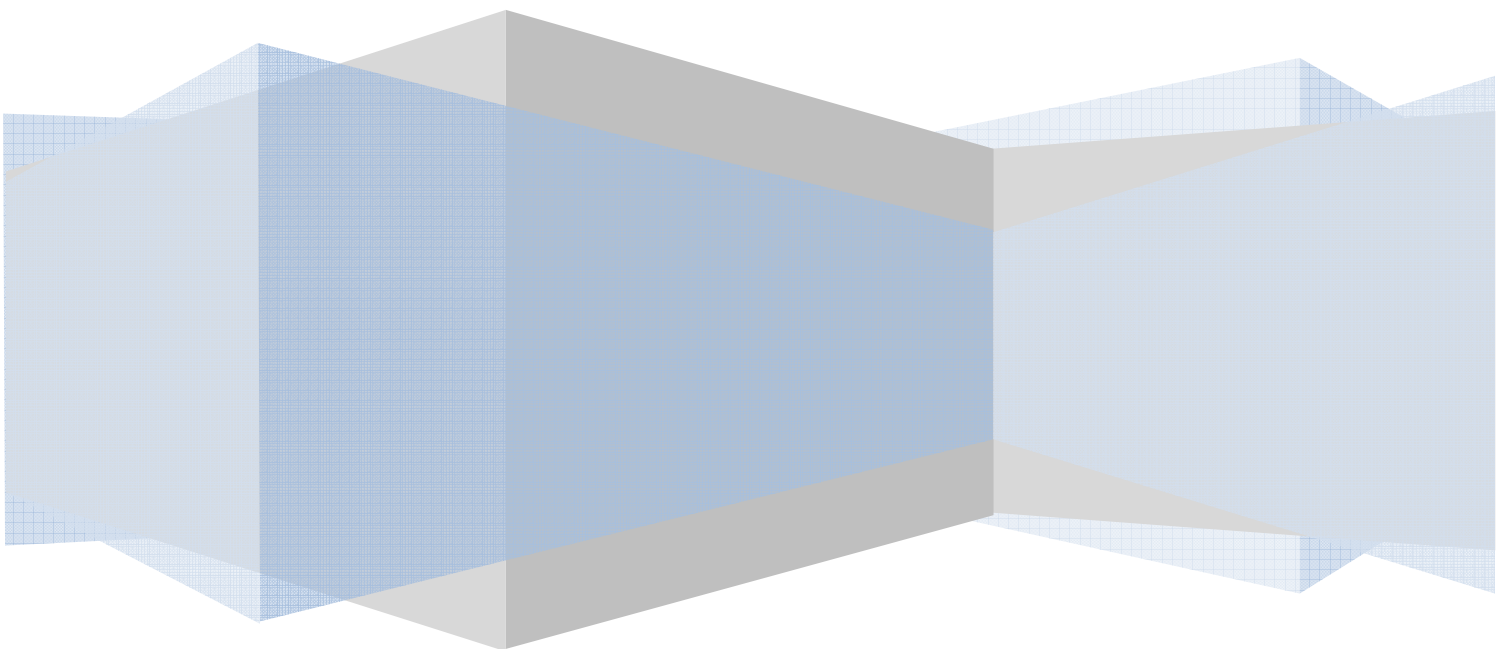


Web 2.0

De doorontwikkeling van het Internet

Ruud Ros, Sander Klijn & Herman van der Meulen



Inleiding

De afgelopen jaren is het internet steeds interactiever geworden. Was het vroeger zo dat je louter informatie op een pagina bekeek. Tegenwoordig kun je volledig interactief op een pagina werken. De doorontwikkeling het internet wordt ook wel Web 2.0 genoemd. Met de term Web 2.0 wordt bedoeld dat we in een nieuwe fase van het internet zitten. Wij zullen in dit essay enkele onderdelen van Web 2.0 bespreken. Te noemen; Participation en Usability (Ruud Ros), Design & CSS (Sander Klijn) en Ajax & Ruby on Rails (Herman van der Meulen)

Participation en Usability

Bij Web 2.0 is er in verhouding met Web 1.0 veel meer participation en usability. Bij Web 1.0 was het meer statisch en presenteren, bij Web 2.0 zijn er meer dingen mogelijk.

