JavaScript

door

Leerkracht: Dany Pinoy

woensdag 28 november 2012

Versie: September 2012

Website: http://webdesign.pindanet.be

SNT Volwassenenonderwijs overdag en 's avonds

Informatica Talen Nederlands voor anderstaligen
Bedrijfsbeheer Koken Mode Vlaamse Gebarentaal

haal hier je bagage
SNT
www.snt.be

Arsenaalstraat 4
8000 Brugge
050 33 76 69
adm@snt.be

BRUGGE

U gaat akkoord met ...

Deze cursus wordt u aangeboden door de Vrienden van de SNT, vzw, in samenwerking met de auteur(s). Als u deze cursus volgt, betekent dit dat u akkoord gaat met het volgende:

- De auteur(s) van deze cursus heeft (hebben) alles in het werk gesteld om een juiste werkwijze voor te stellen en eventuele bijhorende oefenprogramma's zowel geprint of digitaal in staat van goede werking en virusvrij te houden.
- Geen enkel geheel of gedeelte van software aanwezig op de SNT-schoolcomputers mag in enige vorm of op enige wijze worden gekopieerd of opgeslagen naar/op enig welke gegevensdrager zonder uitdrukkelijke voorafgaande toestemming van de onderwijzende SNT-informaticaleerkracht bevoegd voor deze cursus.
- Geen enkel geheel of gedeelte van software mag in enige vorm of op enige wijze worden gekopieerd of opgeslagen naar/op enig welke gegevensdrager van de SNT-schoolcomputers zonder uitdrukkelijke voorafgaande toestemming van de onderwijzende SNT-informaticaleerkracht bevoegd voor deze cursus. Het gebruik van de SNT-internet toegang wordt uitsluitend toegelaten met betrekking tot de theorie of oefeningen van voorliggende cursus en enkel zoals door de leerkracht aangegeven en afgebakend. Enkel legale internet downloads/uploads die gebeuren op vraag van de leerkracht in het kader van deze lessen horend bij deze cursus zijn toegelaten. Meervoudig internetmisbruik leidt tot uitsluiting.
- De vzw, het Centrum voor Volwassenenonderwijs Stedelijke Nijverheids- en Taalleergangen, de auteur(s), de Inrichtende Macht, in casu Stad Brugge, zijn geenszins aansprakelijk in geval de gebruiker van deze cursus en/of eventueel bijhorend oefenmateriaal schade zou lijden aan zijn computerapparatuur of programmatuur die voortvloeit uit enige fout die in het aangeboden materiaal zou kunnen voorkomen.
-

Evaluaties aan de SNT

Voor de evaluatie van uw leervorderingen gebruikt SNT een systeem van "Permanente Evaluatie".

Permanente Evaluatie betekent dat u punten "verdient" gedurende het semester, terwijl u een oefening maakt tijdens de les of op het leerplatform. Concreet betekent dit dat "het eindexamen" wordt vervangen door één of meerdere taken tijdens de lesweken.

Tijdens een permanent evaluatiemoment controleert de leerkracht of u de leerstof voldoende beheerst en krijgt u een score toegewezen (0 tem 3).

Scorewijzer

scorewijzer schaal [3210]
3 – AA: heel sterke prestatie
2 – A: sterke prestatie, streefniveau
1 – B: middelmatig prestatie
0 – C: te zwakke prestatie, leerdoelen niet bereikt

Op het einde van het semester wordt de totale gemiddelde score berekend en via een transformatieschaal omgezet naar een percentage.

Transformatietabel

Bovenste getal is de gemiddelde behaalde score volgens de scorewijzer.
Onderste getal is de gemiddelde score getransformeerd naar een percentage op 100.

0.0	0.2	0.4	0.6	0.8	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0	2.2	2.4	2.6	2.8	3.0
%	30	35	40	45	50	52	54	56	58	60	62	64	66	68	70	75	80	85	90	100%

U behaalde	
Meer dan 70%	heel sterke prestatie
70%	sterke prestatie, streefniveau
Meer dan 50% en minder dan 70%	middelmatige prestatie
Minder dan 50%	te zwakke prestatie, leerdoelen niet bereikt

Indien u nog vragen hebt, aarzel niet om contact op te nemen met uw SNT-leerkracht.

Inhoud

BRUGGE

JavaScript Oriëntatie

Wat is JavaScript?

De basis van elke webpagina is een tekstbestand met HTML code. Deze HTML code verzorgt de structuur en inhoud van de webpagina (cursus Dreamweaver).

CSS wordt gebruikt om webpagina's op te maken (cursus Dreamweaver).

JavaScript is een scripttaal die veel gebruikt wordt om webpagina's interactief te maken en webapplicaties te ontwikkelen (deze cursus).

PHP zorgt voor interactie via de webserver (cursus PHP).

De syntaxis van JavaScript vertoont overeenkomsten met de programmeertaal Java. Omdat beide talen het meest zichtbaar zijn op en rond de browser, maar vooral door de naamgeving, worden ze vaak met elkaar verward. De gelijkenis houdt daar echter op, want JavaScript heeft inhoudelijk meer gemeen met functionele programmeertalen, het biedt prototype-gebaseerde overerving en niet, zoals Java en de meeste objectgeoriënteerde talen, klasse-gebaseerde overerving.

Client-side

In deze toepassing wordt JavaScript vooral gebruikt in interactieve webpagina's. Net als bij andere scripttalen is er een interpreter nodig om de geprogrammeerde opdrachten uit te voeren. De meeste moderne browsers beschikken over een eigen interpreter voor JavaScript. Het besturingssysteem Windows heeft een ingebouwde interpreter, het bestand jscript.dll.

Ook enkele e-mailprogramma's ondersteunen JavaScript in HTML-berichten.

Server-side

JavaScript kan ook gebruikt worden voor server-side scripting. De webservers van Netscape waren de eerste die deze ondersteuning boden. Maar ook de webserver van Microsoft, IIS, ondersteunt JavaScript in Active Server Pages en ASP.NET. De laatste jaren maakt node.js een grote opgang.

JavaScript-Toolkits

Een JavaScript-framework is een library (bibliotheek) welke een aantal JavaScript-functies en widgets bevat voor het ontwikkelen van webapplicaties, waarbij vaak de nadruk ligt op AJAX.

Omgeving

Het Internet Mediatype of MIME voor JavaScript-code is application/javascript, hoewel het niet-officiële text/javascript vaker wordt gebruikt.

Om JavaScript op te nemen in een webpagina die voldoet aan de standaard voor HTML 4.01, moet het type-attribuut expliciet worden opgegeven in de openingstag:

```
<script type="text/javascript">
// code
</script>
```

In XHTML-documenten houden speciale karakters, zoals "<" (kleiner dan), hun betekenis ook binnen script-elementen (in HTML vervalt die speciale betekenis onder bepaalde voorwaarden). Een script dat zulke karakters bevat, moet daarom als CDATA-sectie gemarkeerd worden. De CDATA-markering zelf wordt dan vaak met "//" in commentaar verstopt om te voorkomen dat er problemen ontstaan met browsers die geen CDATA-secties herkennen.

```
<script type="text/javascript">
<!--//--><![CDATA[//><!--
```

```
// code
//--><!]]>
</script>
```

In HTML5 is bepaald dat JavaScript de standaardtaal is en is het lang-attribuut optioneel, net zoals de CDATA-secties:

```
<script>
    alert('Hallo, wereld!');
</script>
```

Invoegen van een JavaScript

Het invoegen van JavaScript in een webpagina is niet altijd hetzelfde. In totaal zijn er een viertal manieren om een script in een webpagina in te brengen nml.:

- in het head gedeelte van de pagina.
- in het body gedeelte.
- in het head én het body gedeelte.
- en in een extern bestand.

Je script staat altijd in een script tag: <script>code</script>

Scripts in het head gedeelte

Het head gedeelte van een pagina wordt het eerst geladen.

Een script dat je in het head gedeelte zet, is al geladen voordat er code uit het body gedeelte wordt uitgevoerd.

Soms is het vereist om bepaalde JavaScript code in het head gedeelte te zetten, bijvoorbeeld bij het gebruik van functies in JavaScript.

De structuur van een script in het head gedeelte:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5   <title>JavaScript Oriëntatie</title>
6   <script type="text/javascript">
7   tekst = "Welkom in de wereld van JavaScript.";
8   </script>
9   </head>
```

Scripts in het body gedeelte

De structuur van een script in het body gedeelte:

```
10  <body>
11  <script type="text/javascript">
12  alert(tekst);
13  </script>
14  </body>
15  </html>
```

Scripts in het head én het body gedeelte

Er is geen limiet aan het aantal scripts dat je in een pagina mag gebruiken.

Je kan dus zowel JavaScript in het head gedeelte en in het body gedeelte van je pagina zetten.

Het komt vaak voor dat er eerst een aantal functies of variabelen in het head gedeelte worden benoemd met een JavaScript-code, waarna ze door een script in het body gedeelte worden opgeroepen en uitgevoerd.

Een extern script maken

Een extern script is JavaScript code die in een apart bestand staat.

Dit is een bestand met een bestandsnaam eindigend op .js.

Zo'n extern script is vooral handig als je een bepaald script op meerdere pagina's wilt uitvoeren.

Nu kun je gewoon naar het externe bestand verwijzen, zonder dat je de JavaScript code op iedere pagina hoeft in te voeren.

In het externe script staat alleen JavaScript code!

Je mag er dus geen HTML tags inzetten.

Dit is een voorbeeld van de inhoud van een .js bestand:

```
1   // JavaScript Document
2   document.write("Dit is een extern script!");
```

Het script bevat alleen JavaScript code en geen HTML tags.

Om er een JavaScript bestand van te maken, is het nu alleen nog nodig om dit bestand op te slaan als een .js bestand, bijv. extern.js.

Nu kunnen we in de HTML pagina (extern.html) naar het externe script verwijzen door middel van de volgende code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5   <title>Extern JavaScript</title>
6   </head>
7   <body>
8   <p><script type="text/javascript" src="extern.js"></script></p>
9   </body>
10  </html>
```

JavaScript invoegen met Dreamweaver

Hoewel er heel wat open source editors voor webdesign bestaan (NetBeans, Aptana, NotePad++, enz.) wordt Dreamweaver gezien als de meest gebruiksvriendelijke editor.

In de Ontwerp weergave

Werkwijze voor JavaScript in het head gedeelte:

1. Activeer de *Code* weergave.
2. Plaats de cursor juist voor de tag </head>.
3. Klik in het paneel *Invoegen* > *Algemeen* op de opdracht *Script* > *Script*.
4. Indien je een extern script wilt gebruiken, gebruik je de knop *Bladeren* om de *Bron* van het externe JavaScript te bepalen.

   Voor een JavaScript die in het document wordt opgenomen, kun je in het tekstvak *Inhoud* de JavaScript code intypen.

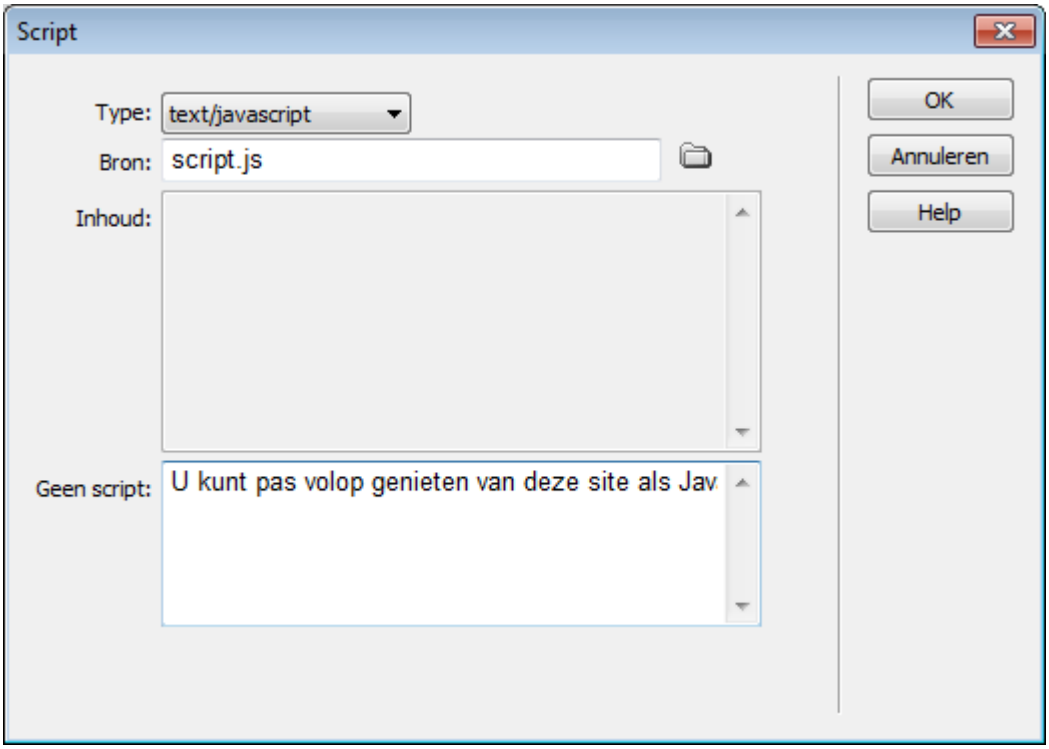
In het tekstvak *Geen script* kun je een tekst (of html code) plaatsen voor mensen die het uitvoeren van Javascript code in hun browser hebben uitgeschakeld.

5. Klik op de knop *OK*.

6. Activeer de Ontwerp weergave.

Werkwijze voor JavaScript in het body gedeelte:

1. Activeer de *Ontwerp* weergave.
2. Plaats de cursor op de plaats waar je de JavaScript code wilt plaatsen.
3. Klik op het paneel *Invoegen* > *Algemeen* op de opdracht *Script* > *Script*.



4. Indien je een extern script wilt gebruiken, gebruik je de knop *Bladeren* om de *Bron* van het externe JavaScript te bepalen.

   Voor een Javascript die in het document wordt opgenomen, kan je in het tekstvak *Inhoud* de JavaScript code intypen.

   In het tekstvak *Geen script* kan je een tekst (of html code) plaatsen voor mensen die het uitvoeren van Javascript code in hun browser hebben uitgeschakeld.

5. Klik op de knop *OK.*

6. Er verschijnt een dialoogvenster met de melding dat je de ingevoerde JavaScript code niet in de Ontwerp weergave zult zien. Deze kan echter wel zichtbaar gemaakt worden via het menu *Weergave* > *Visuele hulpmiddelen* > *Onzichtbare elementen* en het activeren van *Scripts* in de categorie *Onzichtbare elementen* van de opdracht *Voorkeuren* in het menu *Bewerken.* Sluit deze melding.

In de Code weergave

Dit is voor programmeurs de meest voor de hand liggende manier om JavaScript code in te typen.

Dreamweaver gebruikt kleuren en suggesties om je te helpen bij het opsporen van fouten en om efficiënter te werken.

JavaScript uitvoeren en testen

JavaScript wordt gebruikt in webpagina's en worden dus uitgevoerd en getest in een webbrowser.

Daar elke browser een eigen JavaScript interpreter heeft, moet je zoals trouwens elke webpagina de test uitvoeren in verschillende browsers.

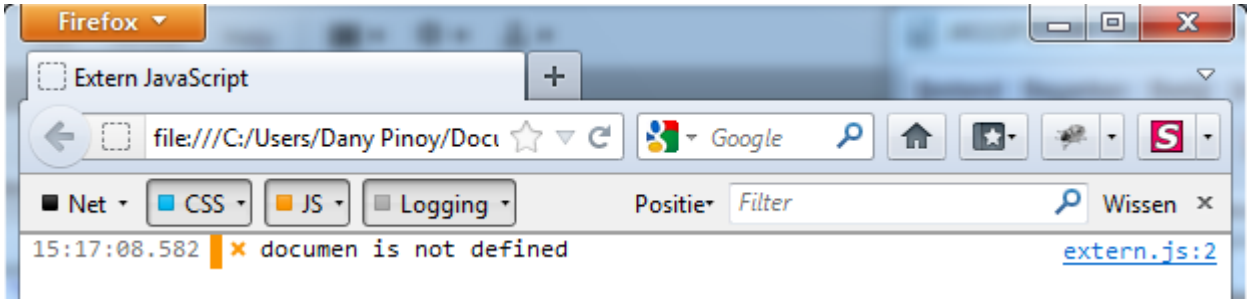
Firefox

Firefox wordt door de beschikbaarheid van hulpmiddelen voor ontwikkelaars veel gebruikt als testplatform voor webpagina's.

Om fouten in Firefox op te sporen, gebruik je in het *Firefox* menu de opdracht *Webontwikkelaar* > *Webconsole*.

Bovenaan in het venster verschijnt een deelvenster met vier knoppen waarmee je meldingen kunt weergeven of verbergen. Om fouten één voor één aan te pakken, schakel je best een paar meldingen uit (begin met het uitschakelen van alle Net meldingen).

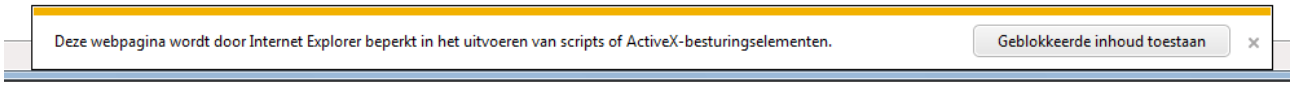
Vernieuw de te testen webpagina.



Zo merk je dat in het bestand extern.js op regel 2 een fout optrad.

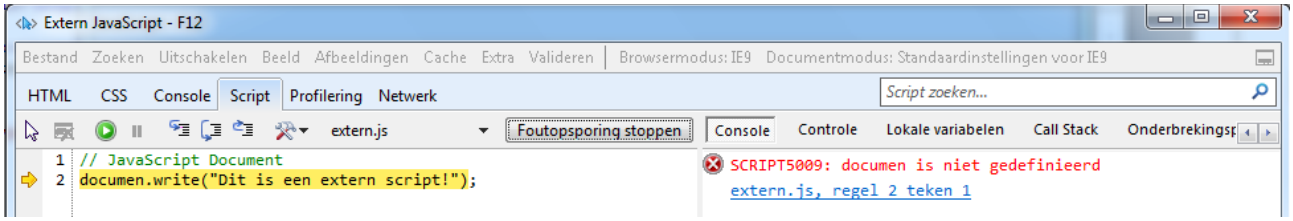
Internet Explorer

Bij het openen van een lokale (vanaf de harde schijf, en dus niet vanaf het internet) webpagina wordt JavaScript standaard geblokkeerd. Om de webpagina correct weer te geven en te testen moet je de JavaScript code activeren door op de knop *Geblokkeerde inhoud toestaan* te klikken.



Druk op de toets F12 om het ontwikkelhulpprogramma in een apart venster te starten.

Activeer het tabblad *Script* en klik op de knop *Foutopsporing starten*.



Ook browsers zoals Safari en Chrome hebben gelijkaardige hulpmiddelen.

JavaScript Initiatie

Weergeven op een HTML pagina

Het volgende voorbeeld toont hoe je aan een HTML pagina (datum.html) een paragraaf met de datum kunt toevoegen:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Datum weergeven</title>
6   </head>
7   <body>
8   <h1>Datum weergeven</h1>
9   <script type="text/javascript">
10  document.write("<p>" + Date() + "</p>");
11  </script>
12  </body>
13  </html>
```

Dit voorbeeld gebruikt de internationale schrijfwijze om de datum weer te geven. Pas het JavaScript als volgt aan om de lokale datum weer te geven:

```
datum = new Date();
document.write("<p>" + datum.toLocaleDateString() + "</p>");
```

De eerste regel slaat de datum met de internationale schrijfwijze op.
De opdracht datum.toLocaleDateString() vormt de internationale datum om tot een datum weergegeven volgens de voorschriften van het land en de taal van de browser.

Opmerking: Probeer in het vervolg document.write() in JavaScript te vermijden. Bij het gebruik van document.write() binnen een functie of nadat de volledige pagina reeds wordt weergegeven, zal de volledige inhoud van de pagina vervangen worden door de nieuwe inhoud (en dus niet toegevoegd worden). De opdracht document.write() is wel de eenvoudigste manier om uitvoer in een webpagina te plaatsen.

HTML elementen aanpassen

Het volgende voorbeeld plaats de datum in een reeds bestaande paragraaf.

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Datum weergeven</title>
6   </head>
7   <body>
8   <h1>Datum weergeven</h1>
9   <p id="datum"></p>
10  <script type="text/javascript">
11  datum = new Date();
12  document.getElementById("datum").innerHTML =
     datum.toLocaleDateString();
13  </script>
14  </body>
15  </html>
```

Om HTML elementen te manipuleren gebruikt JavaScript de DOM methode getElementById(). Deze methode zorgt voor toegang tot het element met het opgegeven id. De eigenschap innerHTML zorgt ervoor dat de inhoud van het element vervangen wordt door de datum.

Let op het feit dat het JavaScript na het <p> element staat. Zo wordt het JavaScript pas uitgevoerd na het aanmaken van het <p> element. M.a.w. het <p> element moet bestaan om het te kunnen aanpassen.

Functies en events (gebeurtenissen)

JavaScript wordt in een HTML pagina uitgevoerd bij het laden van de pagina. Soms willen we JavaScript pas uitvoeren bij een gebeurtenis (bijvoorbeeld bij het klikken op een knop). In zo'n geval plaatsen we de code in een functie.

Events (gebeurtenissen) worden meestal in combinatie met functies gebruikt (een groep opdrachten starten bij een bepaalde gebeurtenis).

Het voorbeeld (event.html) toont het uitvoeren van een functie bij het klikken op een knop:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Reactie op gebeurtenis</title>
6   <script type="text/javascript">
7   function datumWeergeven(){
8     datum = new Date();
9     document.getElementById("datum").innerHTML =
     datum.toLocaleDateString();
10  }
11  </script>
12  </head>
13  <body>
14  <h1>Datum weergeven</h1>
15  <input type="button" onClick="datumWeergeven();" value="Datum
     weergeven" />
16  <p id="datum"></p>
17  </body>
18  </html>
```

Hoofdlettergevoelig

De naam van de functie datumWeergeven bevat een hoofdletter. In tegenstelling tot HTML is JavaScript **hoofdlettergevoelig** – m.a.w. let op het gebruik van hoofdletters bij JavaScript opdrachten, variabelen, objecten en functies.

Opdrachten (statements)

JavaScript opdrachten (statements) voeren bepaalde opdrachten uit. De opdracht

```
document.write("Welkom allemaal");
```

plaatst te tekst Welkom allemaal op de webpagina in de browser.

Meestal worden uitvoerbare opdrachten afgesloten met een puntkomma. De meeste JavaScript programmeurs vinden dit een goede programmeertechniek. Vandaar dat de meeste voorbeelden op het internet dit ook gebruiken.

Volgens de JavaScript standaard is het gebruik van een puntkomma optioneel en moet de browser het einde van de regel als het einde van de opdracht interpreteren.

Opmerking: met behulp van puntkomma's kan je meerdere opdrachten op één regel plaatsen.

Code

JavaScript code (of kortweg JavaScript) is een opeenvolging van JavaScript opdrachten.

Elke opdracht wordt door de browser in de volgorde van voorkomen uitgevoerd.

De volgende JavaScript code schrijft een Kop en twee paragrafen naar de webpagina:

```
<script type="text/javascript">
document.write("<h1>Dit is een Kop 1</h1>");
document.write("<p>Dit is een paragraaf.</p>");
document.write("<p>En nog een paragraaf.</p>");
</script>
```

Blok

JavaScript opdrachten kan je in blokken (blocks) groeperen.

Blokken worden ingesloten door {}.

Het doel van blokken is om een reeks opdrachten als één geheel uit te voeren.

```
function datumWeergeven(){
  datum = new Date();
  document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
}
```

De functie datumWeergeven is een blok met twee opdrachten die steeds samen uitgevoerd worden (namelijk bij het klikken op de knop).

Commentaar

JavaScript commentaar gebruikt je om code leesbaarder te maken.

Commentaarregel

Een enkele commentaarregel start met //.

```
function datumWeergeven(){
  // internationale datum opvragen
  datum = new Date();
  document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
}
```

Meerdere regels commentaar

Het commentaarblok begint met /* en eindigt met */

```
function datumWeergeven(){
  // internationale datum opvragen
  datum = new Date();
  /*
  De inhoud van het element met het id datum krijgt als inhoud
  de datum weergeven naar de normen van het land en de taal.
  */
  document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
}
```

Commentaar gebruiken om opdrachten uit te schakelen

Commentaar wordt ook gebruikt om een opdracht voorlopig uit te schakelen (bijvoorbeeld om fouten op te sporen).

```
function datumWeergeven(){
  // internationale datum opvragen
  datum = new Date();
  // alert(datum);
  /*
  De inhoud van het element met het id datum krijgt als inhoud
  de datum weergeven naar de normen van het land en de taal.
  */
  document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
}
```

Vanzelfsprekend kan je met /* en */ een volledige blok met opdrachten uitschakelen.

Commentaar op het einde van een regel

Alles wat volgt op de tekens // tot het einde van de regel wordt beschouwd als commentaar.

```
function datumWeergeven(){ // uitvoeren bij het klikken op de knop
  // internationale datum opvragen
  datum = new Date();
  /*
  De inhoud van het element met het id datum krijgt als inhoud
  de datum weergeven naar de normen van het land en de taal.
  */
  document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
}
```

Soms is JavaScript in de browser uitgeschakeld

Hoewel het zelden voorkomt, wordt om veiligheidsredenen of om de aantrekkelijkheid van webpagina's fel te verminderen JavaScript in de browser uitgeschakeld. Zo ingestelde browsers voeren de JavaScript niet uit maar tonen de code als tekst op de webpagina.

Om dit te voorkomen, moet je volgens de JavaScript standaard JavaScript voor HTML verbergen met behulp van HTML commentaar.

Plaats juist voor de eerste JavaScript opdracht de HTML commentaar tag <!-- en na de laatste JavaScript opdracht sluit je de HTML commentaar tag met -->.