Participation

Met participation wordt bedoeld dat de gebruiker meedoet met de webpagina. Forums zijn daarom een goed voorbeeld van participation; iemand plaatst een stukje of stelling en anderen kunnen dat lezen en hun reactie daarop plaatsen, ook kunnen mensen reageren op reacties. Een ander goed voorbeeld is natuurlijk wikipedia. Iedereen kan daar informatie over de meest uiteenlopende dingen vragen, maar iedereen kan ook informatie toevoegen of wijzigen op die site. Het wordt natuurlijk wel gecontroleerd dat er geen onzin op gezet wordt, maar in principe is iedereen vrij om zijn eigen informatie op wikipedia te zetten.

Usability

Met usability wordt bedoeld dat de webpagina gebruiksvriendelijk is en in veel gevallen dat de gebruiker de site kan inrichten op zijn eigen manier. Een goed voorbeeld daarvan is igoogel; bij igoogel kan de gebruiker verschillende vakken met informatie hebben staan: nieuws, weer, fun & games, enz... Als de gebruiker voor het eerst op igoogel komt, staat alles op een standaard plaats. Maar als de gebruiker andere dingen die eerst onderaan de webpagina staan interessanter vindt dan de dingen bovenaan de webpagina staan, kan hij de velden verslepen en al die dingen op zijn eigen manier inrichten. Een ander voorbeeld van usability is maps.google; maps.google was een van de eerste sites waarbij je met je muis de kaart beet kon pakken en verslepen, terwijl de webpagina de kaart automatisch update. Voorheen moest je op een pijltje klikken en duurde het een eeuwigheid voor de nieuwe kaart geladen was en kwam je er vervolgens achter dat je tóch op het vorige gedeelte van de kaart moest zijn. Dit probleem heb je niet bij bijvoorbeeld maps.google. Usability speelt dus in op hoe de gemiddelde gebruiker gevoelsmatig met de webpagina om zou gaan als er geen beperkingen waren.

CSS & Design

Geschiedenis

Sinds het begin van de internet-browsers heeft elke zijn eigen opmaaktaal, oftewel stylesheet, gehad. Deze taal zorgde ervoor dat een browser de opmaak van een pagina correct weergeeft. Maar omdat elke pagina een ander had, moest de pagina voor elke browser aangepast worden. In 1994 kwam de Noor Håkon Wium Lie samen met de Nederlander Bert Bos op het idee om één standaard voor deze stylesheets te ontwikkelen. Hij werkte samen met Bert Bos omdat hij al een tijd werkte aan zijn eigen browser, Argo genaamd, die ook gebruik maakte van stylesheets.

In 1995 opperde ze het idee nogmaals op een internet conferentie in Chicago. In die tijd was de W3C orgaan in oprichting, zij begonnen een project om tot één standaard taal te komen. In 1996 werd de eerste versie gepubliceerd, CSS 1 genaamd. In 1997 werd er verder gewerkt aan de uitbreiding van deze stylesheet standaard. In 1998 werd CSS 2 gepubliceerd. Deze is later nog uitgebreid naar CSS 2.1. Op dit moment wordt er nog aan versie 3 gewerkt, deze is al ver in de ontwikkeling, maar nog geen officiële standaard.

In 1996 mocht de standaard voor CSS dan wel al ontwikkeld zijn, de browsers ondersteunde het nog niet. Het duurde tot het jaar 2000 voordat er browser kwam die deze standaard volledig ondersteunde. Dit was Microsoft Internet Explorer 5.0. Met de komst van Internet Explorer 7 wordt CSS 2.1 pas wat beter ondersteund. In versie 8, waarvan de beta reeds beschikbaar is, zal CSS volledig ondersteund worden!