```
<body>
<h1>Datum weergeven</h1>
<p id="datum"></p>
<script type="text/javascript">
<!--
datum = new Date();
document.getElementById("datum").innerHTML =
datum.toLocaleDateString();
//-->
</script>
</body>
```

De tekens // voor de --> verhinderen dat JavaScript --> ziet als een opdracht.

Opdrachten

Datum en/of tijd weergeven

Pas de pagina event.html aan zodat er drie knoppen staan. Elke knop start bij een klik erop een eigen functie.

De eerste knop bestaat reeds en start de functie datumWeergeven().

Schrijf een tweede functie om de lokale tijd weer te geven. Deze functie lijkt als twee druppels water op de eerste functie, maar gebruikt in de plaats van datum.toLocaleDateString() om de datum weer te geven, datum.toLocaleTimeString() om de tijd weer te geven.

Maak een knop *Tijd weergeven* waarbij de pas geschreven functie bij het aanklikken wordt uitgevoerd.

Maak een derde functie met bijhorende knop om de datum en de tijd weer te geven. Gebruik in de functie voor het weergeven van de datum en de tijd datum.toLocaleString().

De Kop aanpassen

De Kop 1 (h1) bevat nu *Datum weergeven*.

Verander de inhoud van de h1 tag naar *Datum en/of tijd weergeven*.

Om de kop bij het klikken op een knop aan te passen:

- Geef je aan de kop tag (h1) het id kop.
- Voeg je aan de functie datumWeergeven de volgende opdracht toe:

```
document.getElementById("kop").innerHTML = "Datum weergeven";
```

Verander naargelang de knop waarop geklikt wordt, de kop naar *Tijd weergeven* of *Datum en tijd weergeven*.

Javascript afzonderen

Verplaats alle JavaScript code naar een apart bestand en sla dit bestand op als datum.js.

Zorg dat de pagina event.html gebruik maakt van het externe JavaScript bestand datum.js.

Alle oplossingen van opdrachten staan op het internet.

Tabbladen

Bij het klikken op een tabblad verschijnt de inhoud van het tabblad en verdwijnt de inhoud van de andere tabbladen.

Met de volgende HTML code maak je een tabblad:

```
<span style="cursor:pointer" onclick="tabblad1();">HTML </span>|
```

De volgende HTML code bevat de onzichtbare inhoud van het eerste tabblad:

```
<div id="tab1" style="border:thin solid #000; padding: 0px 5px;
display:none">HTML verzorgt de structuur en de inhoud van de
webpagina.</div>
```

Schrijf zelf de JavaScript functie om de inhoud van het eerste tabblad zichtbaar te maken en alle andere tabbladen onzichtbaar te maken als je weet dat de opdracht

```
document.getElementById("tab1").style.display="";
```

het eerste tabblad laat verschijnen en de opdracht

```
document.getElementById('tab1').style.display="none";
```

het eerste tabblad laat verdwijnen.

Naast het tabblad HTML, maak je een tabblad

- **CSS** met de tekst *CSS is verantwoordelijk voor de opmaak*,
- **JavaScript** met de tekst *JavaScript voegt functionaliteit toe*,
- **PHP** met de tekst *PHP maakt dynamische onderdelen op de webserver aan*.

Plaats de JavaScript code in een extern bestand.

JavaScript Variabelen en Operatoren

Variabelen

Algebra op school: x = 5, y = 6, z = x + y.

Een letter wordt gebruikt om een getal in op te slaan (x bevat 5), waarmee je dan later z kunt bereken (11).

De letters noemen me variabelen, en worden gebruikt om waarden (bijvoorbeeld het getal 5) of bewerkingen (expressies, bijvoorbeeld: x + y) in op te slaan.

JavaScript variabelen

Variabelen hebben een naam (kort zoals x of beschrijvend zoals voornaam).

De namen van variabelen in JavaScript zijn:

- hoofdlettergevoelig
- beginnen met een letter, $ of een underscore.

De waarde van een variabele kan tijdens het uitvoeren van een script veranderen. Daarbij verwijs je naar de naam van de variabele als je de waarde ervan wilt aanpassen of opvragen.

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Voornaam</title>
6   </head>
7   <body>
8   <p><span id="gepensioneerd"></span> werd opgevolgd door <span
     id="opvolger"></span>.
9   <script type="text/javascript">
10    var voornaam;
11    voornaam = "Ingrid";
12    document.getElementById("gepensioneerd").innerHTML = voornaam;
13    voornaam = "Dany";
14    document.getElementById("opvolger").innerHTML = voornaam;
15  </script>
16  </body>
17  </html>
```

Variabelen declareren

Een variabele aanmaken noemen we declareren. Dit gebeurt met het sleutelwoord **var**.

```
var x;
var voornaam;
```

Na de declaratie zijn de variabelen leeg (ze hebben nog geen waarde).

Tijdens de declaratie kan je variabelen een waarde geven:

```
var x = 5;
var voornaam = "Dany";
```

Opmerkingen: Tekstwaarden beginnen en eindigen met aanhalingstekens. Indien de tekstwaarde zelf aanhalingstekens bevat, gebruik je enkelvoudige aanhalingstekens.

```
var uitspraak = 'De professor zei: "Ik verzet mij tegen dit
maakbaarheidsidealisme."'
```

Een opnieuw gedeclareerde variabele behoudt zijn oorspronkelijke waarde.

Speciale karakters in tekstwaarden (strings)

De backslash (\) wordt gebruikt om speciale karakters die normaal niet toegestaan zijn toch in een tekstwaarde te gebruiken.

```
var tekst="Ze noemen ons de "Vikings" van het noorden.";
```

In JavaScript begint en eindigt een tekstwaarde (string) met enkele of dubbele aanhalingstekens. Dit zorgt ervoor dat in het bovenstaand voorbeeld de variabele tekst de volgende tekstwaarde zal bevatten: Ze noemen ons de.

Om dit te voorkomen plaats je voor de aanhalingstekens in de tekstwaarde een backslash (\). Een karakter na een backslash wordt door JavaScript letterlijk in de tekstwaarde opgenomen en dus niet gezien als het einde van de tekstwaarde:

```
var tekst="Ze noemen ons de \"Vikings\" van het noorden.";
```

De variabel tekst bevat nu: Ze noemen ons de "Vikings" van het noorden.

De volgende tabel toont een lijst met speciale karakters:

Code	Teken (karakter)
\'	Enkele aanhaling (single quote)
\"	Dubbele aanhaling (double quote)
\\	backslash
\n	Nieuwe regel (new line)
\r	Terugloop (carriage return)
\t	tab
\b	backspace
\f	Volgende pagina (form feed)

Lokale variabelen

Een variabele die binnen een functie gedeclareerd wordt, kan enkel in deze functie gebruikt worden. Bij het beëindigen van de functie wordt de lokale variabele vernietigd.

Je kunt dus lokale variabelen met dezelfde naam in verschillende functies gebruiken, zonder dat ze elkaar beïnvloeden, want lokale variabelen worden enkel herkend in de functie waarin ze gedeclareerd werden.

Globale variabelen

Variabelen die buiten een functie gedeclareerd worden zijn globale variabelen. Alle scripts en functies op één webpagina kunnen de globale variabelen gebruiken.

Globale variabelen worden vernietigd bij het verlaten van de webpagina.

Niet gedeclareerde variabelen

Als je een waarde toekent aan een nog niet gedeclareerde variabele dan wordt de variabele automatisch gedeclareerd als globale variabele.

Operatoren

Je kunt bewerkingen (operaties) uitvoeren op en met variabelen.

```
Y = 5;
z = 2;
x = y + z;
```

De = operator wordt gebruikt om waarden aan een variabele toe te kennen.

De + operator wordt gebruikt om waarden samen te tellen.

De waarde van de variabele x is na het uitvoeren van de opdrachten hierboven 7.

Rekenkundige operatoren

Rekenkundige operatoren worden gebruikt om te rekenen met variabelen en/of waarden.

Met **y = 5** toont de tabel de werking van de rekenkundige operatoren:

Operator	**Beschrijving**	**Voorbeeld**	**Resultaat**	
+	Optellen	x = y + 2	x = 7	y = 5
-	Aftrekken	x = y - 2	x = 3	y = 5
*	Vermenigvuldigen	x = y * 2	x = 10	y = 5
/	Delen	x = y / 2	x = 2.5	y = 5
%	Modulus (rest bij deling)	x = y % 2	x = 1	y = 5
++	Verhogen	x = ++y	x = 6	y = 6
		x = y++	x = 5	y = 6
--	Verlagen	x = --y	x = 4	y = 4
		x = y--	x = 5	y = 4

Toewijzingsoperatoren

Toewijzingsoperatoren worden gebruikt om waarden aan variabelen toe te wijzen.

Met **x = 10** en **y = 5** toont de tabel hieronder de werking van de toewijzingsoperatoren:

Operator	**Voorbeeld**	**Hetzelfde als**	**Resultaat**
=	x = y		x = 5
+=	x += y	x = x + y	x = 15
-=	x -= y	x = x - y	x = 5
*=	x *= y	x = x * y	x = 50
/=	x /= y	x = x / y	x = 2
%=	x %= y	x = x % y	x = 0

De + operator en strings (tekenreeksen)

De + operator wordt ook gebruikt om string variabelen of tekst waarden aan elkaar te zetten.

```
begin = "Het is vandaag";
einde = "mooi weer.";
```

```
zin = begin + einde;
```

Na het uitvoeren van deze opdrachten, bevat de variabele zin de tekst "Het is vandaagmooi weer.". Om een spatie tussen de twee strings te plaatsen, voeg je aan één van de twee strings een spatie toe:

```
begin = "Het is vandaag ";
einde = "mooi weer.";
zin = begin + einde;
```

Of voeg een spatie toe aan de bewerking:

```
begin = "Het is vandaag";
einde = "mooi weer.";
zin = begin + " " + einde;
```

Na het uitvoeren van de aangepaste opdrachten bevat de variabele zin de tekst "Het is vandaag mooi weer.".

Strings en getallen samenvoegen

Bij het samenvoegen van een getal met een string zal het resultaat steeds een string zijn.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
    8" />
5  <title>Strings en getallen samenvoegen</title>
6  </head>
7  <body>
8  <p>Twee getallen optellen: <span id="tweegetallen"></span>.<br />
9  Twee tekenreeksen samenvoegen: <span id="tweestrings"></span>.<br />
10 Een getal en een tekenreeks samenvoegen: <span
    id="getalstring"></span>.<br />
11 Een tekenreeks en een getal samenvoegen: <span
    id="stringgetal"></span>.</p>
12 <script type="text/javascript">
13   x = 5 + 5;
14   document.getElementById("tweegetallen").innerHTML = x;
15   x = "5" + "5";
16   document.getElementById("tweestrings").innerHTML = x;
17   x = 5 + "5";
18   document.getElementById("getalstring").innerHTML = x;
19   x = "5" + 5;
20   document.getElementById("stringgetal").innerHTML = x;
21 </script>
22 </body>
23 </html>
```

Gebruikersinvoer opvragen

In HTML kunnen gebruikers informatie doorsturen met behulp van formuliervelden.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Formuliervelden</title>
<script type="text/javascript">
function bereken() {
  // de invoer in het tekstveld met het id prijs lezen
  var prijs = document.getElementById("prijs").value;
  // de invoer in het tekstveld met het id aantal lezen
  var aantal = document.getElementById("aantal").value;
  // het resultaat in het tekstveld met het id resultaat plaatsen
  document.getElementById("resultaat").value = prijs * aantal;
}
</script>
</head>
<body>
<p><label for="prijs">Prijs per stuk</label>
<input type="text" name="prijs" id="prijs" onchange="bereken();" />
x
<label for="aantal">Aantal</label>
<input type="text" name="aantal" id="aantal"
 onchange="bereken();" />
=
<input type="text" name="resultaat" id="resultaat" /></p>
</body>
</html>
```

De opdracht document.getElementById("prijs").value leest welke waarde de gebruikers in het tekstveld met het id prijs heeft ingetypt. Deze waarde kan in een variabele worden opgeslagen om dan verder gebruikt te worden. Deze opdracht kan je ook gebruiken om waarden in een tekstveld door JavaScript te laten invullen. M.a.w. bekijk deze opdracht als een variabele waarmee je waarden kunt opslaan in en lezen uit tekstvelden.

Waarden die in een tekstveld worden ingetypt zijn altijd strings (tekst). Ook als je alleen getallen intypt. Om met waarden uit een tekstveld te kunnen rekenen, moet je ze naar getallen omzetten. Een voorbeeld:

```
var prijs = parseFloat(document.getElementById("prijs").value);
```

Daarna ben je zeker dat de variabele prijs een getal bevat en geen tekst (string). Als je met gehele getallen zonder komma's werkt, gebruik je beter **parseInt(tekst).**

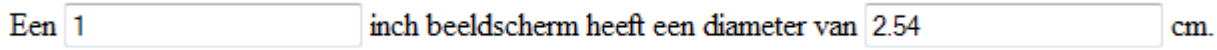
Let op het gebruik van onchange bij de tekstvelden. Bij het aanpassen van de waarde van het tekstveld wordt door onchange de functie bereken() uitgevoerd. De functie bereken wordt uitgevoerd na het intypen van een nieuwe waarde en het verlaten van het tekstveld (m.a.w. als je de cursor buiten het tekstveld plaatst).

Opdrachten

Rekenen

Maak een webpagina waarmee je een lengte in inch kunt omrekenen naar cm.
Waarbij één inch = 2,54 cm.

Maak daarvoor binnen de zin twee tekstvelden (zie afbeelding).



Na het aanpassen van de waarde in het eerste tekstveld moet de waarde van het tweede tekstveld aangepast worden. Bij het aanpassen van de waarde in het eerste tekstveld wordt een functie uitgevoerd waarmee je de waarde in het eerste tekstveld omrekent naar inch en het resultaat in het tweede tekstveld plaatst.

Schrijf de JavaScript functie om inches om te rekenen naar cm.

Bij het testen in de browser wordt de omrekening pas uitgevoerd als je de cursor naar een ander tekstvak verplaatst (werking onchange).

Een 24 inch beeldscherm heeft een diameter van ..... cm.

En omgekeerd

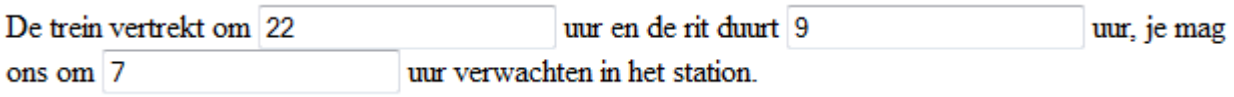
Zorg dat bij het aanpassen van de waarde in het tweede tekstveld (cm) deze waarde naar inch wordt omgerekend (in het eerste tekstveld).

Een ..... inch beeldscherm heeft een diameter van 54 cm.

Zovele uren later

Het bekendste voorbeeld van de bewerking modulus is de tijdrekening in uren, die modulo 12 of modulo 24 gaat. Zo is de tijd bijvoorbeeld 10 uur na 22 uur gelijk aan 8 uur. Dus 10 + 22 = 32 en 32 modulus 24 is 8. Of met andere woorden 32 gedeeld door 24 is 1 met als rest 8.

Maak in HTML de zin met drie tekstvelden (zie afbeelding) en zorg dat bij het aanpassen van het vertrek of de duur het aankomstuur wordt berekend en weergegeven.



Aantal tegels

Om een vloer te betegelen heb je tegels nodig, hoe groter de oppervlakte hoe meer tegels. Vloeren hebben tussen de tegels voegen. Tegels worden verkocht in verpakkingen of palletten met meerdere tegels. Hoe bereken je het aantal benodigde pakketten tegels?

1. Bereken de oppervlakte van de vloer (lengte x breedte).
2. Bereken de lengte en breedte van de tegels met hun voeg (lengte + voegbreedte, breedte + voegbreedte).
3. Bereken de oppervlakte van elke tegel met voeg (lengte x breedte).
4. Bereken het aantal benodigde tegels voor de vloer (oppervlakte van de vloer / oppervlakte van tegel met voeg).
5. Dit is een theoretisch aantal, om verliezen door snijden, e.d.m. in te calculeren neemt men 10 % extra (aantal tegels x 1,1).
6. Bereken het aantal verpakkingen (aantal tegels / aantal tegels per verpakking).

Maak een HTML pagina met tekstvelden om het aantal tegels voor een vloer te berekenen (zie afbeelding).

De resultaten voor het aantal benodigde tegels en verpakkingen zijn meestal kommagetallen. Deze kan je op de volgende manier naar boven afronden:

```
teBestellenTegels = Math.ceil(tegels);
```

Waarbij de variabele tegels een kommagetal is die naar boven wordt afgerond.

Afmetingen van de vloer: 600 x 400 in cm

Afmetingen van de tegels: 30 x 30 in cm

Breedte van de voegen: 0.5 in cm

Tegels per verpakking: 24 Bereken

De oppervlakte van de vloer is 240000 vierkante cm.

Je hebt 284 tegels nodig.

Je hebt 12 verpakkingen met tegels nodig.

Grafische voorstelling percenten

Maak een HTML pagina met een tekstveld waarin je een percentage kunt ingeven.

Zorg voor een knop waarmee je de aanmaak van de grafische voorstelling met een JavaScript functie kunt starten.

Om twee gekleurde balken naast elkaar op het scherm te plaatsen heb je de volgende HTML code nodig:

```
<p><span id="percentage" style="background-color:#ccccff;
display:inline-block; text-align:center"></span><span id="resterend"
style="background-color:#ccffcc; display:inline-block; text-
align:center"></span></p>
```

Daarbij zorgen CSS stijlen (style) voor de opmaak:

- background-color: lichtblauwe of lichtgroene achtergrondkleur,
- display:inline-block: de twee span tags naast elkaar weergeven,
- text-align:center: de tekst in de span tag centreren.

Schrijf een JavaScript functie waarmee je:

1. het resterende percentage berekent (100 – opgegeven percentage),
2. de breedte van de twee span tags overeenkomt met de percentages:

```
document.getElementById('percentage').style.width = percent + "%";
```

3. de percentages in tekstvorm in de twee tags verschijnt.

Percentage: 30 Grafisch voorstellen

30% 70%

JavaScript Controlestructuren

Via controlestructuren kan je de manier waarop je programma wordt uitgevoerd beïnvloeden.

Vergelijkings- en logische operatoren

De vergelijkings- en logische operatoren geven steeds als resultaat juist (true) of fout (false).

Vergelijkingsoperatoren

Vergelijkingsoperatoren onderzoeken de gelijkenissen of verschillen tussen variabelen of waarden.

Met **x = 5** toont de tabel de werking van de vergelijkingsoperatoren:

Operator	Beschrijving	Voorbeeld
==	is gelijk aan	x == 8 is fout
		x == 5 is juist
===	is exact gelijk aan (waarde en type)	x === 5 is juist
		x === "5" is fout
!=	is niet gelijk aan	x != 8 is juist
>	is groter dan	x > 8 is fout
<	is kleiner dan	x < 8 is juist
>=	is groter dan of gelijk aan	x >= 8 is fout
<=	is kleiner dan of gelijk aan	x <= 8 is juist

Vergelijkingsoperatoren worden gebruikt in voorwaardelijke opdrachten om waarden te vergelijken en op basis van het resultaat een actie te ondernemen:

```
if (leeftijd < 18) alert("Te jong");
```

Logische operatoren

Logische operatoren onderzoeken de logische samenhang van variabelen of waarden.

Met **x = 6** en **y = 3** toont de tabel de werking van logische operatoren:

Operator	Beschrijving	Voorbeeld
&&	en	(x < 10 && y > 1) is juist
||	of	(x == 5 || y == 5) is fout
!	niet	!(x == y) is juist

Voorwaardelijke operatoren

Voorwaardelijke operatoren kennen op basis van een voorwaarde een waarde toe aan een variabele.

Syntaxis:

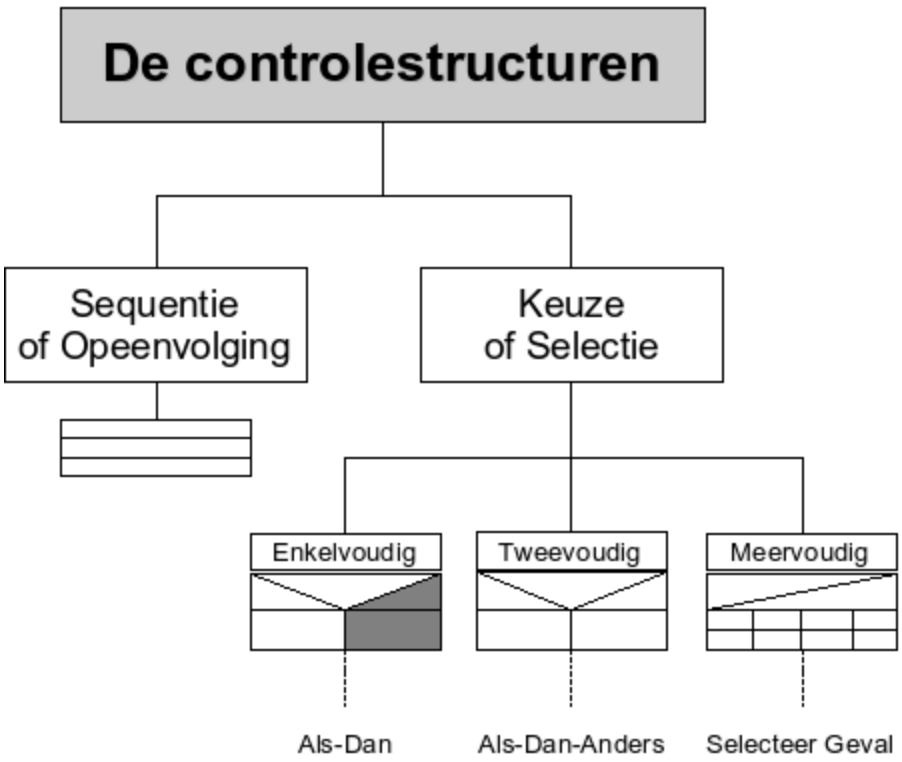
```
variabele=(voorwaarde)?waarde1:waarde2
```

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Welkom</title>
6   </head>
7   <body>
8   <h1 id="begroeting"></h1>
9   <script type="text/javascript">
10  var geslacht = "man";
```

```
11  var naam = "Jansens";
12  var begroeting = (geslacht == "man")?"Mijnheer ":"Mevrouw ";
13  document.getElementById("begroeting").innerHTML = begroeting + naam;
14  </script>
15   </body>
16   </html>
```

Voorwaardelijke opdrachten



Voorwaardelijke opdrachten voeren naargelang de voorwaarden verschillende acties uit.

In JavaScript kun je de volgende voorwaardelijke opdrachten gebruiken:

- **if** opdracht: voert de code enkel uit als aan de voorwaarde is voldaan,
- **if ... else** opdracht: voert bepaalde code uit als aan de voorwaarde is voldaan en andere code als niet aan de voorwaarde is voldaan,
- **if ... else if ... else** opdracht: voert één van de vele blokken code uit,
- **switch** opdracht: voert één van de vele blokken code uit.

If opdracht

Gebruik de if opdracht om de code enkel uit te voeren als aan de voorwaarde is voldaan.

Syntaxis:

```
if (voorwaarde) {
  code die enkel wordt uitgevoerd als aan de voorwaarde is voldaan
}
```

Voorbeeld:

```
15  <h2 id="goede"></h2>
16  <script type="text/javascript">
17  // Goedemorgen indien voor 10 uur
18  var d = new Date();
```

```
19  var time = d.getHours();
20   if (time < 10){
21     document.getElementById("goede").innerHTML = "Goedemorgen.";
22   }
23   </script>
```

Merk op dat er geen ..else.. in deze code voorkomt. M.a.w. de browser zal de code enkel uitvoeren als aan de conditie is voldaan (voor 10 uur 's morgens).

If ... else opdracht

Gebruik de if ... else opdracht om bepaalde code uit te voeren als aan de voorwaarde is voldaan of andere code uit te voeren als niet aan de voorwaarde is voldaan.

Syntaxis:

```
if (voorwaarde) {
  code wordt uitgevoerd als aan de voorwaarde is voldaan
} else {
  code wordt uitgevoerd als niet aan de voorwaarde wordt voldaan
}
```

Voorbeeld:

```
20 if (time < 10){
21   document.getElementById("goede").innerHTML = "Goedemorgen.";
22 } else {
23   document.getElementById("goede").innerHTML = "Goedendag.";
24 }
```

If ... else if ... else opdracht

Gebruik de if ... else if ... else opdracht om één van de vele blokken opdrachten uit te voeren.

Syntaxis:

```
if (eerste voorwaarde) {
  code wordt uitgevoerd als aan de eerste voorwaarde is voldaan
} else if (tweede voorwaarde) {
  code wordt uitgevoerd als aan de tweede voorwaarde is voldaan
} else {
  code wordt uitgevoerd als aan geen enkele voorwaarde is voldaan
}
```

Voorbeeld:

```
20  if (time < 10){
21    document.getElementById("goede").innerHTML = "Goedemorgen.";
22  } else if (time >= 10 && time < 16){
23    document.getElementById("goede").innerHTML = "Goedendag.";
24  } else {
25    document.getElementById("goede").innerHTML = "Goedenavond.";
26  }
27
```

Switch opdracht

Gebruik de switch opdracht om één van de vele blokken opdrachten uit te voeren.

Syntaxis:

```
switch(n){
```

```
  case 1:
    uitvoeren eerste blok opdrachten
    break;
  case 2:
    uitvoeren tweede blok opdrachten
    break;
  default:
    opdrachten die uitgevoerd worden indien n niet gelijk is aan 1 of 2
}
```

Zo werkt het: Eerst hebben we een enkelvoudige bewerking (n) (meestal een variabele) die één keer geëvalueerd wordt. De waarde van de bewerking of variabele wordt daarna vergeleken met elke case waarde in de controlestructuur. Bij een overeenkomst wordt de bijhorende blok opdrachten uitgevoerd. De **break** opdracht sluit de blok uit te voeren opdrachten af. Zonder de break opdracht worden de opdrachten van de volgende case verder uitgevoerd.

Voorbeeld:

```
28  <p id="weekdag"></p>
29  <script type="text/javascript">
30  // begroeting naargelang de weekdag
31  var d=new Date();
32  var weekdag=d.getDay();
33  switch (weekdag){
34    case 5:
35      document.getElementById("weekdag").innerHTML = "Eindelijk
     vrijdag.";
36      break;
37    case 6:
38      document.getElementById("weekdag").innerHTML = "Super
     zaterdag.";
39      break;
40    case 0:
41      document.getElementById("weekdag").innerHTML = "Rustige
     zondag.";
42      break;
43    default:
44      document.getElementById("weekdag").innerHTML = "Ik kijk uit naar
     het weekend.";
45  }
46  </script>
```

Let op:

Als de case waarden teksten (strings) zijn, moet je deze tussen aanhalingstekens plaatsen.

```
  case "Nederlands":
```

Oefeningen

Willekeurige koppeling

Met één koppeling willen we op een webpagina twee verschillende sites bereiken.
Op willekeurige basis wordt bij het aanklikken één van beide sites geopend.

Een koppeling (hyperlink) heeft de volgende HTML code:

```
<a id="koppeling" target="_blank">Doe een gok.</a>
```

Om op een willekeurige basis een opdracht te laten uitvoeren, kun je gebruik maken van:

```
var willekeurig = Math.random();
```

Deze opdracht plaatst in de variabele willekeurig een waarde tussen 0 en 1.

Als je deze willekeurige waarde test op groter dan of kleiner dan 0.5, kun je naargelang het resultaat van de test het doel van de koppeling met het id koppeling aanpassen met:

```
  document.getElementById('koppeling').href = 'http://www.snt.be';
```

In het andere geval wordt de koppeling aangepast naar http://webdesign.pindanet.be.

Door de keuze van de testwaarde 0.5 heeft elke site 50% kans om als doel voor de koppeling gebruikt te worden.

Drie sites voor één koppeling

Zorg ervoor dat een derde site (linux.pindanet.be) als doel voor de koppeling gebruikt kan worden. De drie sites moeten evenveel aan bod komen (m.a.w. elk maken ze ..... % kans om als doel gebruikt te worden).

Spreuk van de maand

Elke maand willen we op de webpagina een spreuk van de maand plaatsen.

Maak een HTML pagina met een aanpasbare alinea (p tag).

De JavaScript code bevat:

- Een variabele datum met als waarde de datum van vandaag.
- Met de JavaScript code

```
datum.getMonth()
```

  kan je de maand uit de variabele datum halen (getallen van 0 tot en met 11, waarbij het getal 0 de maand januari voorstelt).

  Gebruik deze JavaScript code in een switch opdracht om in de aanpasbare alinea elke maand een passende spreuk te plaatsen.

Maand	Spreuk
Januari	Valt op 1 januari sneeuw, die in negen dagen niet verdwijnt, dan ligt hij negen weken naar het schijnt.
Februari	Komt Februari met goed weer, dan vriest 't in voorjaar des te meer.
Maart	Is maart guur, dan krijg je een volle schuur.
April	April heldere maneschijn, zal de bloesem schadelijk zijn.
Mei	Mei koel en nat, vult de boer zijn schuur en vat.
Juni	Zo heet het is in Juni, zo koud het is in december.
Juli	Is de eerste Juli regenachtig, gans de maand is twijfelachtig.
Augustus	Begin Augustus heet, lang en wit winterkleed.
September	Vorst in September, een zachte December Warm in September, koud in December.

Maand	Spreuk
Oktober	Oktober met groene blaân, duidt een strenge winter aan.
November	November heeft maar dertig dagen, maar dubbel wind en regenvlagen.
December	Is 't op Kerstmis nog niet koud, dan vraagt de winter niet veel hout.

Om dezelfde opmaak te bekomen, gebruik je in de teksten (strings) de tag **
** om een nieuwe regel aan te maken. Een voorbeeld:

```
"Is maart guur,<br />dan krijg je een volle schuur."
```

Afbeeldingen toevoegen

- Surf naar de site webdesign.pindanet.be naar het onderdeel JavaScript, Bij de oplossing van deze oefening download en open je de gecomprimeerde map spreuken.zip. Kopieer de map spreuken naar dezelfde map als de HTML pagina met een maandelijkse spreuk.
- Plaats in de HTML code boven de aanpasbare alinea een links uitgelijnde aanpasbare afbeelding:

```
<img id="afb_maand" align="left" />
```

- Voeg voor elke maand de bijhorende afbeelding naast de spreuk. Voorbeeld:

```
  document.getElementById("afb_maand").src = "spreuken/spreuk-mei.gif";
```

Mei koel en nat,
vult de boer zijn schuur en vat.

De juiste taal

Deze opdracht zorgt dat de gebruikers in de taal van hun browser worden aangesproken.

Maak een HTML pagina met de tekst
Welkom! Wilcommen! Bienvenue! Welcome!
in een Kop 1 (h1 tag).

Om in Internet Explorer de taal van de browser te achterhalen, gebruik je de volgende JavaScript code:

```
var taal = window.navigator.userLanguage // IE
```

Als de variabele taal de waarde **niet juist** bevat, werkt de gebruiker niet met Internet Explorer.

Om de taal in andere browsers dan Internet Explorer te achterhalen, gebruik je de volgende JavaScript code:

```
  taal = window.navigator.language;
```

Sommige browsers geven niet alleen de taal weer, maar ook het land. De eerste twee letters van de variabele taal bevatten altijd de taal (**nl** voor Nederlands, **fr** voor Frans en **de** voor Duits). M.a.w. we zijn alleen geïnteresseerd in de eerste twee letters van de variabele taal.

De eerste twee letters van de variabele taal, kan je als volgt afzonderen:

```
taal.substring(0, 2)
```

Gebruik deze JavaScript code in een switch opdracht om per taal de tekst in de Kop 1 (tag h1) aan te passen naar:

- Voor het Nederlands naar **Welkom!**
- Voor het Frans naar **Bienvenue!**
- Voor het Duits naar **Wilcommen!**
- Voor alle andere talen naar **Welcome!**

JavaScript Functies

Alert bericht

Een alert bericht (box) wordt meestal gebruikt om de gebruiker te wijzen op bepaalde handelingen. De gebruiker moet om door te gaan klikken op de knop **OK**.

Een tweede toepassing van een alert bericht is het controleren van de waarde van variabelen, controlestructuren, enz. bij het foutzoeken (debuggen) van JavaScript code.

Syntaxis:

```
alert("tekst");
```

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Fout met Internet Explorer</title>
6   </head>
7   <body>
8   <script type="text/javascript">
9   alert("Regel 9: " + window.navigator.language);
10  </script>
11  </body>
12  </html>
```

Functies

Een functie wordt uitgevoerd bij een gebeurtenis (event) of bij een aanroep naar de functie.

Functies worden niet uitgevoerd bij het laden van de webpagina, ze worden enkel gedefinieerd.

Je kunt functies aanroepen vanop elke plaats op de webpagina (zelfs vanaf andere webpagina's als je de functie in een extern .js bestand hebt opgeslagen).

Functies kun je definiëren in de <head> en in de <body> tag van een webpagina (document). Om er zeker van te zijn dat de functie gedefinieerd is voor hij gebruikt wordt, wordt aangeraden de functie in de <head> van de webpagina te plaatsen.

Functies definiëren

Syntaxis:

```
function functienaam(variabele_1, variabele_2, ..., variabele_X) {
```

```
  JavaScript code
}
```

De argumenten (parameters) variabele_1, variabele_2, enz. zijn variabelen of waarden die je aan de functie doorgeeft. De { en de } bepalen het begin en het einde van de functie code.

Opmerkingen:

- Een functies zonder argumenten heeft achter de functienaam ook de ronde haken () nodig.
- Let op voor de **hoofdlettergevoeligheid** van JavaScript! Het woord function moet in kleine letters worden geschreven. De naam van de functie mag hoofdletters bevatten, maar moet bij het aanroepen op dezelfde manier inclusief hoofdlettergebruik geschreven worden.

Voorbeeld:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
 8" />
<title>Functies</title>
<script type="text/javascript">
function geklikt(){
  alert("Je hebt op de knop geklikt.");
}
</script>
</head>
<body>
<input type="button" value="Klik hier" onclick="geklikt();" />
 </body>
 </html>
```

Indien de alert opdracht niet in een functie was opgenomen, zou de alert opdracht uitgevoerd worden na het laden van de webpagina. Nu wordt de alert opdracht pas uitgevoerd als de gebruiker op de knop klikt. M.a.w. De opdracht binnen de functie geklikt() wordt pas na het klikken op de knop uitgevoerd.

De return opdracht

De return opdracht wordt gebruikt om waarden binnen een functie door te geven aan de opdracht die de functie gebruikt.

Het voorbeeld berekent het product van twee getallen en geeft het resultaat terug:

```
<script type="text/javascript">
function product(a, b){
  return a * b;
}
</script>
<p>4 x 3 = <span id="product"></span></p>
<script type="text/javascript">
  document.getElementById("product").innerHTML = product(4, 3);
</script>
```

Lokale variabelen

Als je binnen een functie een variabele met de opdracht var definieert, kan de variabele enkel in die functie gebruikt worden. Bij het verlaten van de functie wordt zo'n variabele vernietigd. Deze variabelen noemen we lokale variabelen. Je kunt lokale variabelen met dezelfde naam in

verschillende functies definiëren en gebruiken, want ze worden enkel herkend binnen de functie waarin ze aangemaakt werden.

Bij het definiëren van een variabele buiten een functie, kan je de variabele binnen alle functies op een webpagina gebruiken. Zo'n variabele is bruikbaar vanaf het aanmaken tot het verlaten (sluiten) van de webpagina. Dit noemen we **globale** variabelen.

Functie bibliotheken

Om functies op meerdere pagina's te kunnen gebruiken, plaats je ze in een afzonderlijk bestand. Dit bestand kan dan door verschillende pagina's geladen en gebruikt worden.

Het voorbeeld bestaat dan uit het JavaScript bestand functies.js:

```
1   // JavaScript Document
2   function geklikt(){
3     alert("Je hebt op de knop geklikt.");
4   }
5   function product(a, b){
6     return a * b;
7   }
```

en een HTML pagina waarop de functies geladen en gebruikt worden:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Functies</title>
6   <script type="text/javascript" src="functies.js"></script>
7   </head>
8   <body>
9   <input type="button" value="Klik hier" onclick="geklikt();" />
10  <p>4 x 3 = <span id="product"></span></p>
11  <script type="text/javascript">
12    document.getElementById("product").innerHTML = product(4, 3);
13  </script>
14  </body>
15  </html>
```

Op regel 6 worden de functies geladen, op de regels 9 en 12 worden ze gebruikt.

Oefeningen

De grootste

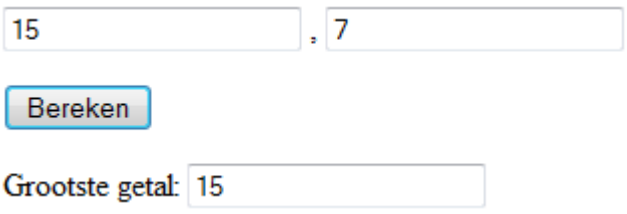
Maak een HTML pagina met twee tekstvelden in één alinea naast elkaar.

Plaats daaronder een alinea met een knop met de tekst Bereken.

Plaats daaronder een alinea met een tekstveld met de tag Grootste getal: (zie afbeelding).

Schrijf een functie die uit twee getallen het grootste getal teruggeeft.

Zorg dat bij het klikken op de knop Bereken een functie wordt uitgevoerd die:



- Het getal in het eerste tekstveld in een variabele plaatst.
- Het getal in het tweede tekstveld in een variabele plaatst.

- Het laatste tekstveld toont het grootste getal. Maak daarvoor gebruik van de eerder geschreven functie om het grootste getal te zoeken.

De grootste van drie

Pas de HTML pagina aan zodat je een derde getal kunt ingeven en daarvan het grootste getal kunt opzoeken.

Klok

Een functie kan je ook aanroepen na een bepaalde tijd. We maken een klok.

Maak een pagina met voor de klok een aanpasbare alinea.

Schrijf de functie startKlok() die:

- De datum van vandaag in een variabele plaats.
- Het uur uit de variabele met de datum haalt en in een variabele plaatst (.getHours()).
- De minuten uit de variabele met de datum haalt en in een variabele plaatst (.getMinutes()).
- De seconden uit de variabele met de datum haalt en in een variabele plaatst (.getSeconds()).
- Stel met behulp van deze variabelen de tijd met de notatie uu:mm:ss samen en plaats deze in de aanpasbare alinea.
- Laat deze functie nogmaals na een halve seconde uitvoeren met de opdracht:

```
tijd = setTimeout('startKlok()',500);
```

setTimeout is de opdracht die in het voorbeeld de functie startKlok() na 500 milliseconden (halve seconde) laat uitvoeren. Daar deze opdracht deel uitmaakt van deze functie wordt de functie tot het afsluiten van de webpagina om de halve seconde uitgevoerd.

Nu rest ons alleen nog de functie te starten na het laden van de webpagina. Dit kan door de body tag met een onload event uit te breiden:

```
<body onload="startKlok()">
```

Minuten en seconden met twee cijfers weergeven

Bij het weergeven van de minuten en seconden worden getallen kleiner dan 10 met één cijfer weergegeven. Het weergeven van steeds twee cijfers is mooier.

Schrijf een functie waaraan je een getal doorgeeft en die getallen kleiner dan 10 laat voorafgaan door een 0 (m.a.w. 2 wordt 02 en 12 blijft 12) en de twee cijfers teruggeeft.

Gebruik deze functie om de minuten en seconden steeds met twee cijfers weer te geven.

Op andere pagina's gebruiken

Aangezien het weergeven van een klok ook op andere pagina's van pas kan komen, plaats je alle functies in een afzonderlijk JavaScript bestand.

In- en uitzoomen

Om veel foto's op een webpagina te kunnen plaatsen, gebruiken we miniatuur weergaven van de foto's. Als je een foto beter wilt bekijken, kun je door erop te klikken de foto in ware grootte bekijken. Nogmaals klikken, verkleint de foto opnieuw.

Maak een HTML pagina met een foto met een breedte en hoogte van 25%. Daarvoor gebruik je de volgende HTML code:

```
<img src="paarden.jpg" style="width:25%; height:25%"
onclick="afbeelding = this; zoom();" />
```

Zoals je ziet gebruiken we CSS om de breedte en hoogte van de foto aan te passen (1/4 of 25 % van de werkelijke grootte).

Bij het klikken op de foto krijgt de globale variabele afbeelding de waarde **this**. De opdracht this bevat het adres van de tag waarop je klikte (net zoals document.getElementById('idnaam') het adres van de tag met het id idnaam bevat).

Pas daarna wordt de functie zoom() uitgevoerd.

Plaats juist onder deze foto de tekst:
Klik op de foto om in of uit te zoomen.

Schrijf de volgende JavaScript code:

- Definieer een globale variabele voor de zoomfactor (als de waarde 1 is wordt ingezoomd, -1 is uitzoomen).
- Definieer een globale variabele voor het adres van de afbeelding.

Schrijf de functie zoom() die:

- De globale variabele met de zoomfactor op inzoomen plaatst als de breedte van de afbeelding 25% is of op uitzoomen plaats als de breedte van de afbeelding 100% is.
- Tel de zoomfactor bij de afmeting (breedte) van de afbeelding.
- Geef de afbeelding de nieuwe afmetingen (breedte en hoogte in percent).
- Voer de functie zoom() om de 5 milliseconden uit tot de afbeelding 25% of 100% van zijn originele afmetingen bereikt.

Aangezien je deze interessante functie ook op andere pagina's wilt gebruiken, zonder je de noodzakelijke JavaScript code af in een apart bestand.

JavaScript lussen

Vaak wil je een stuk code verschillende keren na elkaar uitvoeren. In plaats van die gelijkaardige opdrachtregels telkens te herhalen, plaatsen we die in een lus.

JavaScript kent verschillende lussen:

- **for:** herhaalt een vast aantal keer een codeblok.
- **while**: herhaalt een codeblok tot aan een bepaalde voorwaarde is voldaan.

De for lus

De for lus wordt gebruikt als je op voorhand weet hoeveel keer je een stuk code wilt uitvoeren.

Syntaxis:

```
for (variabele = startwaarde; variabele <= eindwaarde; variabele =
variabele + toename) {
  uit te voeren code
}
```

Voorbeeld (het weergeven van een voorbeeld van alle mogelijke koptags):

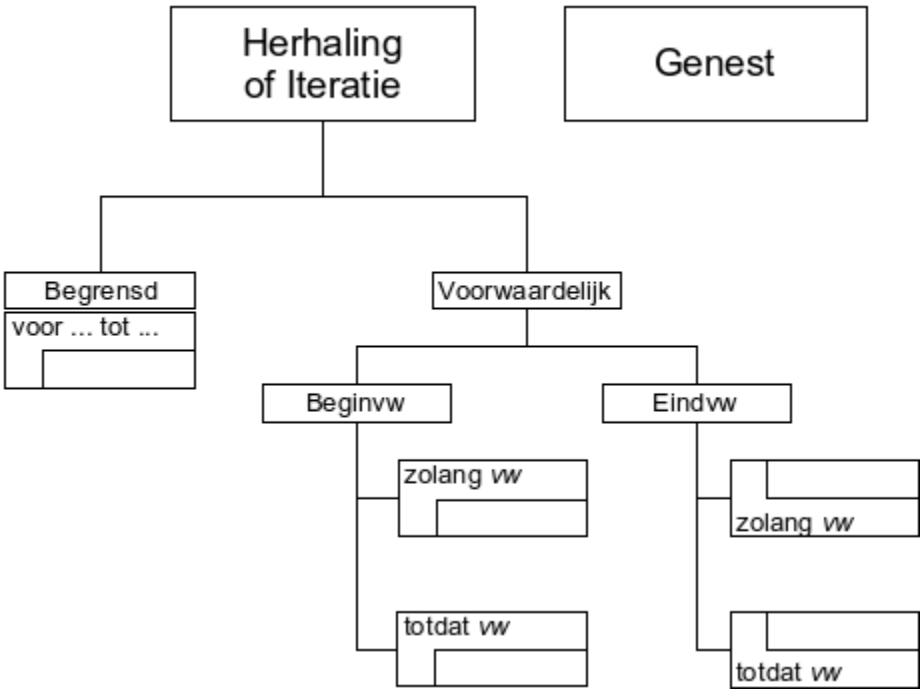
```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
```

```
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
10  var koppen = "";
11  for (kop = 1; kop <= 6; kop++){
12    koppen += "<h" + kop + ">Dit is een Kop " + kop + "</h" + kop +
     ">";
13  }
14  document.getElementById('koppen').innerHTML = koppen;
15  </script>
16  </body>
17  </html>
```

Het voorbeeld gebruikt een lus waarbij in het begin de variabele kop de waarde 1 krijgt. De lus blijft lopen zolang de variabele kop een waarde heeft die kleiner of gelijk is aan zes. Na het voltooien van elke lus wordt bij de variabele blok één bijgeteld.

Opmerkingen:

- Na het uitvoeren van een codeblok kan je de waarde van de teller (kop) ook verlagen.
- De vergelijkingsoperator (<=) kan door elke ander vergelijkingsoperator vervangen worden. In het voorbeeld kan je de voorwaarde zonder het resultaat te beïnvloeden, vervangen door kop < 7.



De while lus

De while lus voert het codeblok uit tot aan de voorwaarde wordt voldaan.

Syntaxis:

```
while (variabele <= eindwaarde){
  uit te voeren code
}
```

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
10  var koppen = "";
11  var kop = 1;
12  while (kop <= 6){
13    koppen += "<h" + kop + ">Dit is een Kop " + kop + "</h" + kop +
     ">";
14    kop++;
15  }
16  document.getElementById('koppen').innerHTML = koppen;
17  </script>
18  </body>
19  </html>
```

De variabele kop krijgt de beginwaarde 1. Het codeblok wordt uitgevoerd zolang de waarde van kop kleiner of gelijk is aan zes. Op het einde van het codeblok wordt de variabele kop met één opgehoogd.

De do ... while lus

De do ... while lus is een variant op de while lus. Deze lus doorloopt het codeblok minstens één keer, en wordt herhaald zolang niet aan de voorwaarde wordt voldaan.

Syntaxis:

```
do {
  uit te voeren code
} while (variabele <= eindwaarde);
```

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
10  var koppen = "";
11  var kop = 1;
12  do {
13    koppen += "<h" + kop + ">Dit is een Kop " + kop + "</h" + kop +
     ">";
14    kop++;
15  } while (kop <= 6);
```

```
16  document.getElementById('koppen').innerHTML = koppen;
17  </script>
18  </body>
19  </html>
```

Zelfs indien niet aan de voorwaarde wordt voldaan wordt de luscode éénmaal uitgevoerd. De voorwaarde wordt namelijk pas op het einde van de lus geëvalueerd.

Onderbreken en doorgaan

De break opdracht

De break opdracht onderbreekt de lus en de aanwezige opdrachten na de lus worden uitgevoerd.

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
10  var koppen = "";
11  for (kop = 1; kop <= 6; kop++){
12    if(kop == 3){
13      break;
14    }
15    koppen += "<h" + kop + ">Dit is een Kop " + kop + "</h" + kop +
     ">";
16  }
17  document.getElementById('koppen').innerHTML = koppen;
18  </script>
19  </body>
20  </html>
```

Het voorbeeld toont dus enkel voorbeelden van Kop 1 en 2.

De continue opdracht

De continue opdracht onderbreekt de lus maar gaat verder met de volgende lusdoorloop.

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
```

```
10  var koppen = "";
11  for (kop = 1; kop <= 6; kop++){
12    if(kop == 3){
13      continue;
14    }
15    koppen += "<h" + kop + ">Dit is een Kop " + kop + "</h" + kop +
     ">";
16  }
17  document.getElementById('koppen').innerHTML = koppen;
18  </script>
19  </body>
20  </html>
```

Het voorbeeld toont alle Kop voorbeelden, uitgenomen het voorbeeld voor Kop 3.

De for .. in lus

De for ... in lus doorloopt de onderdelen (properties) van een object.

Syntaxis:

```
for (variabele in object){
  uit te voeren code
}
```

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Koppen</title>
6   </head>
7   <body>
8   <div id="koppen"></div>
9   <script type="text/javascript">
10  var koppen = "";
11  var koptags = {h1:1, h2:2, h3:3, h4:4, h5:5, h6:6}
12  for (kop in koptags){
13   koppen += "<" + kop + ">Dit is een Kop " + koptags[kop] + "</" +
     kop + ">";
14  }
15  document.getElementById('koppen').innerHTML = koppen;
16  </script>
17  </body>
18  </html>
```

De variabele koptags noemen we een **object**. Deze bevat de variabelen met de namen h1, h2, h3, h4, h5 en h6 (gedeelte voor de :). Deze variabelen hebben respectievelijk de waarden 1, 2, 3, 4, 5 en 6 (gedeelte na de :). M.a.w. in dit geval bevat een object verschillende variabelen met bijhorende waarden. Objecten kunnen dus gebruikt worden om verschillende variabelen te groeperen.

In de for lus krijgt de variabele kop bij elke doorloop als waarde de naam van de volgende objectvariabele. De waarde van de objectvariabele is bereikbaar via de objectnaam[variabelenaam] (in het voorbeeld koptags[kop]). De lus wordt uitgevoerd tot alle objectvariabelen aan bod kwamen.

Geneste lussen

Om complexere structuren zoals tabellen met rijen en kolommen op de bouwen, gebruik je een lus binnen een lus. In elke rij plaatsen we een aantal cellen die de kolommen vormen. De buitenste lus doorloopt dan de rijen, de binnenste lus de kolommen. Het gebruik van een lus binnen een lus noemen we lussen nesten.

Een eenvoudige tabel met twee rijen en twee kolommen bestaat uit de volgende HTML code:

```
<table id="tabel">
  <tr>
    <td>A1</td>
    <td>B1</td>
  </tr>
  <tr>
    <td>A2</td>
    <td>B2</td>
  </tr>
</table>
```

De <tr></tr> tag zorgt binnen een tabel voor een rij.
De <td></td> tag zorgt binnen een rij voor een cel (kolom).

Voorbeeld:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Nesten</title>
6   </head>
7   <body>
8   <table id="tabel">
9   </table>
10  <script type="text/javascript">
11  var tabel = "";
12  for(rij = 1; rij <= 7; rij++){ // rijen binnen de tabel
13    tabel += "<tr>";
14    for(kolom = 1; kolom <= 10; kolom++){ // cellen in een rij
15      tabel += "<td>" + String.fromCharCode(64 + kolom) + rij +
     "</td>";
16    }
17    tabel += "</tr>";
18  }
19  document.getElementById("tabel").innerHTML = tabel;
20  </script>
21  </body>
22  </html>
```

De opdracht String.fromCharCode(ASCII code) zet een getal (ASCII code) om naar een letter (de ASCII code 65 komt overeen met een hoofdletter A).

Oefeningen

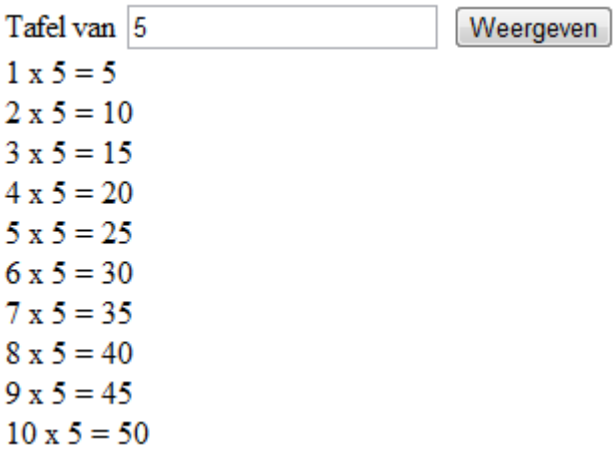
Tafels van vermenigvuldiging

Maak een HTML pagina met een tekstveld met het id tafelvan en de tag Tafel van.

Plaats achter het tekstveld een knop met de tekst Weergeven. Bij het klikken op de knop Weergeven moet de de functie tafel('tafelvan') uitgevoerd worden.

Plaats daaronder een aanpasbare div tag.

Schrijf een functie waarmee je de tafels van vermenigvuldiging van een getal in een tekstveld met het opgegeven id in de aanpasbare div kunt weergeven. M.a.w.:



- Plaats met de opdracht

  parseInt(document.getElementById(getalid).value) het getal waarvan je de tafel van vermenigvuldiging moet weergeven in een variabele. Daarbij bevat getalid het id tafelvan die je bij het onclick event doorgaf.
- Plaats de volledige tafel van vermenigvuldiging in een variabele (de tag om een nieuwe regel te beginnen is <br />).
- Plaats de tafel van vermenigvuldiging in de aanpasbare div tag.

De functie universeler maken

Pas de functie aan zodat je ook het id van de tag waarin de tafel van vermenigvuldiging wordt weergegeven kunt meegeven.

Nu de functie universeler inzetbaar is, plaats je de functie in een afzonderlijk JavaScript bestand.

Scrabble woordwaarde

Maak een webpagina met een tekstveld en de knop Bereken waarde.

Daaronder plaats je een aanpasbare alinea.

Bij het klikken op de knop wordt een functie uitgevoerd die:

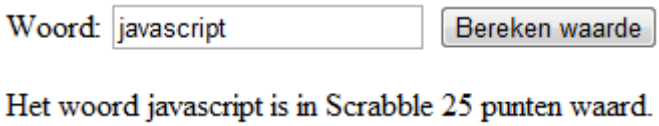
- Het woord in het tekstveld in een variabele plaatst.
- De verschillende letterwaarden staan in het volgende object:

```
  var letterwaarden = {a:1, b:3, c:5, d:1, e:1, f:4, g:3, h:4, i:1,
j:4, k:3, l:3, m:3, n:1, o:1, p:3, q:10, r:2, s:2, t:2, u:4, v:4, w:5,
x:8, y:8, z:4};
```

- Een lus om van elke letter in het opgegeven woord de waarde aan de woordwaarde toe te voegen.

  Om één letter uit een woord (string) te halen, kan je gebruik maken van de opdracht woord[letterpositie], waarbij letterpositie een getal is tussen 0 en de lengte van de tekst in de variabele woord.

  De lengte van de string opgeslagen in de variabele woord kan je met de opdracht woord.length bepalen.
- Plaats de gevonden woordwaarde in de aanpasbare alinea.



De functie universeler maken

Pas de functie aan zodat je van een woord in een tekstveld met om het even welk id de woordwaarde kunt berekenen.

Zorg er eveneens voor dat de berekende woordwaarde in een aanpasbaar element met om het even welk id geplaatst kan worden.

Nu de functie universeler inzetbaar is, plaats je de functie in een afzonderlijk JavaScript bestand.

Webveilige kleurentabel

Je gaat een tabel samenstellen met alle webveilige kleuren.

Maak daarvoor een HTML pagina met een lege aanpasbare tabel.

HTML kleuren bestaan uit de drie basiskleuren rood, groen en blauw. De basiskleuren mengen we om andere kleuren te bekomen. In HTML wordt de hoeveelheid van een basiskleur uitgedrukt in hexadecimale getallen, van 00 (geen) tot FF (max). Zo krijgt rood de HTML kleurcode #FF0000 , groen de kleurcode #00FF00 en blauw de kleurcode #0000FF. Webveilige mengwaarden gebruiken steeds veelvouden van 33.

Om gewone getallen in hexadecimale getallen om te zetten, heb je de volgende functie nodig:

```
function toHex(getal){
  if(getal.toString(16) == 0){ // uitzondering voor het getal nul
      return "00"; // steeds twee nullen teruggeven
  }
  return getal.toString(16).toUpperCase();
}
```

Met de opdracht getal.toString(16) wordt de waarde van de variabele getal omgezet naar een hexadecimaal getal (basis 16). Met de toevoeging .toUppercase() worden eventuele letters direct naar hoofdletters omgezet.

De kleurentabel bouw je als volgt op:

- De gebruikte hoeveelheid blauw varieert van ff naar 99 in stappen van 33:

```
for(b = 0xff; b >= 0x99; b -= 0x33)
```

  Let daarbij op de schrijfwijze van hexadecimale getallen.

- Binnen deze lus varieert de hoeveelheid rood van ff tot 00 in stappen van 33.

  Deze lus gebruiken we om een rij aan te maken.

- De cellen binnen een rij worden aangemaakt door twee **niet** geneste lussen.
  - ✓ De eerste lus varieert de hoeveelheid groen van 00 tot ff in stappen van 33.
  - ✓ De tweede lus varieert de hoeveelheid groen van ff tot 00 in stappen van 33.

  Aangezien dit tweemaal dezelfde kleuren zou opleveren, verminder je de hoeveelheid blauw voor het uitvoeren van de tweede lus met 99.

  Vergeet na het uitvoeren van de tweede lus de hoeveelheid blauw niet te herstellen naar de oorspronkelijke waarde.

  De HTML code voor een cel heeft de volgende structuur:

```
<td style="background-color:#99CCFF">#99CCFF</td>
```

  Waarbij 99 de hexadecimaal hoeveelheid rood, CC de hoeveelheid groen en FF de hoeveelheid blauw voorstelt.

Kleuren beter rangschikken

Bij een test blijken de kleuren van de tweede doorloop voor blauw niet aan te sluiten op de eerste en laatste doorloop.

M.a.w. zorg dat je bij de kleurwaarde CC voor blauw je het doorlopen van de kleurwaarden voor rood omkeert.

Om de tabel af te werken vervang je de teksten in de cellen door een aantal vaste spaties (HTML code voor een vaste spatie:).

JavaScript Events

Events zijn gebeurtenissen die door JavaScript opgemerkt worden, waardoor bij het optreden van zo'n gebeurtenis JavaScript code kan uitvoeren. M.a.w. JavaScript reageert op gebeurtenissen (events) met een actie (event handle).

De events worden in de HTML tags geplaatst.

Voorbeelden van events:

- een muisklik
- een pagina of afbeelding laden
- een onderdeel met de muis aanwijzen
- een veld in een formulier selecteren
- een formulier verzenden
- een toets indrukken

Opmerking: Events worden meestal in combinatie met functies gebruikt. De functie wordt dan pas uitgevoerd na het voorkomen van de gebeurtenis (event).

De belangrijkste events

onLoad en onUnload

De onLoad en onUnload zijn gebeurtenissen die voorkomen bij het betreden en verlaten van een webpagina.

De onLoad gebeurtenis kan gebruikt worden om het gebruikte browsertype en browserversie te achterhalen en daar gepast op te reageren.