Informatie

CSS staat voor **C**ascading **S**tyle **S**heet. CSS bevat de opmaak van een webpagina. Wanneer je CSS goed gebruikt, dan kan je de opmaak van een pagina volledig scheiden van de inhoud. Dit kan een hoop werk schelen wanneer je bijvoorbeeld een site een nieuwe lay-out wilt geven. Het is ook mogelijk om je CSS in een los bestand op te slaan, dit scheelt een hoop dubbele informatie. Nog een bijkomend plus punt is dat je in één keer meerdere pagina's kan wijzigen. Het is de bedoeling dat alle browsers uiteindelijk de zelfde stylesheets gaan gebruiken. Op deze manier zien alle pagina's er in alle browsers het zelfde uit!

Voorbeeld CSS bestand

Hieronder een voorbeeld van een bestand met en zonder CSS toepassing. Bij de eerste is de bron redelijk groot. Dit is een simpel voorbeeld, maar dit kan veel complexer worden bij meer informatie. Bij de bron met CSS geeft het in principe de zelfde output voor de gebruiker. Echter, de bron is hier vele malen kleiner. Alle informatie met betrekking tot de opmaak staat hier in het losse stylesheet bestand. Hierdoor blijft de bron van de pagina ook een stuk overzichtelijker

Bron (zonder CSS)

```
<table>
  <tr>
    <td><font face="Times" color="blue">< b>Cel 1</b></font></td>
    <td><font face="Times" color="blue"><b>Cel 2</b></font></td>
  </tr>
  <tr>
    <td colspan="2"><font face="Times" size="1">Cel 3</font></td>
  </tr>
</table>
```

Bron (met CSS)

```
<div id="div1">Div 1</div>
<div id="div2">Div 2</div>
<div id="div3">Div 3</div>
```

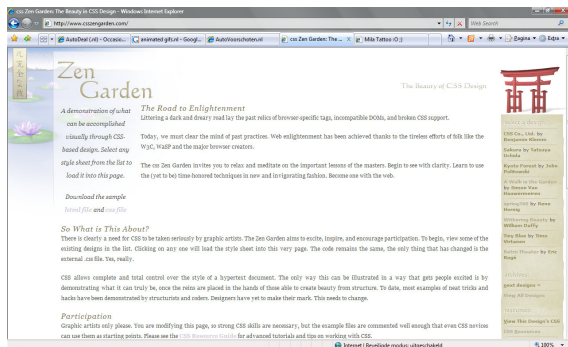
Stylesheet (CSS)

```
.div1{ font-family:times; Color:blue; top:0px; left:0px; width:100px; height:30px; }
.div2{ font-family:times; Color:blue; top:0px; left:100px; width:100px; height:30px; }
```

Voorbeelden

Een goed voorbeeld van een pagina waarin goed gebruik gemaakt is van CSS is 'CSS Zengarden'. Dit is een pagina die volledig met CSS is gebouwd. De volledige lay-out van deze pagina kan gewijzigd worden, zonder dat de bron wijzigt. Alleen het CSS bestand wordt aangepast.

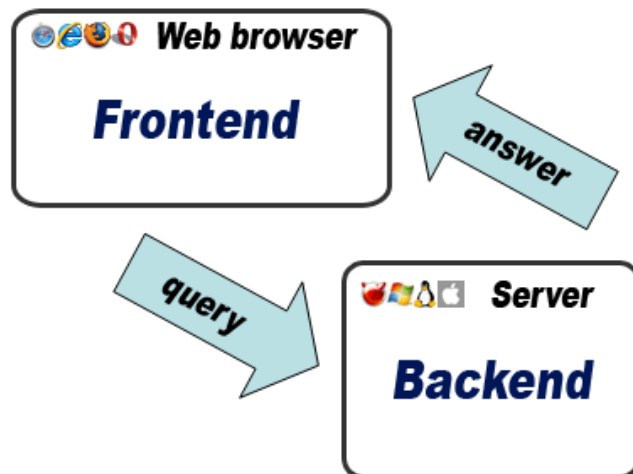
Hieronder een voorbeeld:



AJAX

Sinds het ontstaan van het wereldwijde web hebben de betrokken technieken nooit stil gestaan. Er zijn weinig ontwikkelingen op technisch gebied die met de snelheid als deze gaan. Was men in de beginjaren blij met het fragmentarisch delen van een wetenschappelijke database, tegenwoordig lijkt er bijna geen limiet aan de mogelijkheden van het internet. Nieuwe technieken laten ook vaak nieuwe presentatie mogelijkheden toe, wat de gebruiksvriendelijkheid van het web kan vergroten.

Een van de laatste technieken die vooral voor deze laatste reden is 'ontwikkeld' is AJAX. Ontwikkeld tussen quotes, omdat de gebruikte techniek al wat langer bestaat en ook al gebruikt werd, bijvoorbeeld door Google. Met deze techniek kan een bepaald deel van een website dynamisch vernieuwd worden, zonder dat de hele website opnieuw hoeft te laden. AJAX staat voor "Asynchronous Javascript And XML". Dat is eigenlijk een 'fancy' woord voor XMLHttpRequests. Een XMLHttpRequest wordt als object aangemaakt in Javascript. Bij een gekozen 'event' kan in hetzelfde script informatie in dit object geplaatst worden, verstuurd worden naar de server en een andere waarde terug krijgen van de server. Dit gebeurt allemaal nog zonder dat de gebruiker er iets van merkt. Als het vernieuwde object ontvangen is, kan het onderdeel van de website vernieuwd worden.



AJAX is dus officieel gebaseerd op 2 talen, namelijk Javascript en XML. Javascript is een veelgebruikte client-side scripttaal en XML is een steeds vaker gebruikte gestructureerde data/beschrijving taal. De Javascript wordt in AJAX gebruikt om de website aan de client-side goed te laten zien en bij eventuele events, zoals muisklikken of op een knop drukken, de informatie naar de server te versturen en weer te ontvangen, waarna een gedeelte van de website aangepast kan worden. Dan is er nog

een taal nodig om de te verwerken data in te structureren, zodat zowel client-side scripttalen als verschillende server-side talen ermee om kunnen gaan. Voor server-side talen is de keuze trouwens zeer ruim. Voorbeelden zijn vooral PHP en ASP, maar ook JSP of verschillende CGI opties.

Voor de dataoverdracht wordt dus het XML 'formaat' voorgesteld, maar is niet verplicht. Per toepassing zijn er goede alternatieven te gebruiken. Als je heel simpele onderdelen of stukken tekst van een website wilt laten verwerken, is het meestal het makkelijkst om gewoon ruwe tekst te versturen. Dat is zeer simpel te verwerken. Nadeel is dat je geen structuur en geen opmaak mee kunt geven en dat de hoeveelheid data redelijk beperkt moet blijven. Om snel wat eenvoudige structuur en opmaak mee te geven kun je stukken HTML code meegeven. Als laatste is er nog de mogelijkheid om complete Javascript objecten mee te sturen. Dit wordt vaak gebruikt als de client-side zeer veel ingewikkelde Javascript bevat, zodat er direct met het object gewerkt kan worden. Het verwerken van een javascript object aan de server-side heeft wat beperkingen en is relatief ingewikkeld. Deze

techniek wordt dan ook het meeste gebruikt bij grootschalige webapplicaties. XML is uiteindelijk de meest universele, alles omvattende data structuur waar steeds meer mee gewerkt wordt en zal dus ook veel gebruikt gaan worden in AJAX toepassingen.

Mooie voorbeelden van AJAX toepassingen zijn bijvoorbeeld "Google Maps", en "Meebo.com".