De onLoad en onUnload gebeurtenissen worden vaak gebruikt in combinatie met cookies die aangemaakt worden bij het betreden of verlaten van een pagina. Voorbeeld: bij het eerste bezoek aan een webpagina kan naar de gewenste taal gevraagd worden. Na het opgeven wordt de gewenste taal in een cookie opgeslagen.

onFocus, onBlur en onChange

De onFocus, onBlur en onChange gebeurtenissen worden veel gebruikt voor het valideren van formuliervelden.

De onFocus event wordt actief als de gebruiker een formulierveld activeert om het in te vullen. De onBlur event wordt actief als ge gebruiker een formulierveld verlaat om een ander formulierveld te activeren.

Het voorbeeld maakt gebruik van de onChange event. De functie controleerEmail() wordt uitgevoerd bij het aanpassen van het tekstveld.

```
<input type="text" size="30" id="email" onchange="controleerEmail()" />
```

onSubmit

De onSubmit gebeurtenis wordt gebruikt om alle formuliervelden voor het verzenden te valideren.

Het voorbeeld toont de werking van het onSubmit event. De functie controleerFormulier wordt uitgevoerd als de gebruiker op de knop Verzenden van het formulier klikt. Als de formuliervelden niet correct zijn ingevuld, gaat het verzenden niet door. De functie controleerFormulier() geeft true (alles in orde) of false (niet alles in orde) terug. Bij true wordt het formulier verzonden, bij false wordt het verzenden afgebroken en kan de gebruiker zijn invoer corrigeren.

```
<form method="post" action="form.php" onsubmit="return
controleerFormulier()">
```

onMouseOver en onMouseOut

Bij het aanwijzen van een HTML onderdeel ontvangt JavaScript een onMouseOver gebeurtenis.

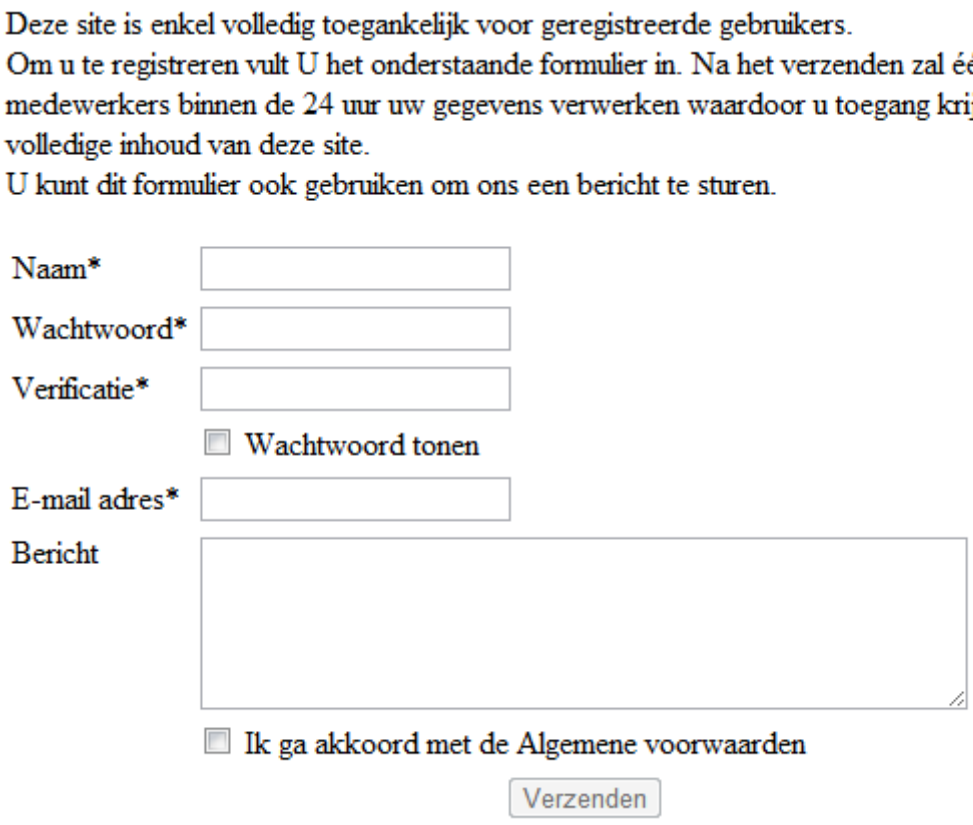
Bij het verlaten van een aangewezen HTML onderdeel ontvangt JavaScript een onMouseOut gebeurtenis. Deze laatste gebeurtenis wordt veel gebruikt om een onMouseOver actie terug ongedaan te maken.

Voorbeeld: Bij het aanwijzen van een knop wordt de achtergrondkleur aangepast (onMouseOver). Bij het met de muiswijzer verlaten van de knop wordt de oorspronkelijke achtergrondkleur terug hersteld (onMouseOut).

Oefeningen

Registratieformulier

Maak het volgende formulier. Opgelet een formulier werkt pas correct als alle onderdelen in één form tag staan.



In HTML code:

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
   8" />
5  <title>Registratieformulier</title>
6  </head>
7  <body>
8  <p>Deze site is enkel volledig toegankelijk voor geregistreerde
   gebruikers.<br />
9    Om u te registreren vult U het onderstaande formulier in. Na het
   verzenden zal één van onze medewerkers binnen de 24 uur uw gegevens
   verwerken waardoor u toegang krijgt tot de volledige inhoud van
   deze site.<br />
10   U kunt dit formulier ook gebruiken om ons een bericht te
   sturen.</p>
11 <form id="form1" name="form1" method="post" action="">
12   <table border="0">
13     <tr>
14       <td><label for="naam">Naam* </label></td>
15       <td><input type="text" name="naam" id="naam" /></td>
16     </tr>
17     <tr>
18       <td><label for="wachtwoord">Wachtwoord* </label></td>
19       <td><input type="password" name="wachtwoord"
   id="wachtwoord" /></td>
20     </tr>
21     <tr>
22       <td><label for="verificatie">Verificatie* </label></td>
23       <td><input type="password" name="verificatie" id="verificatie"
   /></td>
24     </tr>
25     <tr>
26       <td></td>
27       <td><input type="checkbox" name="wachtwoordtonen"
   id="wachtwoordtonen" />
28       <label for="wachtwoordtonen">Wachtwoord tonen</label></td>
29     </tr>
30     <tr>
31       <td><label for="email">E-mail adres* </label></td>
32       <td><input type="text" name="email" id="email" /></td>
33     </tr>
34     <tr>
35       <td valign="top"><label for="bericht">Bericht </label></td>
36       <td><textarea name="bericht" id="bericht" cols="45"
   rows="5"></textarea></td>
37     </tr>
38     <tr>
39       <td></td>
40       <td><input type="checkbox" name="akkoord" id="akkoord" />
41       <label for="akkoord">Ik ga akkoord met de Algemene
   voorwaarden</label></td>
42     </tr>
43     <tr>
44       <td></td>
45       <td align="center"><input name="button" type="submit"
   disabled="disabled" id="verzenden" value="Verzenden" /></td>
```

```
46      </tr>
47    </table>
48  </form>
49  </body>
50  </html>
```

Invulvelden markeren

Om de achtergrondkleur van een HTML onderdeel naar lichtgeel (#ffffcc) te veranderen, gebruik je de volgende functie:

```
function bglichtgeel(adres){
  adres.style.backgroundColor = "#ffffcc";
}
```

Deze functie heeft het adres van het HTML onderdeel nodig. Dit adres kan je bij de aanroep van de functie meegeven met de opdracht **this**. De opdracht this geeft zoals de opdracht document.getElementById(id) een adres van een HTML onderdeel terug. Bij de opdracht this wordt het adres teruggeven van het HTML element van waaruit je de functie aanroept. M.a.w. onfocus="bglichtgeel(this);" zorgt voor lichtgele achtergrond bij het activeren van een formulierveld.

Zorg dat actieve tekstvelden een lichtgele achtergrond krijgen.

Schrijf een functie om de achtergrondkleur te verwijderen. Doe dit door geen achtergrondkleur in te stellen (backgroundColor = "").

Gebruik deze functie om de achtergrondkleur bij het verlaten van een tekstveld te herstellen.

Wachtwoord verificatie

Schrijf een functie met de volgende kenmerken:

- de functie wordt uitgevoerd bij het verlaten van het Verificatie tekstveld,
- de functie krijgt het adres van het verificatieveld en het id van het wachtwoordveld mee,
- als beide invoervelden dezelfde inhoud hebben, wordt het verificatieveld lichtgroen (#ccffcc) en geeft de functie true (alles in orde) terug,
- als beide invoervelden een verschillende inhoud hebben, wordt het verificatieveld lichtrood en geeft de functie false (niet in orde) terug.

Wachtwoorden weergeven

Het verschil tussen een tekstveld voor het invullen van een wachtwoord en gewone tekst zit in het type veld. Voor een wachtwoord is dit "password", voor gewone tekst "text". Om het type van een veld met JavaScript aan te passen of op te vragen, gebruik je de opdracht document.getElementById(id).type

Schrijf een functie waarmee je het type van twee opgegeven id's van velden kunt omschakelen. M.a.w. indien het gewone tekstvelden zijn, worden het wachtwoordvelden en omgekeerd.

Gebruik deze functie als er geklikt wordt op het Wachtwoorden tonen selectievakje.

Validatie e-mailadres

Een e-mailadres moet:

- een @ bevatten,
- een punt bevatten,
- het e-mailadres mag niet beginnen met een @,
- de punt moet achter het @ staan,

- na de punt moeten minstens twee tekens staan.

Schrijf een functie waarmee je de geldigheid van een e-mailadres in een tekstveld kunt controleren.

De positie van een teken in een tekst (string) variabele kan je bepalen met de opdracht tekstvar.indexOf("@"), daarbij is tekstvar de variabele met de tekst en @ het teken waarvan we de positie willen bepalen.

Om de laatste positie van een teken in een string te bepalen, gebruik je de opdracht tekstvar.lastIndexOf("@").

Indien het e-mailadres niet aan de gestelde eisen voldoet, plaats je de achtergrond van het tekstveld lichtrood en geeft de functie de waarde false terug.

Indien het e-mailadres wel aan de gestelde eisen voldoet, plaats je de achtergrond van het tekstveld lichtgroen en geeft de functie de waarde true terug.

Gebruik deze functie bij het verlaten van het tekstveld waar het e-mailadres wordt ingevuld.

Akkoord gaan en verzenden

De knop Verzenden is momenteel niet bruikbaar (disabled="disabled"). De bedoeling is dat de gebruiker een vinkje plaatst bij "Ik ga akkoord met de Algemene voorwaarden", waardoor de knop Verzenden actief wordt (disabled=""). De eigenschap disabled kan je met JavaScript aanpassen zoals je dat deed met de eigenschap type.

Schrijf een functie om de knop Verzenden te activeren.

Om te controleren of in een selectievakje een vinkje staat, gebruik je de opdracht adres.checked, waarbij de variabele adres het adres van het selectievakje bevat. De opdracht geeft de waarde true (vinkje) of false (geen vinkje) terug.

Verplichte velden

Bij het verzenden van het formulier moet voor het verzenden een functie uitgevoerd worden om te controleren of alle verplichte velden (herkenbaar aan *) ingevuld werden.

Schrijf een functie met de volgende kenmerken:

- de verplichte velden worden via een object aan de functie doorgegeven,
- bij een leeg verplicht veld wordt de achtergrond van het formulierveld lichtrood,
- bij een ingevuld verplicht veld wordt de achtergrond van het formulierveld lichtgroen (#ccffcc),
- indien één van de verplichte velden niet is ingevuld, geeft de functie false terug, anders true,
- steeds alle verplichte velden worden gecontroleerd (zo ziet de gebruiker duidelijk wat nog moet ingevuld worden (lichtrood) en wat reeds ingevuld is (lichtgroen).

Gebruik deze functie bij het verzenden van het formulier.

Globale formuliercontrole

Nu de functie voor de verplichte velden werkt, kunnen we een globale formuliercontrole uitvoeren.

Schrijf een functie met de volgende kenmerken:

- controleer de verplichte velden,
- vergelijk het wachtwoord met de verificatie,
- controleer het e-mail adres,
- indien bij één van de controles niet is voldaan aan de voorwaarden wordt het formulier niet verzonden.

Universele functies afzonderen

Veel van de hier geschreven functies kunnen in andere formulieren gebruikt worden.

Probeer de functies zo universeel mogelijk te maken.

Plaats enkel de universele functies in een afzonderlijk bestand.

In- en uitfaden

Maak een HTML pagina met een Kop 1 (h1 tag) met de tekst: De inhoud van deze pagina wordt langzaam zichtbaar.

Plaats daaronder een koppeling (hyperlink, a tag) met de tekst Pagina verlaten en als koppeling # (koppeling die eigenlijk niets doet).

Pagina verlaten

Maak de body tag met een CSS stijl (style="opacity:0") doorzichtig.

Zorg dat de pagina (body) na het laden een functie start om de inhoud zichtbaar te maken.
De functie heeft de volgende kenmerken:

- de transparantie van de body tag kan je opvragen of aanpassen met de opdracht

  document.body.style.opacity

  De body tag is volledig doorzichtig als de opacity stijl nul is, en volledig zichtbaar als de opacity één is, half doorzichtig bij een opacity van 0,5.
- als de pagina (body tag) volledig transparant is, moet de opacity groter worden tot de pagina volledig zichtbaar is
- als de pagina (body tag) volledig zichtbaar is, moet de opacity kleiner worden tot de pagina volledig doorzichtig is.
- de opacity wordt om de 0,1 seconde aangepast in stappen van 0,05.

Uitfaden bij het klikken op een koppeling

Zorg dat bij het klikken op de koppeling (a tag) een functie uitgevoerd wordt.
Deze functie heeft de volgende kenmerken:

- slaat de opgegeven webpagina waar je naar toe wilt surfen op in een variabele,
- voert de bovenstaande functie om de opacity aan te passen uit.

Pas de originele functie om de opacity aan te passen zo aan, dat bij het bereiken van een volledig doorzichtige pagina de opgegeven en opgeslagen webpagina geladen wordt. Om via JavaScript naar een pagina te surfen gebruik je de opdracht:

```
window.location.href = href;
```

waarbij de variabele href de URL van de webpagina bevat.

Plaats de JavaScript code in een afzonderlijk bestand.

Zorg dat alles werkt in verschillende browsers.

JavaScript Objecten

JavaScript is een Object geOriënteerde Programmeertaal (OOP). Met een Object geOriënteerde Programmeertaal kun je objecten en eigen variabeletypen definiëren.

JavaScript zelf bevat reeds een aantal veel gebruikte ingebouwde objecten.

Eigenschappen en methoden

Een object is een speciale manier om met gegevens om te gaan. Ze bevatten eigenschappen (properties) en methoden (methods).

Eigenschappen (properties)

De eigenschappen van een object bevatten de waarden die in het object voorkomen.

```
var tekst="JavaScript is tof!";
alert(tekst.length);
```

In het voorbeeld gebruiken we lenght eigenschap van het String object tekst om de lengte van de tekst te tonen (18).

Methoden (methods)

Methoden zijn acties die je op een object kunt uitvoeren.

```
var tekst="JavaScript is tof!";
alert(tekst.toUpperCase());
```

In het voorbeeld gebruiken we de toUpperCase() methode van het String object tekst om deze in hoofdletters weer te geven (JAVASCRIPT IS TOF!).

Overzicht

Veel gebruikte ingebouwde objecten zijn:

- **Array objecten** worden gebruikt om meerdere waarden in één variabele te plaatsen.
- **Boolean objecten** worden gebruikt om niet Boolean waarden naar Boolean waarden (true of false) om te zetten.
- **Date objecten** worden gebruikt om met datums en tijdstippen te werken.
- **Math objecten** worden gebruikt om wiskundige bewerkingen uit te voeren.
- **Number objecten** bevatten hulpmiddelen voor het werken met basisgetallen.
- **String objecten** worden gebruikt om met teksten te werken.
- **RegEx objecten** worden gebruikt om een groep tekens te omschrijven, van belang bij zoek en vervang opdrachten in een tekst.
- **Global objecten** zijn eigenschappen en methoden die je met elk ingebouwd JavaScript object kunt gebruiken.

Een volledig overzicht van alle JavaScript objecten kun je vinden op de webpagina http://www.w3schools.com/jsref/.

Eigen objecten

Objecten zijn handig om informatie te organiseren.

We gaan dit aantonen met een voorbeeld: een persoon is een object. De eigenschappen van een persoon zijn de naam, lengte, gewicht, leeftijd, huidskleur, oogkleur, enz. Iedereen heeft deze eigenschappen, maar de waarden van deze eigenschappen zullen voor verschillende personen verschillen. De methoden van een persoon zijn bijvoorbeeld eten(), slapen(), werken(), spelen(), enz.

Eigenschappen (properties)

Syntaxis:

```
objectNaam.eigenschapNaam
```

Je kunt een eigenschap aan een object toevoegen door er een waarde aan te geven:

```
vader.naam =  "John";
```

Je kunt een eigenschap van een object gebruiken:

```
alert(vader.naam);
```

Daarbij is de naam van het object vader en de gebruikte eigenschap naam.

Methoden (methods)

Syntaxis:

```
objectNaam.methodeNaam();
```

Argumenten kan je tussen de haken plaatsen.

```
vader.slapen();
```

Daarbij is de naam van het object vader en de uitgevoerde methode slapen().

Een object aanmaken

Een object direct aanmaken

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Objecten</title>
6   </head>
7   <body>
8   <p id="weergeven"></p>
9   <script type="text/javascript">
10  var vader = {voornaam:"John", naam:"De Bakker", leeftijd:50,
     oogkleur:"blauw"};
11  document.getElementById('weergeven').innerHTML = vader.voornaam + "
     " + vader.naam + " is " + vader.leeftijd + " jaar jong.";
12  </script>
13  </body>
14  </html>
```

Een object indirect aanmaken met een functie

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Objecten</title>
6   </head>
7   <body>
8   <p id="weergeven"></p>
9   <script type="text/javascript">
```

```
10  function persoon(voornaam, naam, leeftijd, oogkleur){
11    this.voornaam = voornaam;
12    this.naam = naam;
13    this.leeftijd = leeftijd;
14    this.oogkleur = oogkleur;
15  }
16  var vader = new persoon("John", "De Bakker", 50, "blauw");
17  document.getElementById('weergeven').innerHTML = vader.voornaam + "
     " + vader.naam + " is " + vader.leeftijd + " jaar jong.";
18  </script>
19  </body>
20  </html>
```

De functie gebruikt **this**.eigenschapNaam om waarden aan eigenschappen toe te kennen. We gebruiken this omdat bij het definiëren van de functie de objectnaam nog niet gekend is. Bij het uitvoeren van de functie wordt this vervangen door de opgegeven objectnaam.

Een functie om een object aan te maken noemen we een **contructor**, die je kunt gebruiken om met de opdracht **new** nieuwe objecten aan te maken:

```
var moeder = new persoon("Sandrine", "Ramout", 48, "groen");
```

Op dezelfde manier kun je methoden aan een object toevoegen:

```
function persoon(voornaam, naam, leeftijd, oogkleur){
  this.voornaam = voornaam;
  this.naam = naam;
  this.leeftijd = leeftijd;
  this.oogkleur = oogkleur;
  this.familienaamInvoeren = function(familienaam){
     this.familienaam = familienaam;
  };
}
```

Merk op dat methoden (functies binnen een object) in het object worden gedefinieerd.

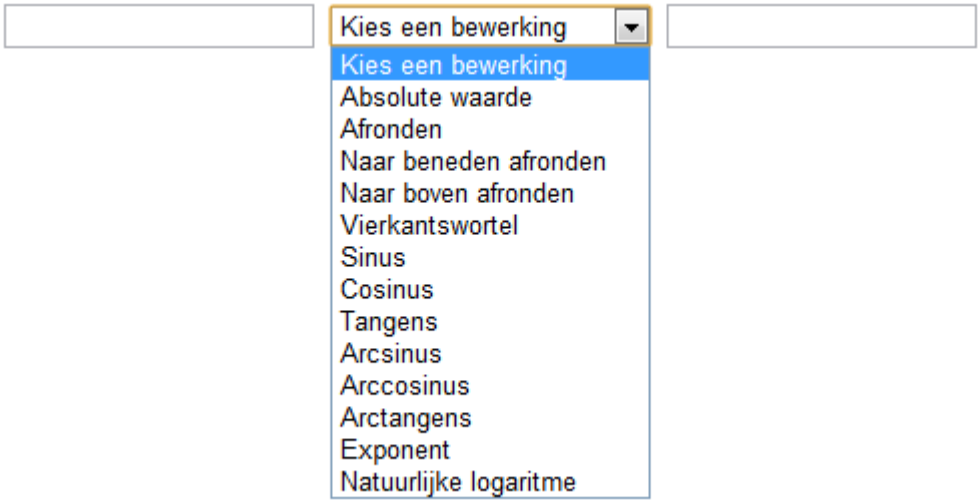
De methode familienaamInvoeren() definieert de familienaam van een persoon en kent deze aan de persoon toe. Door het gebruik van this kunnen we later meegeven over welke persoon het gaat. Om het persoon object moeder een familienaam toe te wijzen, gebruik je:

```
moeder.familienaamInvoeren("De Bakker");
```

Oefeningen

Wetenschappelijke rekenmachine

Maak een HTML pagina met een tekstveld, een selectielijst en een tekstveld.



Dit is de HTML code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Wetenschappelijke rekenmachine</title>
6   </head>
7   <body>
8   <input type="text" name="getal" id="getal" />
9   <select name="bewerking" id="bewerking"
     onchange="bereken(options[selectedIndex].value);">
10    <option>Kies een bewerking</option>
11    <option value="abs">Absolute waarde</option>
12    <option value="round">Afronden</option>
13    <option value="floor">Naar beneden afronden</option>
14    <option value="ceil">Naar boven afronden</option>
15    <option value="sqrt">Vierkantswortel</option>
16    <option value="sin">Sinus</option>
17    <option value="cos">Cosinus</option>
18    <option value="tan">Tangens</option>
19    <option value="asin">Arcsinus</option>
20    <option value="acos">Arccosinus</option>
21    <option value="atan">Arctangens</option>
22    <option value="exp">Exponent</option>
23    <option value="log">Natuurlijke logaritme</option>
24  </select>
25  <input type="text" name="resultaat" id="resultaat" />
26  </body>
27  </html>
```

Bij een nieuwe keuze in de selectielijst wordt de functie bereken gestart. Merk op hoe we hier de waarde van de gekozen optie achterhalen: selectedIndex bevat het rangnummer van de gekozen selectie (van 0 voor de eerste optie tot en met 13 voor de laatste optie), options[] bevat de adressen van alle opties, options[selectedIndex] bevat het adres van de gekozen optie en uiteindelijk bevat options[selectedIndex].value de waarde van de gekozen optie.

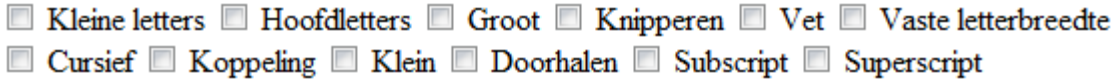
Schrijf de functie bereken() met de volgende kenmerken:

- plaats het opgegeven getal in het tekstveld in een variabele,
- voer de gekozen bewerking uit op het getal (gebruik daarvoor Math object methoden),

- plaats het resultaat in het laatste tekstveld.

Tekstopmaak

Maak een HTML pagina met selectievakjes met verschillende tekstopmaakmogelijkheden en daaronder een aanpasbare div tag.



De HTML code voor deze pagina is:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
 8" />
<title>Tekst opmaken</title>
</head>
<body>
<input type="checkbox" name="lower" id="lower" />
<label for="lower">Kleine letters</label>
<input type="checkbox" name="upper" id="upper" />
<label for="upper">Hoofdletters</label>
<input type="checkbox" name="big" id="big" />
<label for="big">Groot</label>
<input type="checkbox" name="blink" id="blink" />
<label for="blink">Knipperen</label>
<input type="checkbox" name="bold" id="bold" />
<label for="bold">Vet</label>
<input type="checkbox" name="fixed" id="fixed" />
<label for="fixed">Vaste letterbreedte</label>
<input type="checkbox" name="italics" id="italics" />
<label for="italics">Cursief</label>
<input type="checkbox" name="link" id="link" />
<label for="link">Koppeling</label>
<input type="checkbox" name="small" id="small" />
<label for="small">Klein</label>
<input type="checkbox" name="strike" id="strike" />
<label for="strike">Doorhalen</label>
<input type="checkbox" name="sub" id="sub" />
<label for="sub">Subscript</label>
<input type="checkbox" name="super" id="super" />
<label for="super">Superscript</label>
<div id="tekst"></div>
</body>
</html>
```

Bij het aanvinken van een selectievakje moet een tekst met de opgegeven opmaak in de aanpasbare div tag verschijnen. Aangezien we werken met selectievakjes moet je de verschillende opmaakmogelijkheden kunnen combineren. Om de noodzakelijke JavaScript code zo universeel mogelijk te maken, gebruiken we een object.

Schrijf een contructor functie voor ons object met de volgende eigenschappen:

- eigenschap om de tekst in te bewaren,
- twaalf eigenschappen om de adressen van de selectievakjes in te bewaren;

en een methode met de volgende kenmerken:

- de methode gebruikt een variabele die door String object methoden bewerkt worden als er een vinkje staat in het bijhorende selectievakje,
- het resultaat wordt geplaatst in het HTML element met het id dat met de methode werd meegegeven.

Maak op het einde van de body tag een object aan met de constructor.

Zorg dat bij het veranderen van een selectievakje de methode van het aangemaakte object wordt uitgevoerd.

Plaats de JavaScript code die op andere webpagina's gebruikt kan worden in een apart bestand.

JavaScript AJAX

AJAX = Asynchronous JavaScript and XML.

AJAX is geen nieuwe programmeertaal, maar een nieuwe techniek met bestaande standaarden.

AJAX is de kunst van het uitwisselen van gegevens met een server waardoor je zonder de volledige pagina te vernieuwen een gedeelte van een webpagina kunt aanpassen.

AJAX introductie

Welke kennis heb je nodig?

Je hebt een basiskennis nodig van:

- HTML / XHTML (cursus Dreamweaver)
- CSS (cursus Dreamweaver)
- JavaScript / DOM (deze cursus)

Wat is AJAX?

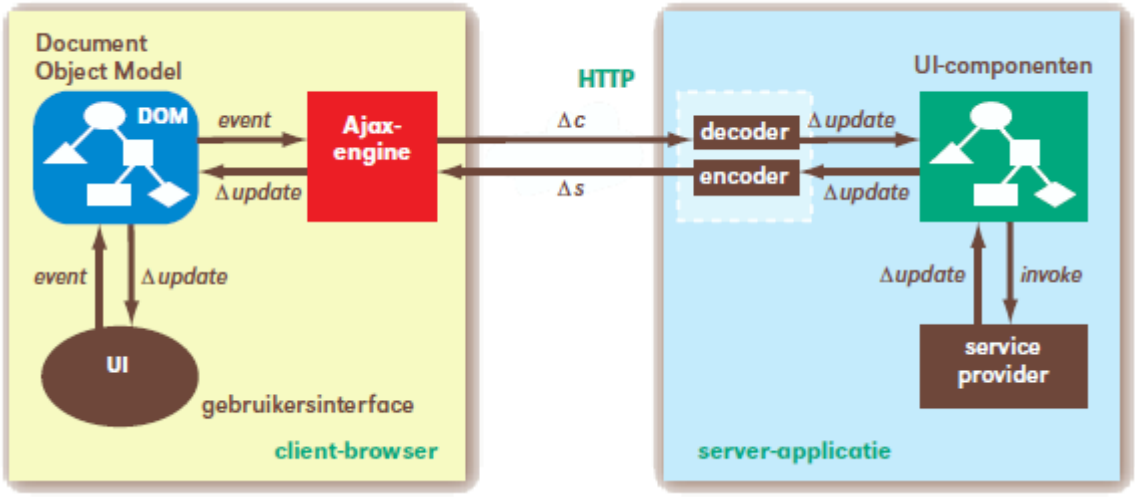
AJAX is een techniek waarmee je snelle en dynamische webpagina's kunt bouwen.

AJAX laat webpagina's toe om asynchroon gedeelten aan te passen door achter de schermen kleine pakketten met de server uit te wisselen. Dit betekent dat het mogelijk is gedeelten van een webpagina te vernieuwen zonder de volledige pagina te vernieuwen.

Klassieke pagina's, die geen AJAX gebruiken moeten voor de kleinste aanpassing de volledige pagina vernieuwen.

Gekende gebruikers van de AJAX techniek zijn: Google Maps, Gmail, Youtube, Facebook, enz.

Hoe werkt AJAX?



AJAX is gebaseerd op Internet Standaarden

AJAX is gebaseerd op internet standaarden en gebruikt een combinatie van:

- XMLHttpRequest objects om informatie asynchroon met de server uit te wisselen,
- JavaScript/DOM om op informatie te reageren en weer te geven,
- CSS om de informatie op te maken,
- XML wordt veel gebruikt als het formaat waarin de informatie wordt uitgewisseld.

AJAX toepassingen zijn browser- en platform-onafhankelijk.

AJAX voorbeeld

Om de snappen hoe AJAX werkt, maken we een eenvoudige AJAX toepassing.

Deze tekst aanpassen met AJAX

Tekst aanpassen

De AJAX toepassing bevat een div tag met een tekst en een knop.

De div tag wordt gebruikt om de informatie die we van de server ontvangen weer te geven. Een klik op de knop start de functie inhoudOphalen().

De HTML en JavaScript code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>AJAX voorbeeld</title>
6   <script type="text/javascript" src="ajax.js"></script>
7   <script type="text/javascript">
8   function inhoudOphalen(){
9     var inhoud = new ajaxObject("voorbeeld_aanpassing.html");
10    inhoud.callback = function (responseText, responseStatus,
     responseXML) {
```

```
11      if (responseStatus == 200) { // Testen op 0 voor locale test in
     Firefox, op 200 online
12        document.getElementById('aanpassen').innerHTML=responseText;
13      }
14    }
15    inhoud.update('','GET');
16  }
17  </script>
18  </head>
19  <body>
20  <div id="aanpassen"><h2>Deze tekst aanpassen met AJAX</h2></div>
21  <input type="button" name="button" id="button" value="Tekst
     aanpassen" onclick="inhoudOphalen();" />
22  </body>
23  </html>
```

AJAX XMLHttpRequest Object

De hoeksteen van AJAX is het XMLHttpRequest object.