Ruby on Rails

Ruby on Rails, kort RoR of gewoon Rails, is een open source webapplicatie framework. De naam bestaat uit 2 delen. Ruby is de gebruikte scripttaal, Rails is het eigenlijke framework. We bespreken deze techniek kort, omdat het een relatief zeer nieuwe techniek is, nog niet volledig doorontwikkeld is en ook nog niet volledig geaccepteerd is.

Wat is Ruby?



Ruby bestaat al wat langer, het is officieel uitgebracht in 1995 door Yukihiro Matsumoto. De naam is geen afkorting maar een woordspeling op de taal Perl. Ruby onderscheidt zich op bepaalde punten. Zo is de taal volledig objectgeoriënteerd, zelfs een INT is een object

met verschillende methoden. Ruby heeft een zeer simpele leesbare syntax, zoals te zien in het voorbeeld hieronder. Ruby is platform onafhankelijk en werkt dus op bijvoorbeeld Windows, Macintosh, Linux, etc. Ruby is open source, wat betekent dat je het gratis mag gebruiken onder een speciale licentie en zelfs naar eigen wensen aan zou kunnen passen. Hieronder een kort voorbeeld van de syntax van Ruby:

```
class Person
  def initialize(name, age)
    @name, @age = name, age
  end
  def <=>(person) # Comparison operator for sorting
    @age <=> person.age
  end
  def to_s
    "#@name #@age"
  end
  attr_reader :name, :age
end

group = [
  Person.new("Jon", 20),
  Person.new("Marcus", 63),
  Person.new("Ash", 16)
]

puts group.sort.reverse
```

In bovenstaand voorbeeld wordt er een klasse "Person" aangemaakt, met de attributen "name" en "age" en de methoden "initialize" en "to_s", verder is er nog een vergelijkingsmethode. De vergelijkingsmethode vertelt op welk of welke attributen een object gesorteerd moet worden

Na de klasse aangemaakt te hebben kan je objecten aanmaken van de klasse. Er worden in het voorbeeld drie personen aangemaakt, elk dus met een naam en leeftijd en worden in de array "Group" gezet.

Uiteindelijk kan je simpelweg "puts group" zeggen om alle personen naar de output te sturen. In het voorbeeld wordt de output heel eenvoudig achterwaarts gesorteerd.

Wat is Rails?

Rails is een gratis webapplicatie framework met als doel: het ontwikkelen van database driven webapplicaties te versnellen en vereenvoudigen. Rails gebruikt altijd het Model, View Controller principe. Hierdoor blijft je project zeer overzichtelijk. Het Model is meestal de database, de Views bevatten meestal de HTML en CSS en de Controllers bevatten de logica om alles te koppelen en werkend te krijgen.



Rails kent 2 belangrijke principes: CoC (Convention over Configuration) en DRY (Don't Repeat Yourself). Je bouwt super snel de basis van je applicatie op en wijzigt zonnodig wat je anders wilt hebben. Je specificeert eigenlijk alleen de bijzonderheden. En natuurlijk het uiterlijk. Als je in je model voor een weblog hebt gespecificeerd dat een post een naam, inhoud en datum heeft, worden er gelijk standaard benodigde controllers en views aangemaakt (scaffolding) waardoor ge meteen complete berichten kan posten, wijzigen en verwijderen. Rails herkent zelf dat de tabel met de naam "posts" meerdere eenheden van "post" bevat. Als je een tabel "comments" aanmaakt en je koppelt deze met een foreign key aan "post", weet Rails dat elk comment aan één post kan zitten. En zo zitten er nog veel meer slimme handigheidjes in die de ontwikkeling van webapplicaties enorm versnellen.

En mocht je dezelfde handigheidjes en voordelen wilt gebruiken maar dan met PHP? De ontwikkelaars van het framework "CakePHP" zijn echte naapers van RoR, lopen misschien wat achter, maar de overstap is natuurlijk een stuk makkelijker. Al is Ruby een heel mooie taal om te leren.

Meer informatie, mooie voorbeelden en tutorials zijn te vinden op rubyonrails.org. Zien is geloven!