Alle moderne browsers ondersteunen het XMLHttpRequest object (IE5 en IE6 gebruiken een ActiveXObject).

Het XMLHttpRequest object wordt gebruikt om achter de schermen gegevens met een server uit te wisselen.

Een XMLHttpRequest object aanmaken

Alle moderne browsers (IE7+, Firefox, Chrome, Safari en Opera) hebben een ingebouwd XMLHttpRequest object.

Om alle AJAX functionaliteit te groeperen, heb ik een eigen object geschreven met de volgende kenmerken (het object ajaxObject in het bestand ajax.js):

- ajaxObject werkt zowel op moderne browsers als op IE5 en IE6,
- ajaxObject werkt zowel met GET als POST,
- ajaxObject kan meerdere AJAX verbindingen (threads) tegelijk verwerken,
- ajaxObject heeft een methode om opgehaalde JavaScript code uit te voeren.

Zo'n XMLHttpRequestObject aanmaken doe je met de opdracht:

```
var inhoud = new ajaxObject("voorbeeld_aanpassing.html");
```

waarbij we het nieuwe object inhoud met de functie ajaxObject() aanmaken. De functie ajaxObject() krijgt als argument de URL van de op te halen gegevens.

De URL – een bestand op de server

Deze bevat in ons voorbeeld de volgende HTML code:

```
1   <p>AJAX is geen programmeertaal<br />
2   AJAX is een techniek waarmee je snelle en dynamische webpagina's
     maakt</p>
```

Dit bestand kan om het even welk soort bestand zijn zoals .txt, .html en .xml, of server scripts zoals .php die acties op de server kunnen uitvoeren voor ze een antwoord (bestand) sturen.

AJAX Callback functie

Aangezien de webserver niet altijd onmiddellijk op onze aanvraag reageert, zorgen we voor een callback functie. De callback functie wordt pas uitgevoerd nadat de gegevens van de server door het XMLHttpRequestObject werden ontvangen. In ons voorbeeld:

```
  inhoud.callback = function (responseText, responseStatus,
responseXML) {
    if (responseStatus == 200) { // Testen op 0 voor locale test in
Firefox, op 200 online
      document.getElementById('aanpassen').innerHTML=responseText;
    }
  }
```

Het object inhoud krijgt een callback functie door een methode aan het object toe te voegen.

Bij een **responseStatus** met een waarde 200 zijn alle gegevens van de server ontvangen en kunnen we beginnen met ze weer te geven.

Daar de gegevens van een webserver moeten komen, werkt dit alleen als alle onderdelen op het internet gepubliceerd zijn. Zolang de AJAX toepassing niet grondig getest is, moet je deze testen op een testserver of lokaal met Firefox. Firefox is de enige browser waarmee je AJAX toepassingen lokaal kunt testen. Om met Firefox lokaal te testen moet je de testwaarde responseStatus voor de test aanpassen naar 0 (m.a.w. 200 werkt alleen op het internet). Vergeet na de test niet de testwaarde terug op 200 te zetten.

De ontvangen gegevens werden door het XMLHttpRequestObject in de tekstvariabele **responseText** opgeslagen. De responseXML variabele bevat de ontvangen gegevens in het XML formaat.

De AJAX aanvraag naar de server zenden

Nu het object op de hoogte is van waar de gegevens op de server te vinden is (URL) en hoe ze verwerkt moeten worden (callback functie) kunnen we een AJAX aanvraag naar de server zenden zodat de server de gevraagde gegevens kan doorsturen. In ons voorbeeld:

```
inhoud.update('','GET');
```

De methode update van het aangemaakte object start de communicatie met de server. Deze verloopt volautomatisch op de achtergrond.

De update methode gebruikt twee argumenten:

- het eerste argument bevat gegevens voor de server; deze gegevens worden met serverscripts op de server verwerkt (zie cursus PHP),
- het tweede argument bepaald de gebruikte methode: GET of POST.

GET of POST?

GET is eenvoudiger en sneller dan POST en wordt in de meeste gevallen gebruikt.

Gebruikt altijd POST als:

- de gegevens altijd rechtstreeks van de server moeten komen (niet in een cache mogen worden opgeslagen) (bij het aanpassen van een bestand of database op de server),
- je grote hoeveelheden gegevens moet verzenden (GET kan maximum 1024 tekens doorsturen, POST kent geen limiet),
- versturen van gebruikersgegevens (waarin onbekende tekens kunnen staan), POST is robuuster en veiliger dan GET.

Met AJAX ontvangen JavaScript code uitvoeren

Als de opgehaalde gegevens JavaScript code bevat moet deze door de methode ajaxObject.javascript(id) uitgevoerd worden.

Dit gebeurt meestal in de callback functie na het weergeven van de ontvangen gegevens. De methode heeft het id nodig van de tag waarin de ontvangen JavaScript code in de webpagina wordt weergegeven (opgeslagen).

In ons voorbeeld zou dit met de opdracht

```
inhoud.javascript('aanpassen');
```

gebeuren.

Oefeningen

Tuingids

Met een selectielijst kunnen gebruikers kiezen welk artikel ze willen lezen. Daar niet elke gebruiker alle artikels zal lezen, gaan we de artikels pas laden als ze aangevraagd worden. Het laden van alle artikels in één pagina zou de bestandsgrootte van de webpagina zo groot maken dat de webpagina zeer traag zou laden. M.a.w. we gaan AJAX gebruiken om de webpagina snel te houden en de artikels pas te laden en weer te geven als de de gebruiker er om vraagt.

Maak een webpagina met een lichtgroene achtergrond met:

- een selectielijst,
- daaronder een aanpasbare div.

De HTML code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Tuingids</title>
6   </head>
7   <body style="background-color:#F4FAEC">
8   <select name="tipkeuze" size="1" style="margin-left:10%">
9     <option>Kies een artikel uit onze Tuingids</option>
10    <option value="tuingids/bolgewassen/blauwedruif.html">Bruisende
     blauwe druif, Muscari</option>
11    <option value="tuingids/bolgewassen/bloembollen.html">Bloembollen:
     planten in het voorjaar</option>
12    <option
     value="tuingids/bolgewassen/bollenbalkon.html">Bollenbalkon: Een,
     twee, drie.. en het is lente!</option>
13    <option
     value="tuingids/bolgewassen/bollenmode.html">Interieurtrends:
     Bollenmode</option>
14    <option value="tuingids/bolgewassen/galtonia.html">Geurende
     Galtonia, Kaapse hyacint</option>
15    <option value="tuingids/bolgewassen/lenteborder.html">Kleurrijke
     lenteborder</option>
16    <option value="tuingids/bolgewassen/parkiettulpen.html">Extra
     vrolijke tulpen: Parkiettulpen</option>
17    <option value="tuingids/bolgewassen/tulpen.html">Tulpentrends:
     Veelzijdige tulpen</option>
```

```
18  </select>
19  <div id="inhoud"></div>
20  </body>
21  </html>
```

Zorg ervoor dat de webpagina AJAX kan gebruiken.

Zorg ervoor dat bij het maken van een keuze in de selectielijst een functie wordt uitgevoerd:

- de functie gebruikt AJAX om een onderdeel van de webpagina in een bestand op de webserver te laden (de URL van het bestand is de waarde van de geselecteerde optie),
- het met AJAX opgehaalde bestand moet weergeven worden in de aanpasbare div tag.

Voeg de artikels in zomerbollen.html en bloesempracht.html toe aan de selectielijst.

Testen na publicatie

Publiceer alle onderdelen van deze oefening naar de webserver van de school.

Gebruik daarvoor de volgende gegevens:

- Verbinding maken via: FTP
- FTP-adres: ..... (vragen aan leerkracht)
- Gebruikersnaam: pcXX (waarbij je XX moet vervangen door het cijfer op uw computer, vb: pc01)
- Wachtwoord: snt+456
- Passieve FTP gebruiken
- Je kunt de publicatie testen door te surfen naar http://...../pcXX/webpagina.html.

Dynamisch keuzemenu

We maken een dynamische pagina waarbij een keuze in een selectielijst een bijhorende selectielijst laat verschijnen, waardoor de gebruiker steeds dichter bij zijn doel komt. Vooral bij een grote diversiteit is dit de meest aangeraden manier van werken.

Maak een webpagina met een tabel met één rij en twee kolommen.

De eerste cel bevat:

- een Kop 2 met de tekst Informatica,
- daaronder een alinea met een selectielijst met de opties Richting, Webdesign en Kantoor; de optie Webdesign heeft als waarde keuzemenu/webdesign.html en de optie Kantoor de waarde keuzemenu/kantoor.html;
- daaronder een aanpasbare alinea voor de selectielijst met het vakmenu;
- daaronder een aanpasbare alinea voor de selectielijst met het graadmenu.

De tweede aanpasbare cel zal de inhoud van de gekozen cursus weergeven.

De HTML code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Dynamisch keuzemenu</title>
6   </head>
7   <body>
8   <table border="0">
```

```
 9     <tr>
10       <td valign="top"><h3>Informatica</h3>
11           <p><select name="richting" id="richting">
12               <option>Richting</option>
13               <option
    value="keuzemenu/webdesign.html">Webdesign</option>
14               <option value="keuzemenu/kantoor.html">Kantoor</option>
15             </select></p>
16             <p id="vakmenu"></p>
17             <p id="graadmenu"></p>
18       </td>
19       <td valign="top" id="inhoud"></td>
20     </tr>
21   </table>
22   </body>
23   </html>
```

Zorg ervoor dat de webpagina AJAX kan gebruiken.

Zorg ervoor dat bij het maken van een keuze in de selectielijst een functie wordt uitgevoerd:

- de functie gebruikt AJAX om een onderdeel van de webpagina in een bestand op de webserver te laden (de URL van het bestand is de waarde van de geselecteerde optie),
- het met AJAX opgehaalde bestand moet weergeven worden in een tag met een id die je met de functie meegeeft.

Het bestand voor de optie Kantoor bevat een selectielijst:

- met drie opties:
  - ✓ Tekstverwerking met de waarde keuzemenu/kantoor/tekstverwerking.html,
  - ✓ Rekenblad met de waarde keuzemenu/kantoor/rekenblad.html,
  - ✓ Database met de waarde keuzemenu/kantoor/database.html,
- en roept bij een selectie de reeds geschreven functie aan.

Het bestand voor de optie Tekstverwerking bevat een selectielijst:

- met twee opties:
  - ✓ Deel 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Deel 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
- en roept bij een selectie de reeds geschreven functie aan.

Het bestand voor de optie Rekenblad bevat een selectielijst:

- met vier opties:
  - ✓ Deel 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Deel 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Programmeren 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Programmeren 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
- en roept bij een selectie de reeds geschreven functie aan.

Het bestand voor de optie Database bevat een selectielijst:

- met vijf opties:
  - ✓ Deel 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Deel 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,

- ✓ Programmeren 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
- ✓ Programmeren 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
- ✓ Programmeren 3 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,

- en roept bij een selectie de reeds geschreven functie aan.

Het bestand voor de optie Webdesign bevat een selectielijst:

- met drie opties:
  - ✓ Dreamweaver,
  - ✓ JavaScript,
  - ✓ PHP,
- en roept bij een selectie de reeds geschreven functie aan.

Het bestand voor de optie Dreamweaver bevat een selectielijst:

- met vier opties:
  - ✓ Deel 1 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Deel 2 met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
  - ✓ Eindwerk met als waarde de URL naar een reeds bestaande webpagina met de inhoud,
- en roept bij een selectie de reeds geschreven functie aan.

Bij het bestand voor de opties JavaScript en PHP heb je een probleem, daar zijn namelijk geen graden. M.a.w. De inhoud van de cursus moet direct zichtbaar worden. Dit kan enkel door in de HTML code een stuk JavaScript op te nemen. De JavaScript code moet zonder enige actie vanwege de gebruiker de juiste inhoud tonen.

Zorg ervoor dat deze JavaScript code door AJAX wordt uitgevoerd.

Testen na publicatie

Publiceer alle onderdelen van deze oefening naar de webserver van de school. Test de gepubliceerde webpagina.

Foto's Brugge

Moderne browsers zijn veel sneller dan hun voorgangers. Deze snelheidswinst in deels te wijten aan een techniek die als twee druppels water lijkt op AJAX. Moderne browsers gaan alle externe bestanden die op de webpagina (in de body tag) gebruikt worden tegelijkertijd laden na het weergeven van de webpagina zonder de externe bestanden. M.a.w. alle HTML onderdelen die in het HTML bestand zitten worden weergegeven en pas daarna worden de afbeeldingen geladen en weergegeven. Je kunt dan reeds beginnen met het lezen van de tekst terwijl de afbeeldingen en reclameboodschappen nog aan het laden zijn.

In de volgende oefening gaan we dus geen gebruik maken van AJAX, maar van de ingebouwde laadtechnieken van moderne browsers.

Maak een webpagina met een Kop 1 (h1) met de tekst Brugge in beeld.

Daaronder plaats je een aanpasbare span tag met de volgende CSS stijl:

- absoluut gepositioneerd.

Daaronder plaats je achttien span tags met de weer te geven foto's met een hoogte van 135 beeldpunten. Deze hoogte zorgt ervoor dat alle foto's onafhankelijk van hun werkelijke grootte even hoog en zonder vervorming worden weergegeven.

De HTML code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-
 8" />
<title>Foto's Brugge</title>
</head>
<body>
<h1>Brugge in beeld</h1>
<span id="aandacht" style="position:absolute"></span>
<span><img src="fotos/begijnenkerk.jpg" height="135" /></span>
<span><img src="fotos/begijnhof.jpg" height="135" /></span>
<span><img src="fotos/belfort_1.jpg" height="135" /></span>
<span><img src="fotos/belfort_2.jpg" height="135" /></span>
<span><img src="fotos/de_halve_maan_1.jpg" height="135" /></span>
<span><img src="fotos/de_halve_maan_2.jpg" height="135" /></span>
<span><img src="fotos/grote_markt.jpg" height="135" /></span>
<span><img src="fotos/minnewater_park.jpg" height="135" /></span>
<span><img src="fotos/onze_lieve_vrouwekerk.jpg" height="135"
 /></span>
<span><img src="fotos/onze_lieve_vrouwekerk_binnen.jpg" height="135"
 /></span>
<span><img src="fotos/provinciaal_hof.jpg" height="135" /></span>
<span><img src="fotos/rijen.jpg" height="135" /></span>
<span><img src="fotos/sint_salvatorskathedraal.jpg" height="135"
 /></span>
<span><img src="fotos/stadhuis_1.jpg" height="135" /></span>
<span><img src="fotos/stadhuis_2.jpg" height="135" /></span>
<span><img src="fotos/van_eyck_plein.jpg" height="135" /></span>
<span><img src="fotos/walplein.jpg" height="135" /></span>
<span><img src="fotos/windmolen.jpg" height="135" /></span>
</body>
</html>
```

Zorg dat bij het aanwijzen met de muis van een span tag met een foto een functie wordt uitgevoerd waarmee we de foto ter plaatse wat groter weergeven. Deze functie heeft de volgende kenmerken:

- De functie ontvangt het adres van de aangewezen tag.
- Berekent de positie van de tag en verschuiven die positie telkens tien beeldpunten naar boven en naar links:

```
  links = adres.offsetLeft - 10;
  boven = adres.getElementsByTagName('img')[0].offsetTop - 10; // fout
in IE en Firefox
```

  Bij de berekening van de y coördinaat in de variabele boven gebruik je de positie van de eerste in de span tag aanwezige img tag omdat de berekening van de y coördinaat van de span tag in Internet Explorer en Firefox foutieve resultaten oplevert.

- Deze positie geef je door aan de absoluut geplaatste span tag met het id aandacht. Dit gaat via een CSS stijl voor de x positie met:

```
  document.getElementById('aandacht').style.left = links + "px";
```

- Pas op een gelijkaardige manier de y positie (CSS stijl top) aan.
- Kopieer de inhoud van de aangewezen span tag naar de span tag met het id aandacht.
- Vervang in de inhoud van de span tag met het id aandacht de hoogte van de afbeelding van 135 door 155 (Tip: denk aan een String object methode).

Maak bij het met de muis verlaten van de span tag met het id aandacht deze tag leeg, waardoor deze niet meer in de browser wordt weergegeven.

Bij het aanwijzen van elke foto wordt deze nu iets groter weergeven.

De foto centraal groter weergeven

Plaats op de webpagina een aanpasbare span tag voor het weergeven van een grotere foto. Deze span tag gebruikt CSS om de inhoud rechts uit te lijnen, wordt absoluut geplaatst, heeft een zwarte achtergrond en witte tekst. De HTML code:

```
<span id="groter" style="text-align: right; position:absolute;
background-color: #000; color: #FFF;"></span>
```

Zorg dat bij het klikken op de span tag met het id aandacht een functie wordt uitgevoerd. Deze functie heeft de volgende kenmerken:

- Plaats in de span tag met het id groter de HTML code

```
<span style="cursor: pointer;">Originele resolutie</span><span
style="cursor: pointer;" title='Afsluiten'> X </span><br />
```

  gevolgd door de inhoud van de span tag met het id aandacht. De extra HTML code zorgt voor twee tekstknoppen.

- Om de foto's op normale grootte weer te geven moet je de eigenschap height van de afbeelding in de span tag met het id groter verwijderen, dit kan met:

```
document.getElementById("groter").getElementsByTagName('img')
[0].removeAttribute("height");
```

De foto centreren

Schrijf een functie waarmee je een absoluut geplaatste span tag met id kunt centreren als je weet dat:

- window.innerHeight (Width) de hoogte (breedte) van het browservenster teruggeeft,
- adres.offsetHeight (Width) de hoogte (breedte) van een tag met een adres teruggeeft,
- adres.style.top (left) de CSS stijl is waarmee je de afstand tussen de bovenste (linker) rand van het browservenster en de bovenste (linker) rand van de tag met een adres bepaald.

Gebruik deze functie om de grote foto te centreren.

Bij het klikken op een foto wordt deze nu groter in het midden van de webpagina weergegeven.

Afsluiten

Zorg dat bij het klikken op de tekstknop X een functie wordt uitgevoerd waarmee de grote foto van het scherm verdwijnt.

Maximaliseren

Zorg dat bij het klikken op de tekstknop Originele resolutie een functie wordt uitgevoerd die de volgende aanpassingen aan de span tag met het id groter uitvoert:

- De knoptekst 'Originele resolutie' wordt aangepast naar 'Verkleinen'.
- I.p.v. de foto's uit de map fotos weer te geven, moeten de foto's uit de map fotosgroot worden weergegeven. Daarbij hebben de foto's dezelfde bestandsnamen.
- Met de volgende opdracht kan je aan de img tag in de span tag met het id groter het onload event aanpassen om na het laden van de grote foto deze te centreren:

```
document.getElementById('groter').getElementsByTagName('img')[0].onload
= function() { centreer(); };
```

Verkleinen

Zorg dat bij het klikken op de tekstknop Verkleinen een functie wordt uitgevoerd waarmee de foto in de span tag met het id groter terug de grote niet gemaximaliseerde foto gecentreerd op de webpagina toont. Vergeet de tekstknop niet aan te passen.

Afwerken

Werk het geheel af door bijvoorbeeld:

- de JavaScript functies zo universeel mogelijk te maken en in een afzonderlijk bestand te plaatsen.
- de tekstknoppen te vervangen door echte knoppen (formulierknoppen of afbeeldingen),
- de grotere foto's langzaam te vergroten of te verkleinen,
- de grotere foto's langzaam op te laten komen (van doorzichtig tot volledig zichtbaar),
- tijdens het laden van een originele foto een animated gif tonen.

Voor wie van een uitdaging houdt:

- toevoegen van knoppen voor de volgende en vorige foto's,

JavaScript Cookies

Een cookie wordt dikwijls gebruikt om een gebruiker te identificeren.

Naar aanleiding van een Europese privacyrichtlijn, is het gebruik van cookies aan bepaalde regels onderworpen. Volgens deze richtlijn moeten gebruikers per site het volgen van de gebruiker kunnen uitschakelen. Sommige landen waaronder België en Nederland gaan nog verder en eisen dat websites toestemming vragen voordat ze niet-essentiële cookies plaatsen. Meer informatie kun je vinden op http://www.zdnet.be/help/140749/wat-moet-je-als-sitebeheerder-doen-voor-cookiewet-/.

Wat is een Cookie?

Een cookie is een variabele die op de computer van de gebruiker wordt opgeslagen. Telkens een webpagina op de computer wordt geopend, zal de bijhorende cookie actief worden. Met JavaScript kun je zowel cookies aanmaken als uitlezen.

Een paar voorbeelden:

- Naam cookie – Bij het eerste bezoek aan een webpagina, wordt gevraagd een naam op te geven. De opgegeven naam wordt in een cookie opgeslagen. Bij een volgend bezoek aan de webpagina wordt de cookie uitgelezen en verschijnt op de website een persoonlijke begroeting (Welkom terug, Dany).
- Datum cookie – Bij het eerste bezoek aan een webpagina wordt de bezoekdatum (huidige datum) in een cookie opgeslagen. Bij een volgend bezoek wordt uw laatste bezoek op de webpagina weergegeven (U heeft onze site voor het laatst bezocht op dinsdag 11 september 2012). De vorige bezoekdatum wordt uit een cookie uitgelezen.

Een Cookie aanmaken en opslaan

In het voorbeeld maken we een cookie met de naam van de bezoeker. Bij het eerste bezoek aan de webpagina wordt gevraagd een naam in te geven. De naam wordt daarna in een cookie opgeslagen. Bij een volgend bezoek aan de webpagina wordt een persoonlijk begroeting weergeven.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
    8" />
5  <title>Cookies</title>
6  <script type="text/javascript">
7  function setCookie(c_naam, waarde, dagen){
8  /* Deze functie slaat de naam van de gebruiker op in een cookie
9  De argumenten van de functie bevatten de naam van de cookie, de
    inhoud (waarde) van de cookie en het aantal dagen dat de cookie
    geldig blijft*/
10  var vervaldatum = new Date();
11 // Bereken de vervaldatum van de cookie
12  vervaldatum.setDate(vervaldatum.getDate() + dagen);
13 // Stel de inhoud van de cookie samen
14 // de escape opdracht zorgt dat niet conventionele karakters correct
    worden opgeslagen
15 // cookies gebruiken datums in het UTC standaardformaat
16  var c_waarde = escape(waarde) + ((dagen == null) ? "" : ";
    expires=" + vervaldatum.toUTCString());
17 // Maak de cookie aan (sla de cookie op)
18  document.cookie = c_naam + "=" + c_waarde;
19 }
20 function getCookie(c_naam){
21 /* Deze functie maakt een array van de cookie namen en waarden,
22 controleert of de cookie naam bestaat en
23 geeft de inhoud van de cookie terug */
24  var i, x, y, ARRcookies;
25 // splits de inhoud van de cookies op en plaats deze in een array
26  ARRcookies = document.cookie.split(";");
27 // doorloop de array met cookies
28  for (i = 0; i < ARRcookies.length; i++){
29   // de naam van de cookie staat voor het = teken
30   x = ARRcookies[i].substr(0, ARRcookies[i].indexOf("="));
31   // de waarde van de cookie staat na het = teken
32   y = ARRcookies[i].substr(ARRcookies[i].indexOf("=")+1);
33   // verwijder alle witruimte voor (^\s+) en achter de naam van de
    cookie (\s+$)
34   x = x.replace(/^\s+|\s+$/g,"");
35   // controleer of de cookie met de opgegeven naam bestaat
36   if (x == c_naam){
37         // geef de waarde van de gevonden cookie terug
38         return unescape(y);
39   }
40  }
41 }
42 function checkCookie(){
43 /* Deze functie geeft een begroeting weer als de cookie bestaat,
44 of vraagt naar de gebruikersnaam indien de cookie niet bestaat*/
45 // lezen van de cookie variabele
46  var gebruikersnaam = getCookie("gebruikersnaam");
47  if (gebruikersnaam != null && gebruikersnaam != ""){
48   // bij een geldige gebruikersnaam, de begroeting weergeven
49   document.getElementById("welkom").innerHTML = gebruikersnaam + ",
    wij heten u terug van harte welkom op onze webpagina.";
```

```
50  } else { // bij een ongeldige gebruikersnaam, naar een nieuwe
    gebruikersnaam vragen
51   gebruikersnaam = prompt("Indien gewenst, kunt u hier uw naam
    ingeven:","");
52   if (gebruikersnaam != null && gebruikersnaam != ""){
53        // bij een geldige gebruikersnaam een cookie aanmaken
54        setCookie("gebruikersnaam", gebruikersnaam, 365);
55   }
56   }
57  }
58  </script>
59  </head>
60  <body onload="checkCookie();">
61  <h3 id="welkom">Welkom op onze webpagina.</h3>
62  </body>
63  </html>
```

Na het laden van de pagina in de browser (onload) wordt de cookie gelezen of aangemaakt.

Om een cookie een andere waarde te geven, overschrijf je de oude cookie met de nieuwe waarde.

Om een cookie te wissen, maak je een cookie zonder waarde ("") en zonder vervaldatum (0), de cookie wordt dan bij het sluiten van de browser of het verlaten van de site opgeruimd.

Oefeningen

Laatst bezocht op

Pas het voorbeeld aan zodat de begroeting steeds de datum van het laatste bezoek weergeeft.

Sla de de datum in de cookie op in het UTC standaardformaat.

De datum wordt in de taal van de browser weergegeven.

Uw laatste bezoek dateert van zondag 16 september 2012 19:08:47

Aantal bezoeken

Pas het script aan zodat het aantal bezoeken aan de site wordt bijgehouden.

U bezocht onze site reeds 3 maal. Uw laatste bezoek dateert van zondag 16 september 2012 19:53:47

Privacy

Plaats een knop op de webpagina om de gegevens te wissen en geen gegevens meer bij te houden.

Zorg dat ook bij een volgend bezoek aan de webpagina geen gegevens worden bijgehouden.

Om de webpagina na het in- of uitschakelen van de privacy te vernieuwen, gebruik je de volgende code:

```
location.reload();
```

Als de privacy is ingeschakeld wordt in de knop de tekst **Privacy uitschakelen** weergegeven.

U bezocht onze site reeds 2 maal. Uw laatste bezoek dateert van zondag 16 september 2012 20:24:26

Privacy inschakelen

Persoonlijk thema

Om de gebruiker (surfer) in staat te stellen zijn eigen webthema te gebruiken, maak je een pagina met een formulier waarmee verschillende onderdelen van de pagina een eigen kleurenpalet krijgt.

Maak de volgende HTML pagina:

Persoonlijk thema samenstellen

Onderdeel	Achtergrondkleur	Voorgrondkleur
Pagina		
Kop 1		
Kop 2		
	OK	Annuleren

Voorbeeldweergave

Deze tekst verschijnt in de voorgrondkleur van de pagina.

Subtitels

Titels en subtitels worden opgemaakt met Koppen. Met koppen opgemaakte teksten krijgen een structuur waardoor deze beter overkomen. Daarnaast worden deze koppen door zoekmachines op het internet gebruikt om belangrijke zoekwoorden (in koppen) te onderscheiden van minder belangrijke zoekwoorden (in de leestekst).

De teksten **Persoonlijk thema samenstellen** en **Voorbeeldweergave** gebruiken het Kop 1 formaat (h1 tag).

De tekst **Subtitel** gebruikt het Kop 2 formaat.

Schrijf een functie waarmee na het aanpassen van een tekstveld de overeenkomstige CSS stijl aangepast wordt.

- Om de voorgrondkleur van alle elementen met een bepaalde tag aan te passen, kun je de volgende functie gebruiken:

```
function inputChanged (tag, kleur) {
  // alle elementen met een bepaalde tag in een pagina opzoeken en in
een array opslaan
  var x = document.getElementsByTagName(tag);
  // alle array elementen doorlopen
  for( var tagelement = 0; tagelement < x.length; ++tagelement) {
    // de voorgrondkleur van een element instellen
    x[tagelement].style.color = "#" + kleur;
  }
```

```
}
```

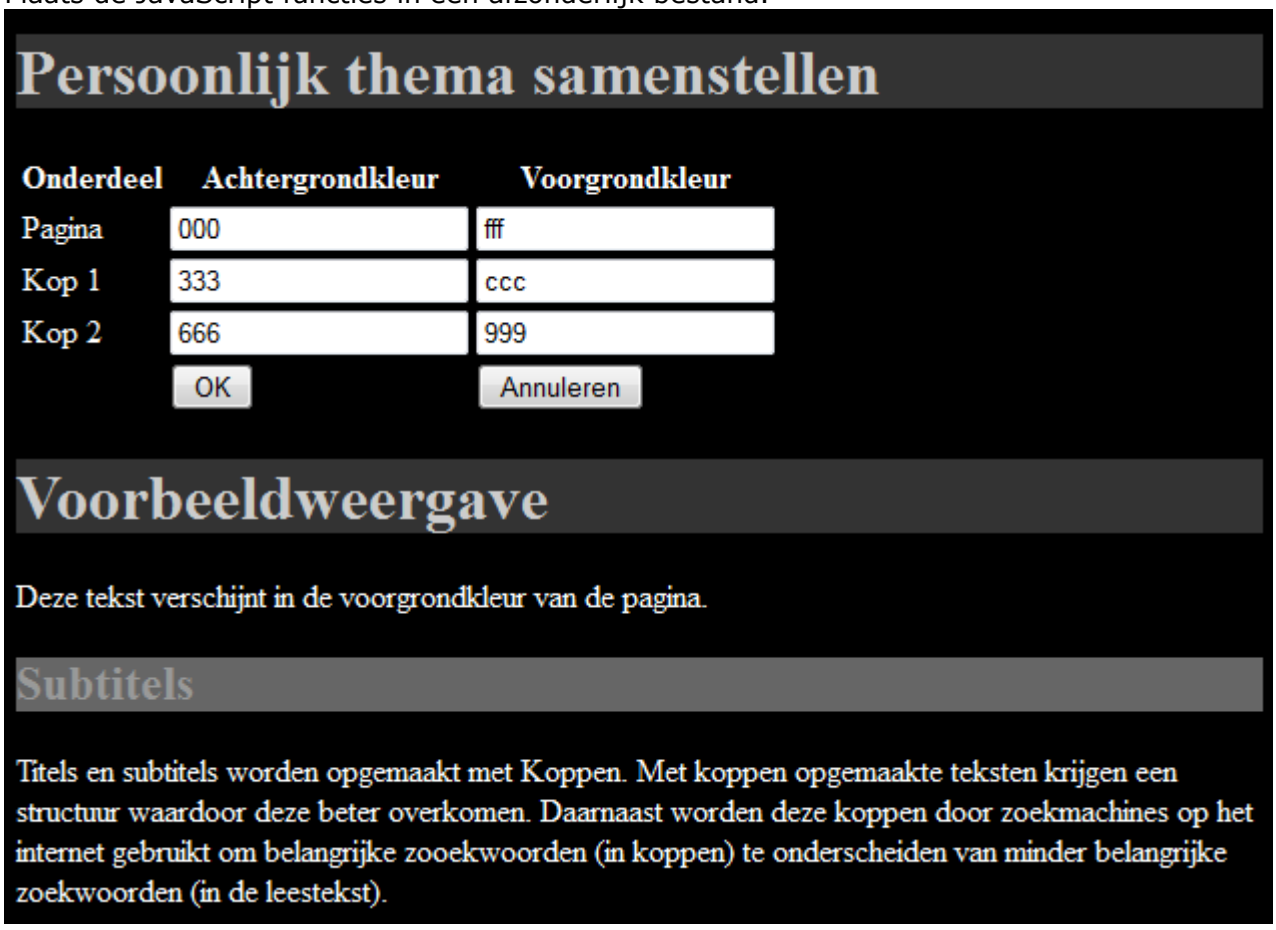
- Pas deze functie aan zodat je ook een achtergrondkleur van alle elementen met een bepaalde tag kunt aanpassen.

Schrijf een functie om bij het klikken op de knop Annuleren de inhoud van alle tekstvelden te wissen en alle opmaak van de pagina en de twee koppen te wissen.

Schrijf een functie om bij het klikken op de knop OK de inhoud van alle tekstvelden in cookies op te slaan.

Indien bij het laden van de pagina reeds cookies met thema-instellingen aanwezig zijn, moeten deze toegepast worden en in de velden van het formulier ingevuld worden.

Plaats de JavaScript functies in een afzonderlijk bestand.



Maak een tweede pagina waarin zonder formulier, met wat tekst, een kop 1 en enkele kop 2 subtitels. Deze pagina maakt gebruik van de themakleuren in de cookies.

Om de pagina met het formulier af te werken, surf je naar http://jscolor.com en gebruik je de op deze pagina beschreven kleurenkiezer.

JavaScript Timing Events

Met JavaScript kun je om de zoveel tijd bepaalde code laten uitvoeren. Dit noemen we timing events.

Om in JavaScript getimede gebeurtenissen te gebruiken, bestaan er twee methoden:

- setInterval() – voert met een bepaalde tijdsinterval een bepaalde functie uit (blijft maar doorgaan)
- setTimeout() – voert na een opgegeven tijd een bepaalde functie één keer uit

De setInterval() methode

De setInterval methode wacht een aantal milliseconden en voert daarna een bepaalde functie uit. Na de opgegeven intervaltijd, wordt de opgegeven functie telkenmale herhaald.

Syntaxis:

```
setInterval("JavaScript functie",milliseconden);
```

Het eerste argument is de uit te voeren functie.

Het tweede argument bepaalt het tijdsinterval tussen het herhaaldelijk uitvoeren van de functie. Dit tijdsinterval wordt uitgedrukt in milliseconden (1/1000 van een seconde).

Een voorbeeld: timing/interval.html

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>JavaScript Interval</title>
6   <script type="text/javascript">
7   function startKlok(){
8     setInterval("klok()", 1000);
9   }
10  function klok(){
11    var datum = new Date();
12    var tijd = datum.toLocaleTimeString();
13    document.getElementById("klok").innerHTML = tijd;
14  }
15  </script>
16  </head>
17  <body onload="startKlok();">
18  <p id="klok"></p>
19  </body>
20  </html>
```

Dit voorbeeld geeft de tijd weer. Daarbij wordt de setInterval() methode gebruikt om elke seconden de tijd telkens opnieuw weer te geven.

Stoppen

De clearInterval() methode wordt gebruikt om de setInterval() methode te onderbreken en het uitvoeren van de bijhorende functie te stoppen.

Syntaxis:

```
clearInterval(intervalVariabele)
```

Om de clearInterval() methode te kunnen gebruiken, moet je bij het activeren van een setInterval() methode een globale variabele gebruiken.

Syntaxis:

```
intervalVariabele = setInterval("JavaScript functie",milliseconden);
```

Een voorbeeld: timing/interval.html

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
```

```
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>JavaScript Interval</title>
6   <script type="text/javascript">
7   var klokInterval;
8   function startKlok(){
9     klokInterval = setInterval("klok()", 1000);
10  }
11  function klok(){
12    var datum = new Date();
13    var tijd = datum.toLocaleTimeString();
14    document.getElementById("klok").innerHTML = tijd;
15  }
16  </script>
17  </head>
18  <body onload="startKlok();">
19  <p id="klok"></p>
20  <input type="button" onclick="clearInterval(klokInterval);"
     value="Klok stoppen" />
21  </body>
22  </html>
```

De setTimeout() methode

De setTimeout() methode wacht een aantal milliseconden en voert daarna de opgegeven functie uit.

Syntaxis:

```
setTimeout("JavaScript functie",milliseconden);
```

Het eerste argument bevat de uit te voeren functie.

Het tweede argument bepaalt na hoeveel milliseconden de opgegeven functie wordt uitgevoerd.

Een voorbeeld timing/timeout.html

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>JavaScript Timeout</title>
6   <script type="text/javascript">
7   var klokTimeout;
8   function klok(){
9     var datum = new Date();
10    var tijd = datum.toLocaleTimeString();
11    document.getElementById("klok").innerHTML = tijd;
12    klokTimeout = setTimeout("klok()", 1000);
13  }
14  </script>
15  </head>
16  <body onload="klok();">
17  <p id="klok"></p>
18  <input type="button" onclick="clearTimeout(klokTimeout);"
     value="Klok stoppen" />
```

```
19  </body>
20  </html>
```

Stoppen

De clearTimeout() methode onderbreekt een setTimeout() methode.

Syntaxis:

```
clearTimeout(timeoutVariabele)
```

Om de clearTimeout() methode te kunnen gebruiken, moet je bij het activeren van een setTimeout() methode een globale variabele gebruiken.

Syntaxis:

```
timeoutVariabele = setTimeout("JavaScript functie",milliseconden);
```

Oefeningen

Aftelklok

We gaan een webpagina maken met een aftelklok in seconden tot het volgende nieuwe jaar.

Maak een HTML pagina met de volgende tekst:

Nog … seconden tot Nieuwjaar 2013

Vervang de puntjes door een span tag met een id.

Schrijf de Javascript code om in de span tag een afteller in seconden tot het nieuwe jaar weer te geven:

- var nieuwjaardatum = new Date("2013"); maakt een datum object van de eerste seconde van het jaar 2013.
- nieuwjaardatum.valueOf() berekent hoeveel milliseconden er liggen tussen nieuwjaardatum en 1 januari 1970 middernacht.

Leesbaarder presenteren

Zorg dat de afteller het aantal dagen, uren, minuten en seconden weergeeft (voorbeeld: Nog 100 dagen 6 uren 8 minuten 20 seconden tot Nieuwjaar 2013):

- enkel de eenheden verschillend van nul mogen weergegeven worden

Interactie toevoegen

Vervang het jaartal door een tekstveld waarin de gebruiker zelf een jaartal mag invullen:

- geef het tekstveld een standaard value (jaartal)
- bij het aanpassen van het jaartal met de afteller automatisch aangepast worden
- bij het opgeven van een jaartal in het verleden, moet een melding verschijnen en wordt het aftellen stil gelegd:
  - ✓ datum.getFullYear() geeft het jaartal van een datum object weer.

Vloeiend

Deze opdracht toont hoe je met behulp van wiskunde vloeiende bewegingen kunt weergeven. Aangezien ik jullie de wiskunde wil besparen, gebruiken we de bestaande bibliotheek SmoothMovement.js.

Paneel openen

Maak de webpagina timing/vloeiend.html:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Vloeiende bewegingen</title>
6   </head>
7   <body>
8   <span style="cursor:pointer;" id="knop">&raquo;</span>
9   <div id="jaargetijden" style="overflow:hidden; width:0px;
     background-color:#000;">
10  <img src="lente.jpg" width="400" height="315" /><br />
11  <img src="zomer.jpg" width="400" height="315" /><br />
12  <img src="herfst.jpg" width="400" height="315" /><br />
13  <img src="winter.jpg" width="400" height="315" />
14  </div>
15  </body>
16  </html>
```

Voeg in de head tag het externe script SmoothMovement.js toe.

Zorg dat door het klikken op de span tag met het id knop de functie uitschuiven() wordt uitgevoerd.

De volgende JavaScript code zorgt voor het vloeiend openen van het paneel:

```
// globaal object met tijdstipberekeningen
var uitschuifbeweging;
// globaal object met het adres van het paneel
var uitschuifelement;
// klik functie
function uitschuiven() {
  // opzoeken adres van het paneel
  uitschuifelement = document.getElementById('jaargetijden');
  // object met beweging van punt 0 tot 400 aanmaken (hier de breedte
van het paneel)
  uitschuifbeweging = new SmoothMovement(0, 400);
  // de beweging vloeiend animeren
  // 20 bepaalt de snelheid (< is sneller)
  // updateUitschuiven wordt uitgevoerd op de berekende tijdstippen
  // stopUitschuiven wordt uitgevoerd bij het beëindigen van de
animatie
  uitschuifbeweging.animate(20, updateUitschuiven, stopUitschuiven);
}
function updateUitschuiven(frame){
  // frame bevat de positie van de vloeiende beweging (hier tussen 0 en
800)
  uitschuifelement.style.width = frame + "px";
}
function stopUitschuiven(){
  // functie wordt uitgevoerd na het beëindigen van de vloeiende
beweging
  document.getElementById("knop").innerHTML = "&laquo;";
}
```

Vul deze JavaScript code aan zodat het paneel bij een tweede klik op de knop gesloten wordt. M.a.w. De knop werkt als een aan/uit schakelaar.

Foto's verkleinen

Zorg dat bij het laden van de afbeeldingen de functie verklein(this) wordt uitgevoerd:

- De originele afmetingen van de foto's kan je opvragen met adres.width en adres.height (adres bevat het adres van de img tag met de foto).
- Voeg deze afmetingen met een underscore samen tot één tekenreeks en sla deze op in het id van de img tag van de foto (adres.id).
- Verklein de breedte en hoogte van de foto's tot één vierde van de originele afmetingen.

Foto's zoomen

Zorg dat bij het klikken op een verkleinde foto de functie zoom(this) wordt uitgevoerd:

- Deze functie zorgt dat de originele afmetingen met een vloeiende overgang hersteld worden.
- De originele afmetingen haal je uit het id van de foto.

Zorg dat klikken op een grote foto, de foto-afmetingen met een vloeiende beweging terug op één vierde van de originele afbeeldingen brengt. M.a.w. Zorg voor een aan/uit werking bij het klikken op een foto.

Overvloeiende foto's

We maken van de vier jaargetijde foto's één overvloeiende foto.

Zorg dat de foto's naast elkaar worden weergegeven.

Voeg in de head tag het externe script Fader.js toe.

Maak van de div tag met de vier foto's een overvloeiende foto met de code:

```
var fader = new Fader('jaargetijden');
```

Activeer het automatisch overvloeien met de code:

```
fader.setInterval(5);
```

Bij het verkleinen van de foto's voeg je het adres van de foto's toe aan een array.

Pas de zoom functie aan zodat met een klik op om het even welke foto alle foto's aangepast worden.

Zoek zelf uit waarom het uitschuiven niet meer werkt.

JavaScript Object Notation (JSON)

JSON is een syntaxis voor het opslaan en uitwisselen van informatie. Vergelijkbaar met XML.

JSON is compacter dan XML, sneller en eenvoudiger om te ontleden.

Een voorbeeld:

```
{
  "werknemers": [
    { "voornaam":"Johan" , "naam":"Den Doven" },
    { "voornaam":"Ann" , "naam":"Segers" },
    { "voornaam":"Peter" , "naam":"Jansens" }
  ]
}
```

Het object werknemers bevat 3 werknemers (objecten).

Wat is JSON?

- JSON staat voor JavaScript Object Notation.
- JSON is een compact tekstformaat om gegevens uit te wisselen.
- JSON is taal taalonafhankelijk. Hoewel JSON de JavaScript syntaxis van objecten gebruikt is JSON taal- en platformonafhankelijk. Er bestaan JSON parsers en bibliotheken voor bijna alle programmeertalen.
- JSON omschrijft zichzelf en eenvoudig te begrijpen.

JSON en JavaScript objecten

Het JSON tekstformaat is identiek aan de JavaScript code voor het aanmaken van objecten.

Door deze overeenkomst kun je in een JavaScript programma de ingebouwde eval() functie gebruiken om JSON gegevens om te zetten naar JavaScript objecten.

Introductie

Een voorbeeld:

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
    8" />
5  <title>JSON</title>
6  </head>
7  <body>
8  <h2>JSON Object Creatie in JavaScript</h2>
9  <p>Naam: <span id="jnaam"></span><br />
10 Leeftijd: <span id="jleeftijd"></span><br />
11 Adres: <span id="jadres"></span><br />
12 Telefoon: <span id="jtelefoon"></span><br />
13 </p>
14 <script type="text/javascript">
15   var JSONObject = {
16     "naam":"Johan Den Doven",
17     "adres":"Arsenaalstraat 4, 8000 Brugge",
18     "leeftijd":33,
19     "telefoon":"050 33 76 69"};
20   document.getElementById("jnaam").innerHTML = JSONObject.naam;
21   document.getElementById("jleeftijd").innerHTML =
    JSONObject.leeftijd;
22   document.getElementById("jadres").innerHTML = JSONObject.adres;
23   document.getElementById("jtelefoon").innerHTML =
    JSONObject.telefoon;
24 </script>
25 </body>
26 </html>
```

Juist zoals XML

- JSON is pure tekst
- JSON is zelfbeschrijvend (eenvoudig te lezen)

- JSON is hiërarchisch opgebouwd (waarden binnen waarden)
- JSON kan door JavaScript verwerkt worden
- JSON gegevens kunnen met behulp van AJAX uitgewisseld worden

Niet zoals XML

- Gebruikt geen afsluitende tag
- compacter
- sneller te lezen en te schrijven
- kan verwerkt worden door de ingebouwde eval() functie
- gebruikt arrays
- geen gereserveerde worden

Waarom JSON gebruiken?

Bij AJAX toepassingen is JSON sneller en eenvoudiger dan XML:

Bij XML

- Het XML document ophalen
- Doorloop het document op zoek naar onderdelen (XML DOM)
- Sla de gevonden waarden op in variabelen.

Bij JSON

- De JSON tekenreeks ophalen
- eval() de JSON tekenreeks.

Syntaxis

De JSON syntaxis is een onderdeel van de JavaScript syntaxis

JSON syntaxis regels

De JSON syntaxis is afgeleid van de JavaScript object notatie.

- Gegevens bestaan uit koppels naam/waarde
- Gegevens worden door komma's gescheiden
- Accolades bevatten objecten
- Rechte haken bevatten arrays

JSON naam/waarde koppels

JSON gegevens worden opgeslagen in naam/waarde koppels.

Een naam/waarde koppel bestaat uit een veldnaam (tussen dubbele aanhalingstekens), gevolgd door een komma, gevolgd door een waarde:

```
"voornaam":"Johan"
```

Dit is eenvoudig te begrijpen, en komt overeen met de volgende Javascript opdracht:

```
voornaam = "Johan";
```

JSON waarden

Mogelijke JSON waarden zijn:

- getallen (gehele of kommagetallen)

- tekenreeksen (tussen dubbele aanhalingstekens)
- booleans (true of false)
- array's (tussen rechte haken)
- objecten (tussen accolades)
- null

JSON objecten

JSON objecten staan tussen accolades en kunnen meerdere naam/waarde koppels bevatten:

```
{ "voornaam":"Johan" , "naam":"Den Doven" }
```

Ook dit is eenvoudig te begrijpen en komt overeen met de volgende JavaScript opdrachten:

```
voornaam = "Johan";
naam = "Den Doven";
```

JSON arrays

JSON arrays staan tussen rechte haken. Elke array kan meerdere objecten bevatten:

```
{
  "werknemers": [
    { "voornaam":"Johan" , "naam":"Den Doven" },
    { "voornaam":"Ann" , "naam":"Segers" },
    { "voornaam":"Peter" , "naam":"Jansens" }
  ]
}
```

Het object werknemers is een array met drie objecten. Elk object bevat een record (rij) met de gegevens van één persoon (de velden voornaam en naam).

JSON en JavaScript

Daar JSON de JavaScript syntaxis gebruikt, heb je geen extra software nodig om in JavaScript met JSON te werken.

Met JavaScript maakt je als volgt een array met objecten met gegevens:

```
var werknemers = [
  { "voornaam":"Johan" , "naam":"Den Doven" },
  { "voornaam":"Ann" , "naam":"Segers" },
  { "voornaam":"Peter" , "naam":"Jansens" }
];
```

De eerste waarde in het JavaScript object kan als volgt gelezen worden:

```
werknemers[0].voornaam;
```

Dit levert als resultaat **Johan**.

Op de volgende manier kun je gegevens aanpassen:

```
werknemers[0].voornaam = "Jonathan";
```

JSON bestanden

JSON bestanden eindigen op **.json**.

Het MIME type voor een JSON tekst is **application/json**.

JSON toepassen

Eén van de meest gebruikte toepassingen van het JSON tekstformaat is het ophalen van gegevens van een webserver (via een bestand of via HttpRequest), waarna de JSON gegevens naar een JavaScript object worden omgezet en in de webpagina gebruikt worden.

Voor de eenvoud gebruiken we in de volgende voorbeelden een tekenreeks (string) als JSON bron.

```
25  <h2>Object Creatie met een JSON tekenreeks</h2>
26  <p>
27  Voornaam: <span id="jsvoornaam"></span><br />
28  Naam: <span id="jsnaam"></span>
29  </p>
30  <script>
31    var tekst = '{"werknemers":[' +
32      '{"voornaam":"Johan","naam":"Den Doven" },' +
33      '{"voornaam":"Ann","naam":"Segers" },' +
34      '{"voornaam":"Peter","naam":"Jansens" }]}';
35
36    var obj = eval ("(" + tekst + ")");
37
38    document.getElementById("jsvoornaam").innerHTML =
    obj.werknemers[1].voornaam;
39    document.getElementById("jsnaam").innerHTML =
    obj.werknemers[1].naam;
40  </script>
```

Daar JSON dezelfde syntaxis als JavaScript gebruikt, kan je de functie eval() gebruiken om JSON teksten naar JavaScript objecten om te zetten (regel 36).

De eval() functie gebruikt de JavaScript compiler om de JSON tekst om te zetten naar een JavaScript object. Om fouten te voorkomen moet de tekst tussen ronde haken geplaatst worden.

JSON parser

De eval() functie compileert en voert om het even welke JavaScript code uit. Dit is een potentieel gevaarlijke situatie.

Het gebruik van een JSON parser (verwerker) om JSON tekst naar een JavaScript object om te zetten is veiliger. Een JSON parser herkent alleen JSON teksten en compileert geen scripts.

De browsers die JSON ondersteunen, verwerken JSON sneller met de ingebouwde parser.

JSON parser ondersteuning is voorhanden in de recente browsers en in de recente ECMAScript (JavaScript) standaard.

Web browser ondersteuning	Web software ondersteuning
Firefox (Mozilla) 3.5	JQuery
Internet Explorer 8	Yahoo UI
Chrome	Prototype
Opera 10	Dojo
Safari 4	ECMAScript 1.5

Als voorbeeld vervang je in het vorige voorbeeld de regel

```
  var obj = eval ("(" + tekst + ")");
```

door de regel

```
var obj = JSON.parse(tekst);
```

Oefeningen

Slideshow

Maak een HTML pagina met één afbeelding met een bijhorende titel.

```
<img src="herfst.jpg" width="400" height="315" title="Herfst" />
```

Maak een JavaScript object waarin vier bestandsnamen (en hun eventuele pad) met bijhorende titel wordt opgeslagen.

Schrijf een functie om na het laden van de pagina de foto's om de vijf seconden te wisselen. Gebruik de volgende JavaScript code om een afbeelding met het id slideshow de foto met de bestandsnaam opgeslagen in de variabele bestandsnaam te laten weergeven:

```
document.getElementById("slideshow").src = bestandsnaam;
```

Op dezelfde manier kun je de bijhorende titel aanpassen.

JSON

Maak van het JavaScript object een JSON tekenreeks.

Beschrijving toevoegen

Aangezien de titel van een afbeelding pas verschijnt bij het aanwijzen van een foto ga je een beschrijving onder de foto plaatsen.

Voeg aan de JSON tekenreeks voor elke foto een beschrijving toe.

Plaats juist onder de afbeelding een aanpasbare span tag.

Zorg dat de beschrijving van de foto's in deze span tag wordt weergegeven.

Eenvoudige database met gegevens van vrienden of leden van een vereniging

Maak een HTML pagina waarmee je de gegevens van een persoon in een JavaScript object opslaat en wordt weergegeven zoals op de afbeelding.

Voor de lay-out gebruik je best een tabel zonder randen.

Om de structuur op het scherm te laten overeenkomen met de object structuur, zorg je dat het object vijf blokken bevat (voornaam, naam, leeftijd, adres en telefoon). De eerste drie blokken bestaan telkens uit één koppel. Het blok met het adres bevat de koppels straat, woonplaats en postcode), het blok telefoon bevat de koppels vast en mobiel.

Persoonlijke gegevens

Voornaam:	Dany	
Naam:	Pinoy	
Leeftijd:	25	
Adres:	Straat nr:	arsenaalstraat 4
	Woonplaats:	Brugge
	Postcode:	8000
Telefoon:	Vast:	050 33 76 69
	Mobiel:	

JSON gegevens ophalen met AJAX

Maak van het JavaScript object een JSON gegevensbestand met de naam dany.pinoy.json.

Zorg ervoor dat de webpagina AJAX kan gebruiken.

Gebruik AJAX om het JSON bestand op te halen en daarna naar een JavaScript object om te zetten.

Database uitbreiden

Maak een tweede JSON bestand met eigen gegevens en de bestandnaam voornaam.naam.json.

Plaats boven de tabel een tekstveld waarin de gebruiker een login kan intypen.

Plaats naast dit tekstveld een knop met de tekst **Gegevens opvragen**.

Zorg dat bij het klikken op deze knop de gegevens van de login (bijvoorbeeld: dany.pinoy) in de tabel worden weergegeven.

Beveiligen

Bestandsnamen verdoezelen

Om het verband tussen de login en de bestandsnaam te verdoezelen gebruiken we niet te raden willekeurige bestandsnamen:

dany.pinoy.json wordt 1093A49D.json.

Kies zelf een willekeurige bestandsnaam voor de tweede persoon.

Maak een derde JSON bestand waarin je per login de bijhorende bestandsnaam kunt opslaan.

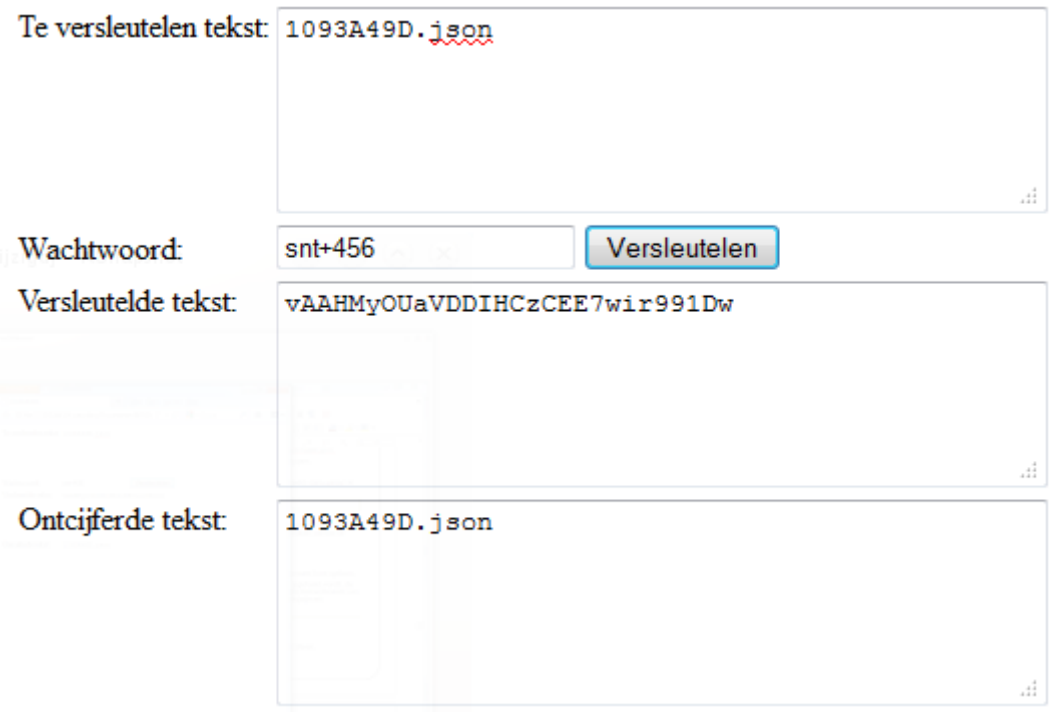
Zorg dat bij het klikken op de knop dat het JSON bestand met de logins opgehaald wordt, de juiste login wordt opgezocht en indien deze wordt gevonden, de bijhorende bestandsnaam van het JSON bestand met de persoonlijke gegevens wordt opgehaald en weergegeven.

Wachtwoord toevoegen

Maak een nieuwe HTML pagina met een formulier zoals op de afbeelding.

Zorg dat de pagina het externe script aes.js laadt.

Dit externe script bevat functies om tekst te versleutelen en terug te ontcijferen.



Met de functie Aes.Ctr.encrypt("tekst", "wachtwoord", 256) kun je een tekst versleutelen. Deze functie geeft een versleutelde tekenreeks terug waarmee je indien je het wachtwoord kent de originele tekst kunt achterhalen.

Met de functie Aes.Ctr.encrypt("versleutelde tekst", "wachtwoord", 256) kun je een versleutelde tekst terug ontcijferen.

Zorg dat bij het klikken op de knop **Versleutelen** deze functies gebruikt worden om de inhoud van de velden **Versleutelde inhoud** en **Ontcijferde tekst** aan te maken. De inhoud van het veld **Ontcijferde tekst** wordt berekend met behulp van de Versleutelde tekst en het opgegeven wachtwoord (en is dus bedoeld als test).

Gebruik deze pagina om de bestandsnamen in het JSON bestand met de login's met een wachtwoord te versleutelen.

Plaats op de webpagina met de persoonlijke gegevens onder het login tekstveld een wachtwoord veld.

Pas de JavaScript code aan zodat de versleutelde bestandsnamen gebruikt worden.

Zo kan iemand die toegang heeft tot het JSON bestand met de login's niet achterhalen in welk JSON bestand de gegevens opgeslagen zijn.

Gegevens beveiligen

Pas dezelfde versleuteling toe op de persoonlijke gegevens in de JSON bestanden.

JavaScript en XML

XML

XML is belangrijk en eenvoudig aan te leren.

Een voorbeeld:

```
<?xml version="1.0" encoding="utf-8"?>
<nota>
  <voor>Tania</voor>
  <van>Jan</van>
  <titel>Geheugensteun</titel>
  <bericht>Stuur je me een verslag van de vergadering.</bericht>
</nota>
```

Introductie

Wat is XML?

XML staat voor e**X**tensible **M**arkup **L**anguage.

XML is een markup taal zoals HTML.

XML is ontworpen om gegevens uit te wisselen en op te slaan.

XML tags zijn niet voorgedefinieerd. Je moet ze zelf definiëren.

XML is zelf beschrijvend.

XML is software- en hardware onafhankelijk.

XML wordt sedert 10 februari 1998 door W3C aanbevolen.

De verschillen tussen XML en HTML

XML en HTML hebben elk hun eigen functie:

- XML is ontworpen om gegevens uit te wisselen en op te slaan. De gegevens staan centraal.
- HTML is ontworpen om gegevens weer te geven. De lay-out staat centraal.

HTML toont informatie, XML transporteert informatie.

XML doet niets

XML is ontworpen om informatie te structureren, op te slaan en te transporteren.

De nota in het voorbeeld is zelf beschrijvend. Het heeft een afzender (van), een bestemming (voor), een titel en een tekstbericht.

M.a.w. dit XML document doet helemaal niets. Het wordt enkel gebruikt om informatie in tags op te slaan. Er zijn dus programma's nodig om XML te verzenden, te ontvangen en weer te geven.

In XML gebruik je eigen tags

De tags in het voorbeeld (zoals <voor> en <van>) maken geen deel uit van de XML standaard. Deze tags zijn verzonnen door de auteur van het XML document.

De XML standaard bevat dus geen voorgedefinieerde tags.

De tags die in HTML gebruikt worden, zijn voorgedefinieerd en maken deel uit van de HTML standaard.

De auteur van een XML document definieert zijn eigen tags en documentstructuur.

XML is overal

XML is vandaag even belangrijk voor het web als HTML bij het ontstaan van het web.

XML is de meest gebruikte manier om informatie tussen toepassingen uit te wisselen.

Waar en hoe wordt XML gebruikt?

XML wordt meestal gebruikt om de opslag en het delen van gegevens op het web te vereenvoudigen.

XML scheidt de gegevens van HTML

Om dynamisch gegevens in een HTML document weer te geven, moet je niet telkens het HTML document aanpassen.

De gegevens worden opgeslagen in afzonderlijke XML bestanden. De HTML/CSS tandem zorgt dus enkel voor het weergeven en de lay-out, waardoor de onderliggende gegevens geen invloed meer hebben op de HTML en CSS code.

Met behulp van JavaScript wordt een extern XML bestand geladen en wordt de inhoud van de webpagina met de geladen gegevens uitgebreid en/of aangepast.

XML vereenvoudigd het delen van gegevens

Computersystemen en databases bevatten gegevens in niet compatibele formaten.

XML gebruikt het platte tekst formaat. Deze methode is een software- en hardware onafhankelijke manier om gegevens op te slaan.

Dit maakt het veel eenvoudiger om gegevens die door verschillende toepassingen gebruikt worden op te slaan.

XML vereenvoudigd het gegevenstransport

Eén van de meest tijdrovende uitdagingen voor ontwikkelaars is het uitwisselen van gegevens tussen niet compatibele systemen.

Gegevens uitwisselen in het XML formaat is een welkome vereenvoudiging.

XML vereenvoudigd platform migraties

Upgraden naar nieuwe systemen (hardware of software platforms) is steeds tijdrovend. Grote hoeveelheden gegevens moeten daarbij omgezet worden waarbij niet compatibele gegevens dikwijls verloren gaan.

XML gegevens zijn in tekstformaat opgeslagen. Dit maakt het veel eenvoudiger om zonder verlies aan gegevens over te stappen naar een ander besturingssysteem, andere toepassingen of een andere browser.

XML zorgt voor een betere beschikbaarheid

Verschillende toepassingen kunnen de gegevens raadplegen, niet enkel HTML pagina's maar ook diverse andere toepassingen.

XML maakt uw gegevens beschikbaar voor verschillende apparaten (van draagbare computers, spraakcomputers, nieuwslezers – feeds, enz.), ook voor apparaten en software voor blinden en andersvaliden.

XML vormt de basis van nieuwe Internet talen

De volgende Internet talen gebruiken XML als basis:

- XHTML
- WSDL om beschikbare webdiensten te beschrijven
- WAP en WML als markup talen voor draagbare apparaten
- RSS voor nieuwsberichten (news feeds)
- RDF en OWL voor het beschrijven van bronnen en ontologieën
- SMIL voor het beschrijven van multimedia op het web

De toekomst

Indien XML de standaard wordt om gegevens tussen toepassingen uit te wisselen, komen er in de nabije toekomst tekstverwerkers, rekenbladen en databanken die zonder conversie gegevens onderling kunnen uitwisselen.

XML boom

XML documenten gebruiken een boomstructuur die bij de “stam” start en zich tot aan de “bladeren” vertakt.

Het voorbeeld:

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <nota>
3     <voor>Tania</voor>
4     <van>Jan</van>
5     <titel>Geheugensteun</titel>
6     <bericht>Stuur je me een verslag van de vergadering.</bericht>
7   </nota>
```

De eerste regel is de XML declaratie. Het definieert de gebruikte XML versie (1.0) en de karakterset (utf-8).

De volgende regel beschrijft de stam (**root** element) van het document (dit document is een nota).

De volgende vier regels beschrijven vier bladeren (**child** elementen) van de root (voor, van, titel en bericht).

De laatste regel sluit het root element af.

Je kunt er dus vanuit gaan dat dit XML document een nota van Jan voor Tania bevat.

Dit is wat we bedoelen met zelf beschijvend.

XML documenten bestaan uit een boomstructuur

XML documenten moeten een **root** element bevatten. Dit element is de ouder (**parent**) van alle andere elementen.

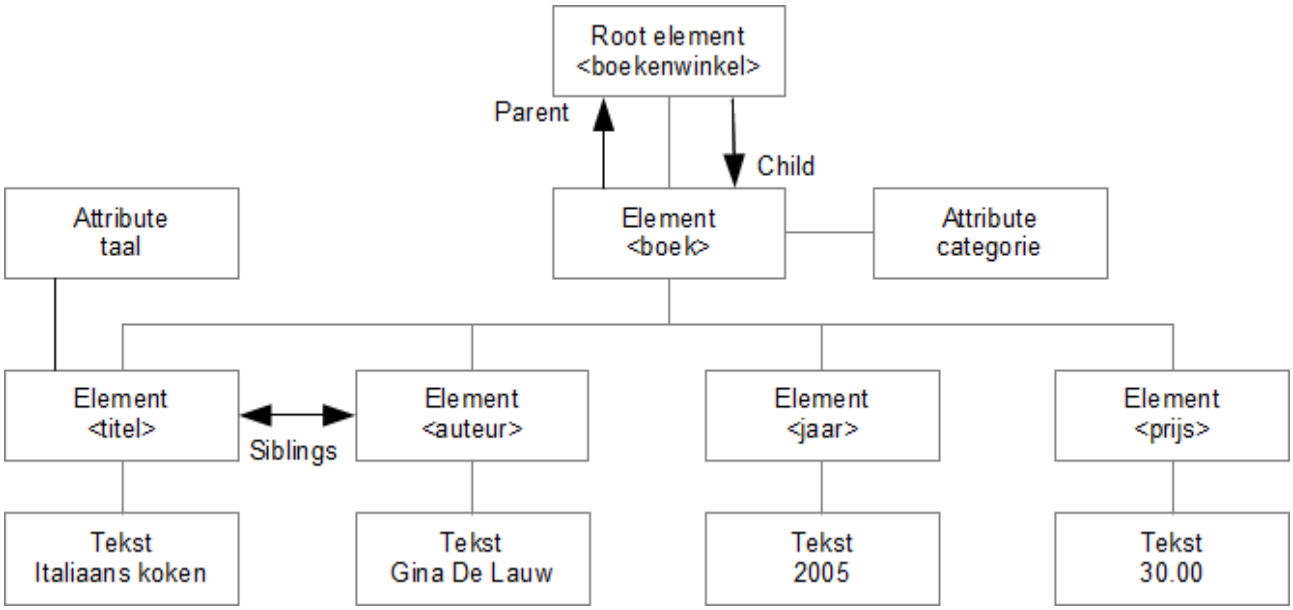
De elementen in een XML document vormen een documentboom. De boom start bij de root en vertakt zich verder tot het laatste niveau van de boom.

Alle elementen kunnen subelementen bevatten (child elementen):

```
<root>
   <child>
     <subchild>.....</subchild>
   </child>
 </root>
```

De termen parent, child en sibling worden gebruikt om de relaties tussen elementen te beschrijven. Parent elementen hebben children. Children in hetzelfde niveau noemen we siblings (broers en zussen).

Alle elementen kunnen zoals bij HTML tekst en kenmerken (attributes) bevatten.



De afbeelding hierboven is een grafische voorstelling van onderstaand XML document:

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <boekenwinkel>
3     <boek categorie="KOKEN">
4       <titel taal="nl">Italiaans koken</titel>
5       <auteur>Gina De Lauw</auteur>
6       <jaar>2005</jaar>
7       <prijs>30.00</prijs>
8     </boek>
9     <boek categorie="KINDEREN">
10      <titel taal="nl">Harry Potter</titel>
11      <auteur>J K. Rowling</auteur>
12      <jaar>2005</jaar>
13      <prijs>29.99</prijs>
14    </boek>
15    <boek categorie="WEB">
16      <titel taal="en">Learning XML</titel>
17      <auteur>Erik T. Ray</auteur>
18      <jaar>2003</jaar>
19      <prijs>39.95</prijs>
20    </boek>
21  </boekenwinkel>
```

Het root element in het voorbeeld is <boekenwinkel>. Alle <boek> elementen staan binnen het <boekenwinkel> root element.

XML Syntaxis

De syntaxis regels van XML zijn eenvoudig en logisch.

Alle XML elementen moeten worden afgesloten

In HTML hebben sommige elementen geen afsluitende tag.

In XML moet elke tag afgesloten worden.

XML tags zijn hoofdlettergevoelig

XML tags zijn hoofdlettergevoelig. De tag <boek> is niet hetzelfde als de tag <Boek>.

Openende (start) en afsluitende (eind) tags moeten dezelfde letters gebruiken:

```
<prijs>39.95</prijs>
```

XML tags moeten correct genest worden

Met correct genest bedoelen we dat een tag die binnen een andere tag wordt geopend, ook binnen dezelfde tag wordt gesloten.

```
<boek categorie="WEB">
  <titel taal="en">Learning XML</titel>
  <auteur>Erik T. Ray</auteur>
  <jaar>2003</jaar>
  <prijs>39.95</prijs>
</boek>
```

XML documenten bevatten een root element

XML documenten bevatten een tag waarin alle andere elementen staan.

XML attributen gebruiken steeds aanhalingstekens

```
<titel taal="en">Learning XML</titel>
```

Entity References

Sommige tekens hebben in XML een speciale betekenis.

Bij het gebruik van het teken "<" binnen een XML element krijg je een fout. De XML verwerker ziet dit namelijk als het begin van een nieuw element.

Dit veroorzaakt een fout:

```
<bericht>Lonen < 1000 krijgen</bericht>
```

Om een fout te vermijden, vervang je het "<" teken door een **entity reference**:

```
<bericht>Lonen &lt; 1000 krijgen</bericht>
```

XML kent vijf voorgedefinieerde entity references:

&lt;	<	Kleiner dan
&gt;	>	Groter dan
&amp;	&	Ampersand
&apos;	'	apostrof
&quot;	"	Aanhalingsteken

XML commentaar

De syntaxis voor XML commentaar is dezelfde als in HTML.

```
<!-- Dit is commentaar -->
```

XML respecteert witruimte

HTML herleidt meerdere opeenvolgende witruimte karakters tot één spatie.

Bij XML zijn alle opeenvolgende witruimte karakters van betekenis.

XML slaat een nieuwe regel op als LF

Windows toepassingen slaan een nieuwe regel meestal op met twee tekens: carriage return (CR) en line feed (LF). OS X en Unix toepassingen slaan een nieuwe regel meestal op als een LF teken.

XML slaat een nieuwe regel op als een LF teken.

XML elementen

Een XML element is alles vanaf en inclusief de start tag tot en met de eind tag van een element.

Een element bevat:

- Andere elementen
- Tekst
- Kenmerken (attributen)
- Of een mengsel van de drie bovenstaande onderdelen.

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <boekenwinkel>
3     <boek categorie="KINDEREN">
4       <titel taal="nl">Harry Potter</titel>
5       <auteur>J K. Rowling</auteur>
6       <jaar>2005</jaar>
7       <prijs>29.99</prijs>
8     </boek>
9     <boek categorie="WEB">
10      <titel taal="en">Learning XML</titel>
11      <auteur>Erik T. Ray</auteur>
12      <jaar>2003</jaar>
13      <prijs>39.95</prijs>
14    </boek>
15  </boekenwinkel>
```

In bovenstaand voorbeeld bevatten de elementen <boekenwinkel> en <boek> andere elementen. <boek> bevat daarnaast een attribuut (categorie = "KINDEREN"). <titel>, <auteur>, <jaar> en <prijs> bevatten tekst.

XML namen

Namen van XML elementen voldoen aan de volgende regels:

- Namen mogen letters, cijfers en andere tekens bevatten
- Namen mogen niet beginnen met een cijfer of een leesteken
- Namen mogen niet starten met de letters xml (of XML, Xml, enz.)
- Namen mogen spaties bevatten.

Goede naamgeving

Zorg voor beschrijvende namen. Namen met een underscore worden aangeraden: <naam_auteur>.

Zorg voor korte namen: <boek_titel>.

Gebruik geen – tekens, komma's, dubbele punten.

XML documenten hebben vaak een overeenkomstige database. Gebruik dezelfde namen in het XML document en de database.

Speciale tekens zoals éóà zijn toegelaten in XML, test de XML verwerker grondig met zulke tekens, niet alle XML verwerkers ondersteunen deze speciale tekens.

XML elementen zijn uitbreidbaar

XML elementen kan je uitbreiden met extra informatie.

```
<boek categorie="WEB">
  <titel taal="en">Learning XML</titel>
  <auteur>Erik T. Ray</auteur>
  <prijs>39.95</prijs>
</boek>
```

Laten we veronderstellen dat je een toepassing hebt geschreven waarbij je de elementen <titel>, <auteur> en <prijs> gebruikt.

De auteur van het XML voegt achteraf een element met extra informatie toe:

```
<boek categorie="WEB">
  <titel taal="en">Learning XML</titel>
  <auteur>Erik T. Ray</auteur>
  <jaar>2003</jaar>
  <prijs>39.95</prijs>
</boek>
```

Uw toepassing zal nog steeds werken. M.a.w. XML documenten kunnen uitgebreid worden zonder de werking van een toepassing te storen.

XML Attributen

De attributen verschaffen aanvullende informatie over een element.

Attributen bevatten informatie dat geen deel uitmaakt van de gegevens. Een voorbeeld: het bestandsformaat is geen deel van het gegeven, maar is belangrijk voor de software die het element moet behandelen:

```
<bestand type="gif">computer.gif</bestand>
```

XML attributen staan steeds tussen aanhalingstekens

Attributen staan steeds tussen enkele of dubbele aanhalingstekens:

```
<persoon geslacht="vrouw">
<persoon geslacht='vrouw'>
```

Indien een attribuutwaarde zelf aanhalingstekens bevat gebruik je enkele aanhalingstekens of entity references:

```
<gangster naam='Patrick "Le grand blond" Haemers'>
<gangster naam="Patrick &quot; Le grand blond&quot; Haemers'>
```

XML elementen versus attributen

Bekijk de volgende voorbeelden:

```
<persoon geslacht="vrouw">
   <voornaam>Ann</voornaam>
   <naam>Saelens</naam>
 </persoon>
```

```
<persoon>
   <geslacht>vrouw</geslacht>
   <voornaam>Ann</voornaam>
   <naam>Saelens</naam>
 </persoon>
```

In het eerste voorbeeld is het geslacht een attribuut, in het tweede voorbeeld een element. Beide voorbeelden leveren dezelfde informatie.

Er zijn geen regels over wanneer je een attribuut of een element gebruikt. Mij lijkt het eenvoudiger om attributen te vermijden en steeds elementen te gebruiken.

Vermijdt attributen

Attributen hebben de volgende nadelen:

- attributen kunnen maar één waarde hebben
- attributen kunnen geen boomstructuur bevatten
- attributen kun je niet eenvoudig uitbreiden

Attributen zijn daarenboven moeilijker te lezen en te onderhouden. Gebruik elementen voor gegevens, attributen voor informatie die niet relevant is voor de gegevens.

Mijn favoriete manier van werken:

```
<nota>
  <datum>
    <dag>10</dag>
    <maand>01</maand>
    <jaar>2012</jaar>
  </datum>
  <voor>Tania</voor>
  <van>Jan</van>
  <titel>Geheugensteun</titel>
  <bericht>Stuur je me een verslag van de vergadering.</bericht>
</nota>
```

XML attributen voor metadata

Soms worden ID referenties toegekend aan elementen. Deze ID's worden gebruikt om XML elementen te identificeren (juist zoals HTML ID's):

```
<berichten>
  <nota id="501">
    <voor>Tania</voor>
    <van>Jan</van>
    <titel>Geheugensteun</titel>
    <bericht>Stuur je me een verslag van de vergadering.</bericht>
  </nota>
  <nota id="502">
    <voor>Jan</voor>
```

```
    <van>Tania</van>
    <titel>Re: Geheugensteun</titel>
    <bericht>Doe ik.</bericht>
  </nota>
</berichten>
```

Het id attribuut wordt gebruikt om de verschillende nota's te identificeren. Het is geen deel van de nota zelf.

Metadata (data over de data) sla je op als attributen, de data zelf als elementen.

XML Validatie

Well Formed XML documenten

Well Formed XML documenten hebben een correcte XML systaxis:

```
<nota>
  <voor>Tania</voor>
  <van>Jan</van>
  <titel>Geheugensteun</titel>
  <bericht>Stuur je me een verslag van de vergadering.</bericht>
</nota>
```

Valid XML documenten

Een Valid XML document is een Well Formed XML document die de regels van een Document Type Definition (DTD) volgt:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <!DOCTYPE nota SYSTEM "nota.dtd">
3  <nota>
4    <voor>Tania</voor>
5    <van>Jan</van>
6    <titel>Geheugensteun</titel>
7    <bericht>Stuur je me een verslag van de vergadering.</bericht>
8  </nota>
```

De DOCTYPE declaratie is een referentie naar een extern DTD bestand.

XML DTD

Het doel van een DTD is de structuur van een XML document vast te leggen. Het definieert een structuur aan de hand van toegelaten elementen:

```
1  <!DOCTYPE nota
2   [
3   <!ELEMENT nota (voor,van,titel,bericht)>
4   <!ELEMENT voor (#PCDATA)>
5   <!ELEMENT van (#PCDATA)>
6   <!ELEMENT titel (#PCDATA)>
7   <!ELEMENT bericht (#PCDATA)>
8   ]>
```

XML Schema

W3C ondersteunt een op XML gebaseerd alternatief voor DTD: XML Schema:

```
1  <xs:element name="nota">
2   <xs:complexType>
3     <xs:sequence>
```

```
4       <xs:element name="voor" type="xs:string"/>
5       <xs:element name="van" type="xs:string"/>
6       <xs:element name="titel" type="xs:string"/>
7       <xs:element name="bericht" type="xs:string"/>
8     </xs:sequence>
9   </xs:complexType>
10 </xs:element>
```

XML XSLT

Met XSLT kan je een XML document omzetten naar HTML.

XSLT is de stijlpagina taal voor XML

XSLT (eXtensible Stylesheet Language Transformations) is veel geavanceerder dan CSS.

XSLT kan gebruikt worden om XML naar HTML om te zetten, voor het in de browser wordt weergeven.

XML JavaScript

Het XMLHttpRequest object

Het XMLHttpRequest object wordt gebruikt om op de achtergrond gegevens met een server uit te wisselen.

Het XMLHttpRequest object is een droom voor elke ontwikkelaar, want je kunt:

- een webpagina aanpassen zonder te herladen (vernieuwen)
- data van een server opvragen na het laden van de pagina
- data van een server ontvangen na het laden van de pagina
- data in de achtergrond naar een server sturen

Alle moderne browsers (IE7+, Firefox, Chrome, Safari en Opera) hebben een ingebouwd XMLHttpRequest object.

Syntaxis om een XMLHttpRequest object aan te maken:

```
xmlhttp = new XMLHttpRequest();
```

Oudere versies van Internet Explorer (IE5 en IE6) gebruiken ActiveX objecten:

```
xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
```

XML Parser

Alle moderne browsers hebben een ingebouwde XML parser.

Een XML parser zet een XML document om naar een XML DOM object – die je met JavaScript kunt bewerken.

Een XML document omzetten

De volgende code zet een XML document om in een XML DOM object:

```
xmlhttp=new XMLHttpRequest();
xmlhttp.open("GET","boeken.xml",false);
xmlhttp.send();
xmlDoc=xmlhttp.responseXML;
```

Een XML tekenreeks omzetten

De volgende code zet een XML tekenreeks om in een XML DOM object:

```
tekst="<boekenwinkel><boek>";
tekst = tekst + "<titel>Italiaans koken</titel>";
tekst = tekst + "<auteur>Gina De Lauw</auteur>";
tekst = tekst + "<jaar>2005</jaar>";
tekst = tekst + "</boek></boekenwinkel>";
if (window.DOMParser){
  parser = new DOMParser();
  xmlDoc = parser.parseFromString(tekst,"text/xml");
} else { // Internet Explorer
  xmlDoc = new ActiveXObject("Microsoft.XMLDOM");
  xmlDoc.async = false;
  xmlDoc.loadXML(tekst);
}
```

Internet Explorer gebruikt de loadXML() methode om een XML tekenreeks om te zetten, terwijl andere browsers het DOMParser object gebruiken.

Toegang tot andere domeinen

Moderne browsers krijgen geen toegang tot andere domeinen, dit zou een beveiligingsprobleem opleveren.

Dit betekent dat de webpagina en het XML bestand dat de pagina wil laden op dezelfde webserver moeten staan.

XML DOM

Het XML DOM (Document Object Model) definieert een standaard om XML documenten te lezen en te bewerken.

Het XML DOM bekijkt een XML document als een boomstructuur.

Alle elementen kunnen via de DOM boomstructuur bereikt worden. Hun inhoud (tekst en attributen) kunnen bewerkt of vernietigd worden, nieuwe elementen kunnen aangemaakt worden.

Een XML bestand laden

Een voorbeeld met het volgende nota.xml bestand:

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <nota>
3    <voor>Tania</voor>
4    <van>Jan</van>
5    <titel>Geheugensteun</titel>
6    <bericht>Stuur je me een verslag van de vergadering.</bericht>
7  </nota>
```

De volgende JavaScript code zet het XML document nota.xml om naar een XML DOM object en geeft de informatie weer.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
    8" />
5  <title>Interne nota</title>
6  </head>
```

```
7   <body>
8   <h1>Interne Nota</h1>
9   <div>
10    <b>Voor:</b> <span id="voor"></span><br />
11    <b>Van:</b> <span id="van"></span><br />
12    <b>Bericht:</b> <span id="bericht"></span>
13  </div>
14  <script type="text/javascript">
15  xmlhttp=new XMLHttpRequest();
16  xmlhttp.open("GET","nota.xml",false);
17  xmlhttp.send();
18  xmlDoc=xmlhttp.responseXML;
19  document.getElementById("voor").innerHTML =
     xmlDoc.getElementsByTagName("voor")[0].childNodes[0].nodeValue;
20  document.getElementById("van").innerHTML =
     xmlDoc.getElementsByTagName("van")[0].childNodes[0].nodeValue;
21  document.getElementById("bericht").innerHTML =
     xmlDoc.getElementsByTagName("bericht")[0].childNodes[0].nodeValue;
22  </script>
23  </body>
24  </html>
```

Opmerkingen:

- Dit voorbeeld kan lokaal enkel getest worden in Firefox, testen met andere browsers kan alleen als de bestanden op een webserver gepubliceerd zijn.
- Regel 19 haalt de tekst "Tania" uit het element "voor". Zelfs als het XML bestand maar één "voor" element bevat, moet je toch de array index [0] gebruiken. De methode getElementsByTagName() geeft namelijk altijd een array terug.

XML gegevens in een HTML tabel weergeven

Een voorbeeld met het volgende cd_catalogus.xml bestand:

```
1   <?xml version="1.0" encoding="utf-8"?>
2   <catalogus>
3     <cd>
4       <titel>Empire Burlesque</titel>
5       <artiest>Bob Dylan</artiest>
6       <land>USA</land>
7       <label>Columbia</label>
8       <prijs>10.90</prijs>
9       <jaar>1985</jaar>
10    </cd>
11    <cd>
12      <titel>Hide your heart</titel>
13      <artiest>Bonnie Tyler</artiest>
14      <land>UK</land>
15      <label>CBS Records</label>
16      <prijs>9.90</prijs>
17      <jaar>1988</jaar>
18    </cd>
19    <cd>
20      <titel>Greatest Hits</titel>
21      <artiest>Dolly Parton</artiest>
22      <land>USA</land>
23      <label>RCA</label>
24      <prijs>9.90</prijs>
25      <jaar>1982</jaar>
```

```
  </cd>
  <cd>
    <titel>Still got the blues</titel>
    <artiest>Gary Moore</artiest>
    <land>UK</land>
    <label>Virgin records</label>
    <prijs>10.20</prijs>
    <jaar>1990</jaar>
  </cd>
  <cd>
    <titel>Eros</titel>
    <artiest>Eros Ramazzotti</artiest>
    <land>EU</land>
    <label>BMG</label>
    <prijs>9.90</prijs>
    <jaar>1997</jaar>
  </cd>
  <cd>
    <titel>One night only</titel>
    <artiest>Bee Gees</artiest>
    <land>UK</land>
    <label>Polydor</label>
    <prijs>10.90</prijs>
    <jaar>1998</jaar>
  </cd>
  <cd>
    <titel>Sylvias Mother</titel>
    <artiest>Dr.Hook</artiest>
    <land>UK</land>
    <label>CBS</label>
    <prijs>8.10</prijs>
    <jaar>1973</jaar>
  </cd>
  <cd>
    <titel>Maggie May</titel>
    <artiest>Rod Stewart</artiest>
    <land>UK</land>
    <label>Pickwick</label>
    <prijs>8.50</prijs>
    <jaar>1990</jaar>
  </cd>
  <cd>
    <titel>Romanza</titel>
    <artiest>Andrea Bocelli</artiest>
    <land>EU</land>
    <label>Polydor</label>
    <prijs>10.80</prijs>
    <jaar>1996</jaar>
  </cd>
  <cd>
    <titel>When a man loves a woman</titel>
    <artiest>Percy Sledge</artiest>
    <land>USA</land>
    <label>Atlantic</label>
    <prijs>8.70</prijs>
    <jaar>1987</jaar>
  </cd>
  <cd>
```

```
    <titel>Black angel</titel>
    <artiest>Savage Rose</artiest>
    <land>EU</land>
    <label>Mega</label>
    <prijs>10.90</prijs>
    <jaar>1995</jaar>
  </cd>
  <cd>
    <titel>1999 Grammy Nominees</titel>
    <artiest>Many</artiest>
    <land>USA</land>
    <label>Grammy</label>
    <prijs>10.20</prijs>
    <jaar>1999</jaar>
  </cd>
  <cd>
     <titel>For the good times</titel>
     <artiest>Kenny Rogers</artiest>
     <land>UK</land>
     <label>Mucik Master</label>
     <prijs>8.70</prijs>
     <jaar>1995</jaar>
   </cd>
   <cd>
     <titel>Big Willie style</titel>
     <artiest>Will Smith</artiest>
     <land>USA</land>
     <label>Columbia</label>
     <prijs>9.90</prijs>
     <jaar>1997</jaar>
   </cd>
   <cd>
     <titel>Tupelo Honey</titel>
     <artiest>Van Morrison</artiest>
     <land>UK</land>
     <label>Polydor</label>
     <prijs>8.20</prijs>
     <jaar>1971</jaar>
   </cd>
   <cd>
     <titel>Soulsville</titel>
     <artiest>Jorn Hoel</artiest>
     <land>Norway</land>
     <label>WEA</label>
     <prijs>7.90</prijs>
     <jaar>1996</jaar>
   </cd>
   <cd>
     <titel>The very best of</titel>
     <artiest>Cat Stevens</artiest>
     <land>UK</land>
     <label>Island</label>
     <prijs>8.90</prijs>
     <jaar>1990</jaar>
   </cd>
   <cd>
     <titel>Stop</titel>
     <artiest>Sam Brown</artiest>
```

```
142     <land>UK</land>
143     <label>A and M</label>
144     <prijs>8.90</prijs>
145     <jaar>1988</jaar>
146   </cd>
147   <cd>
148     <titel>Bridge of Spies</titel>
149     <artiest>T'Pau</artiest>
150     <land>UK</land>
151     <label>Siren</label>
152     <prijs>7.90</prijs>
153     <jaar>1987</jaar>
154   </cd>
155   <cd>
156     <titel>Private Dancer</titel>
157     <artiest>Tina Turner</artiest>
158     <land>UK</land>
159     <label>Capitol</label>
160     <prijs>8.90</prijs>
161     <jaar>1983</jaar>
162   </cd>
163   <cd>
164     <titel>Midt om natten</titel>
165     <artiest>Kim Larsen</artiest>
166     <land>EU</land>
167     <label>Medley</label>
168     <prijs>7.80</prijs>
169     <jaar>1983</jaar>
170   </cd>
171   <cd>
172     <titel>Pavarotti Gala Concert</titel>
173     <artiest>Luciano Pavarotti</artiest>
174     <land>UK</land>
175     <label>DECCA</label>
176     <prijs>9.90</prijs>
177     <jaar>1991</jaar>
178   </cd>
179   <cd>
180     <titel>The dock of the bay</titel>
181     <artiest>Otis Redding</artiest>
182     <land>USA</land>
183     <label>Atlantic</label>
184     <prijs>7.90</prijs>
185     <jaar>1987</jaar>
186   </cd>
187   <cd>
188     <titel>Picture book</titel>
189     <artiest>Simply Red</artiest>
190     <land>EU</land>
191     <label>Elektra</label>
192     <prijs>7.20</prijs>
193     <jaar>1985</jaar>
194   </cd>
195   <cd>
196     <titel>Red</titel>
197     <artiest>The Communards</artiest>
198     <land>UK</land>
199     <label>London</label>
```

```
200     <prijs>7.80</prijs>
201     <jaar>1987</jaar>
202   </cd>
203   <cd>
204     <titel>Unchain my heart</titel>
205     <artiest>Joe Cocker</artiest>
206     <land>USA</land>
207     <label>EMI</label>
208     <prijs>8.20</prijs>
209     <jaar>1987</jaar>
210   </cd>
211 </catalogus>
```

Met de volgende JavaScript code doorlopen we het XML bestand cd_catalogus.xml en geven we elke element weer in een tabelcel:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>CD catalogus</title>
6   </head>
7   <body>
8   <script type="text/javascript">
9   xmlhttp = new XMLHttpRequest();
10  xmlhttp.open("GET","cd_catalogus.xml",false);
11  xmlhttp.send();
12  xmlDoc=xmlhttp.responseXML;
13  document.write("<table border='1'>");
14  var x = xmlDoc.getElementsByTagName("cd");
15  for (i = 0; i < x.length; i++){
16    document.write("<tr><td>");
17    document.write(x[i].getElementsByTagName("artiest")
     [0].childNodes[0].nodeValue);
18      document.write("</td><td>");
19      document.write(x[i].getElementsByTagName("titel")
     [0].childNodes[0].nodeValue);
20      document.write("</td></tr>");
21  }
22  document.write("</table>");
23  </script>
24  </body>
25  </html>
```

XML toepassingen

We gebruiken XML, HTML, XML DOM en JavaScript om een eenvoudige internet toepassing te schrijven.

Als XML bron gebruiken we het reeds voordien gebruikte cd_catalogus.xml bestand.

De eerste CD in een HTML div tag weergeven

De volgende code haalt de XML data van de eerste CD op en geeft deze weer in de HTML div tag met het id="CDweergeven". De functie Cdweergeven() wordt uitgevoerd na het laden van de pagina.

```
x = xmlDoc.getElementsByTagName("cd");
```

```
i = 0;
function CDweergeven(){
  artiest = (x[i].getElementsByTagName("artiest")
[0].childNodes[0].nodeValue);
  titel = (x[i].getElementsByTagName("titel")
[0].childNodes[0].nodeValue);
  jaar = (x[i].getElementsByTagName("jaar")
[0].childNodes[0].nodeValue);
  tekst = "Artiest: " + artiest + "<br>Titel: " + titel + "<br>Jaar: "+
jaar;
  document.getElementById("CDweergeven").innerHTML = tekst;
}
```

Door de CD's bladeren

Om door de CD's te bladeren, voegen we twee functies toe, volgende() en vorige():

```
function volgende(){
  if (i < x.length-1){
    i++;
    CDweergeven();
  }
}
function vorige(){
  if (i > 0){
    i--;
    CDweergeven();
  }
}
```

Oefeningen

Voorbeeld afwerken

Gebruik de twee voorgaande scripts om een pagina te maken waarbij bij het laden van de pagina alle gegevens van de eerste CD weergegeven worden.

Daaronder plaats je twee knoppen om de vorige en volgende CD weer te geven.

Daaronder plaats je een tabel van de artiest en de titel van alle CD's. Bij het klikken op een artiest of titel in deze tabel worden alle gegevens van de CD bovenaan weergeven.

Artiest: Percy Sledge
Titel: When a man loves a woman
Jaar: 1987
Land: USA
Label: Atlantic
Prijs: 8.70

<< >>

Klik op een artiest of titel in de tabel om de CD informatie te raadplegen.

Bob Dylan	Empire Burlesque
Bonnie Tyler	Hide your heart
Dolly Parton	Greatest Hits
Gary Moore	Still got the blues
Eros Ramazzotti	Eros

JSON naar XML

Pas de oefeningen uit het vorige hoofdstuk aan zodat XML in plaats van JSON gebruikt wordt.

jQuery

Introductie

Voorkennis

Voor je begint met het aanleren van jQuery, heb je een basiskennis van de volgende onderdelen nodig:

- HTML
- CSS
- JavaScript

Wat is jQuery?

jQuery is een lichtgewicht, "write less, do more", JavaScript bibliotheek.

Het doel van jQuery is om het gebruik van JavaScript op uw webpagina's te vereenvoudigen.

Zo pakt jQuery veel voorkomende taken, waarvoor meerdere regels JavaScript code nodig is, samen tot methoden in één enkele regel JavaScript code.

jQuery vereenvoudigt het gebruik van complexe JavaScript taken, zoals AJAX en DOM manipulaties.

De jQuery bibliotheek heeft de volgende mogelijkheden:

- HTML/DOM manipulaties
- CSS manipulaties
- HTML events verwerking
- Effecten en animatie
- AJAX
- Allerlei hulpmiddelen

Als aanvulling kan jQuery plug-ins laden voor bijna elke taak.

Waarom jQuery gebruiken?

Er bestaan naast jQuery verschillende alternatieve JavaScript bibliotheken, jQuery blijkt voor het ogenblik de meest populaire en meest uitbreidbare bibliotheek (grootste aantal plug-ins).

Grote bedrijven zoals Google, Microsoft, IBM en Netflix gebruiken jQuery op hun sites.

Daar elke browser (en versie) in bepaalde gevallen verschillend reageert, probeert het jQuery team deze browser specifieke gedragingen op te vangen, waardoor met jQuery geschreven JavaScript code op alle browsers op een gelijkaardige manier werkt.

jQuery gebruiken

jQuery downloaden

Er zijn twee versies van jQuery:

- Production: deze versie is verkleind en gecomprimeerd en dus geschikt voor het gebruik op webpagina's op het internet.
- Development: deze versie wordt gebruikt om te testen en verder te ontwikkelen (leesbare JavaScript code).

Beide versies kun je vanaf jquery.com downloaden.

Let op in welke map je de bibliotheek in uw sitestructuur opslaat (in dezelfde map als de HTML pagina waarin je jQuery gebruikt, zorgt voor de minste kopzorgen).

jQuery invoegen

Om jQuery te kunnen gebruiken, moet je de jQuery bibliotheek downloaden en in de webpagina opnemen.

De jQuery bibliotheek bestaat uit één JavaScript bestand en voeg je met de volgende code in de head tag toe aan een webpagina.

```
<script src="jquery.js"></script>
```

Het **type="text/javascript"** attribuut is niet verplicht in HTML5. JavaScript is de standaard script taal in HTML5 en alle moderne browsers.

CDN alternatief

Als je jQuery niet wilt downloaden en op uw site plaatsen, kan je gebruik maken van een CDN (Content Delivery Network) zoals Google en Microsoft.

De code om jQuery via Google te gebruiken:

```
<script
src="http://ajax.googleapis.com/ajax/libs/jquery/1.8.2/jquery.min.js"><
/script>
```

Om de meest recente versie van jQuery via Google te gebruiken, verwijder je een gedeelte van de versie aanduiding. Bij het gebruik van versie 1.8 zal Google de meest recente versie uit de 1.8 reeks doorsturen (1.8.0, 1.8.1, 1.8.2, enz.). Bij het gebruik van versie 1 zal Google de meest recente versie in de 1 reeks doorsturen (van 1.1.0 tot 1.9.9).

De code om jQuery via Microsoft te gebruiken:

```
<script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-
1.8.2.js"></script>
```

Het gebruik van een CDN heeft als grote voordeel, dat veel gebruikers reeds jQuery via een andere site hebben gedownload. De jQuery bibliotheek zit daardoor in de cache van uw browser en wordt dus geen tweede keer gedownload. M.a.w. uw webpagina werkt sneller. Daarnaast hebben zowel Google als Microsoft overal op de wereld webservers staan, waardoor de gebruiker de bibliotheek steeds van een server bij hem of haar in de buurt ophaalt.

Voorbeeld

Het volgende voorbeeld demonstreert de jQuery hide() methode. Bij het klikken op een knop, worden alle alinea's (<p> tags) verborgen.

```
1   <!DOCTYPE html>
2   <html>
3   <head>
4   <title>jQuery voorbeeld</title>
5   <script src="jquery.js"></script>
6   <script>
7   $(document).ready(function(){
8     $("button").click(function(){
9       $("p").hide();
10    });
11  });
12  </script>
13  </head>
14  <body>
15  <h2>jQuery voorbeeld</h2>
```

```
<p>Klik op onderstaande knop.</p>
<p>Hou daarbij deze twee alinea's in het oog.</p>
<button>Klik hier</button>
</body>
</html>
```

jQuery syntaxis

De jQuery syntaxis is ontwikkeld om HTML tags te selecteren en er acties op uit te voeren.

Syntaxis

```
jQueryObject = $(kiezer).actie()
```

Het **dollarteken** is het sein dat je jQuery wilt gebruiken.

De **kiezer** zoekt naar een HTML tags.

De **actie** wordt op de gevonden HTML tags uitgevoerd.

De variabele **jQueryObject** bevat na het uitvoeren van de jQuery actie het object waarop de actie werd uitgevoerd (gebruikte methode), handig om later nog jQuery acties op dezelfde HTML tag uit te voeren:

```
jQueryObject.actie()
```

Voorbeelden

$(this).hide() - verbergt de huidige tag.

$("p").hide() - verbergt alle <p> tags.

$(".test").hide() - verbergt alle tags met class="test".

tweedeActie = $("#test").hide() - verbergt de tag met het id="test" en bewaar het jQuery object voor later gebruik.

tweedeActie.show() – toont alle tags opgeslagen in het jQuery object tweedeActie (in ons voorbeeld de tag met het id="test").

Het Document Ready event

Alle jQuery methoden in ons voorbeeld staan in het document ready event:

```
$(document).ready(function(){
  // hier worden jQuery methoden gebruikt...
});
```

Dit voorkomt het uitvoeren van jQuery code voordat de webpagina volledig is geladen (klaar is).

Door het wachten op de webpagina tot hij klaar is, kun je de JavaScript code in de head tag van uw webpagina opnemen.

Sommige acties mislukken indien de webpagina nog niet volledig geladen is:

- het verbergen van een tag dat nog niet bestaat
- het opvragen van de afmetingen van een afbeelding die nog niet geladen is.

jQuery bevat een verkorte syntaxis voor het document ready event:

```
$(function(){
  // hier worden jQuery methoden gebruikt...
});
```

jQuery kiezers

jQuery kiezers (selectors) selecteren HTML tags.

De jQuery kiezers kunnen HTML tags selecteren op basis van hun id, klasse, attributen, attribuut waarden , enz. De kiezer is gebaseerd op bestaande CSS kiezers aangevuld met eigen kiezers.

Alle kiezers in jQuery beginnen met een dollarteken en staat tussen haken: $().

De tag kiezer

De jQuery tag kiezer selecteert HTML tags op basis van hun tagnaam.

De code om alle <p> tags op een webpagina te selecteren:

```
$("p")
```

De #id kiezer

De jQuery #id kiezer gebruikt het id attribuut van HTML tags om tags te selecteren.

Id's zijn uniek op een webpagina. De #id kiezer wordt dus gebruikt om één unieke tag op een webpagina te selecteren.

De code om de tag met het id test te selecteren:

```
$("#test")
```

De .klasse kiezer

De jQuery klasse kiezer selecteert HTML tags met een specifieke CSS klasse.

De code om de tags met de klasse test te selecteren:

```
$(".test")
```

Nog meer jQuery kiezers

Syntaxis	Beschrijving
$("*")	Alle tags selecteren
$(this)	De huidige tag selecteren (this stelt het adres van een HTML tag voor)
$("p.intro")	De <p> tags met de klasse .intro selecteren
$("p:first")	De eerste <p> tag selecteren
$("p:last")	De laatste <p> tag selecteren
$("ul li:first")	De eerste tag van de eerste tag selecteren
$("ul li:first-child")	De eerste tag van elke tag selecteren
$("[href]")	Alle tags met een href attribuut selecteren
$("a[target='_blank']")	Selecteer alle <a> tags met een target attribuut gelijk aan _blank
$("a[target!='_blank']")	Selecteer alle <a> tags met een target verschillend van _blank
$(":button")	Selecteer alle <button> en <input> met type="button" tags
$("tr:even")	Selecteer alle even <tr> tags
$("tr:odd")	Selecteer alle oneven <tr> tags

Syntaxis	Beschrijving
$("#navigatie").find("a:first")	Selecteert in de HTML tag met het id #navigatie de eerste <a> tag

Een volledig overzicht van alle jQuery kiezers vind je op http://www.w3schools.com/jquery/jquery_ref_selectors.asp.

jQuery Events

De behandeling van events (gebeurtenissen op de webpagina) behoren tot de basisfuncties van jQuery.

De behandeling van events gebeurt met behulp van event handlers (acties of methoden die uitgevoerd worden na het opmerken van een gebeurtenis op een webpagina).

Meestal worden de event handler methoden in de head tag van de webpagina geplaatst.

```
<head>
<script src="jquery.js"></script>
<script>
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
```

Het voorbeeld zorgt dat bij het klikken op een button tag de alinea tags verborgen worden.

Syntaxis

```
$(kiezer).event(function(){
  // bij het event uit te voeren JavaScript code
});
```

jQuery events

Event	Beschrijving
$(document).ready(function)	Definieert een functie voor de onload event van de pagina
$(**kiezer**).click(function)	Definieert een functie voor onclick events op de geselecteerde tags
$(**kiezer**).dblclick(function)	Definieert een functie voor ondblclick events op de geselecteerde tags
$(**kiezer**).focus(function)	Definieert een functie voor onfocus events op de geselecteerde tags
$(**kiezer**).mouseover(function)	Definieert een functie voor onmouseover events op de geselecteerde tags
$(**kiezer**).unload(function)	Definieert een functie voor de onunload event van de geselecteerde tags

Een volledig overzicht van jQuery events staat op http://www.w3schools.com/jquery/jquery_ref_events.asp

jQuery Effecten

jQuery hide() en show()

Met de jQuery hide() en show methoden kunnen HTML tags verborgen worden en terug zichtbaar worden.

Voorbeeld:

```
$("#hide").click(function(){
  $("p").hide();
});
$("#show").click(function(){
  $("p").show();
});
```

Syntaxis:

```
$(kiezer).hide(snelheid, terugroep);
$(kiezer).show(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag verdwijnt of verschijnt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het verbergen of verschijnen wordt uitgevoerd.

Voorbeeld met het argument snelheid:

```
$("button").click(function(){
  $("p").hide(1000);
});
```

jQuery toggle()

Om met jQuery te schakelen tussen de hide() en show() methoden gebruik je de toggle() methode.

De zichtbare tags verdwijnen, terwijl de onzichtbare tags verschijnen.

Voorbeeld:

```
$("button").click(function(){
  $("p").toggle();
});
```

Syntaxis:

```
$(kiezer).toggle(snelheid, terugroep);
```

Oefeningen

Tabbladen

De HTML code voor deze opdracht:

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-
    8" />
```

```
5   <title>Tabbladen</title>
6   </head>
7   <body>
8   <span>HTML</span>
9   <span>CSS</span>
10  <span>JavaScript</span>
11  <span>PHP</span>
12  <div class="tab" id="tab1">HTML verzorgt de structuur en de inhoud
     van de webpagina.</div>
13  <div class="tab" id="tab2">CSS is verantwoordelijk voor de
     opmaak.</div>
14  <div class="tab" id="tab3">JavaScript voegt functionaliteit
     toe.</div>
15  <div class="tab" id="tab4">PHP maakt dynamische onderdelen op de
     webserver aan.</div>
16  </body>
17  </html>
```

Zorg dat je op deze pagina jQuery kunt gebruiken.

Schrijf de jQuery code om na het laden van de webpagina alle tags met de klasse .tab te verbergen. Daarna zorg je dat de eerste tag met de klasse .tab terug wordt weergegeven.

Schrijf een functie waarmee je bij het klikken op een tabblad het bijhorende tab weergeeft en alle andere tabs verbergt.

Zorg voor een geleidelijke overgang bij het weergeven en verbergen van een tab (overgangstijd: 1 seconde).

Om het geheel af te werken gebruik je de volgende CSS opmaak:

```
<style type="text/css">
.tabblad {
  color: #fff;
  padding-bottom: 3px;
  padding-top: 2px;
  padding-right: 3px;
  padding-left: 3px;
  box-shadow: 3px 3px 6px rgba(0, 0, 0, 0.20),
        0px 0px 3px rgba(0, 0, 0, 0.10),
        inset 0px 5px 12px #004499,
        inset 0px 25px 25px rgba(59, 131, 240, .50),
        inset 0px 25px 25px #004499;
  border-radius: 5px 5px 0px 0px;
  text-shadow: 1px 1px 1px #000;
  cursor: pointer;
}
.tabblad:hover {
    box-shadow:
        2px 2px 4px rgba(0,0,0,0.20),
        0px 0px 3px rgba(0, 0, 0, 0.10),
        inset 0px 10px 12px #004499,
        inset 0px 20px 2px rgba(59, 131, 240, .7),
        inset 0px 25px 25px #004499;
    text-shadow: -1px -1px -1px #000;
}
.tab {
  box-shadow: 3px 3px 6px #004499;
  margin-top: 3px;
  border: 1px solid rgba(59, 131, 240, 1);
  padding-top: 0px;
```

```
  padding-right: 5px;
  padding-bottom: 3px;
  padding-left: 5px;
}
</style>
```

Accordeon

Maak met jQuery het volgende systeem:

- Bij het klikken op een div wordt een div binnen de aangeklikte div met een overgang weergegeven (geopend).
- Na het laden van de pagina is enkel de div binnen de eerste aanklikbare div zichtbaar.
- Er mag nooit meer dan één openklikbare div geopend zijn.

Aanschouw de werking van dit systeem bij de oplossingen op het internet.

Om het systeem af te werken, kan je de volgende CSS opmaak gebruiken:

```
<style type="text/css">
.accordeon {
  color: #fff;
  padding-bottom: 3px;
  padding-top: 2px;
  padding-right: 3px;
  padding-left: 3px;
  box-shadow: 3px 3px 6px rgba(0, 0, 0, 0.20),
        0px 0px 3px rgba(0, 0, 0, 0.10),
        inset 0px 5px 12px #004499,
        inset 0px 25px 25px rgba(59, 131, 240, .50),
        inset 0px 25px 25px #004499;
  text-shadow: 1px 1px 1px #000;
  cursor: pointer;
}
.accordeon:hover {
    box-shadow:
        2px 2px 4px rgba(0,0,0,0.20),
        0px 0px 3px rgba(0, 0, 0, 0.10),
        inset 0px 10px 12px #004499,
        inset 0px 20px 2px rgba(59, 131, 240, .7),
        inset 0px 25px 25px #004499;
    text-shadow: -1px -1px -1px #000;
}
.accordeon_inhoud {
  color: #000;
  background-color: #fff;
  box-shadow: 3px 3px 6px #004499;
  margin-top: 3px;
  border: 1px solid rgba(59, 131, 240, 1);
  padding-top: 0px;
  padding-right: 5px;
  padding-bottom: 3px;
  padding-left: 5px;
  text-shadow: 0px 0px 0px #000;
```

```
  cursor: default;
}
</style>
```

jQuery Fading

jQuery heeft acties om HTML tags geleidelijk te laten verdwijnen of op te laten komen.

jQuery fadeIn() methode

De jQuery fadeIn() methode wordt gebruikt om HTML elementen geleidelijk op te laten komen.

Syntaxis

```
$(kiezer).fadeIn(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag verschijnt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het verschijnen wordt uitgevoerd.

Voorbeeld met het argument snelheid:

```
$("#fotopaneel").fadeIn("slow");
```

jQuery fadeOut() methode

De jQuery fadeOut() methode wordt gebruikt om HTML elementen geleidelijk te laten verdwijnen.

Syntaxis

```
$(kiezer).fadeOut(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag verdwijnt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het verbergen wordt uitgevoerd.

Voorbeeld met het argument snelheid:

```
$("#fotopaneel").fadeOut("slow");
```

jQuery fadeToggle() methode

De jQuery fadeToggle() methode wordt gebruikt om zichtbare HTML elementen geleidelijk te laten verdwijnen en verborgen HTML elementen geleidelijk op te laten komen.

Syntaxis

```
$(kiezer).fadeToggle(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag verdwijnt of opkomt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het verbergen of opkomen wordt uitgevoerd.

Voorbeeld met het argument snelheid:

```
$("#fotopaneel").fadeToggle("slow");
```

jQuery fadeTo() methode

De jQuery fadeTo() methode wordt gebruikt om HTML elementen geleidelijk naar een bepaalde dekkingsgraad te laten evolueren.

Syntaxis

```
$(kiezer).fadeTo(snelheid, dekking, terugroep);
```

Het verplichte argument **snelheid** bepaald de snelheid waarmee de tag verdwijnt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het verplichte argument **dekking** (opacity) bepaald de einddekkingsgraad. De dekking heeft een waarde tussen 0 (geen dekking = volledig doorzichtig) en 1 (maximale dekking = volledig ondoorzichtig).

Het facultatieve argument **terugroep** is de naam van een functie die na het verbergen wordt uitgevoerd.

Voorbeeld:

```
$("#fotopaneel").fadeTo("slow", 0.4);
```

HTML eigenschap aanpassen

De jQuery attr() methode wordt gebruikt om HTML eigenschappen (attributen) van tags aan te passen.

Het volgende voorbeeld past de koppeling (href eigenschap) van een a tag met het id #cursus aan:

```
$("#cursus").attr("href","http://webdesign.pindanet.be");
```

Oefeningen

Pagina's in- en uitfaden

De HTML code voor deze opdracht:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Pagina's in- en uitfaden</title>
6   </head>
7   <body>
8   <p>Klik <a href="#">hier</a> om deze pagina nogmaals te laden.
9   </p>
10  </body>
11  </html>
```

Zorg dat je op deze pagina jQuery kunt gebruiken.

Schrijf een functie om bij het klikken op de a tag de pagina (body tag) te vervagen en dezelfde pagina opnieuw te laden:

- De te laden pagina geef je als argument mee aan de onclick functie.
- Het uitfaden zelf duurt 1,5 seconden.
- Na het uitfaden kun je met de volgende opdracht een in de variabele url opgeslagen pagina laden:

```
window.location.href = url;
```

Schrijf de jQuery code om na het laden van de webpagina de pagina te verbergen. Daarna zorg je dat de pagina in 1,5 seconde opkomt.

Banner

De HTML code voor deze opdracht:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Banner</title>
6   <style type="text/css">
7   .banner {
8     position: absolute;
9     top: 0px;
10  }
11  </style>
12  </head>
13  <body>
14  <img class="banner" src="banners/banner1.png" width="905"
     height="324" />
15  <img class="banner" src="banners/banner2.png" width="905"
     height="324" />
16  </p>
17  </body>
18  </html>
```

Zorg dat je op deze pagina jQuery kunt gebruiken.

Na de laden van de webpagina wordt de laatste tag met de klasse .banner verborgen.

Schrijf een functie waarmee je in 2 seconden de zichtbare foto geleidelijk laat verdwijnen en de niet zichtbare foto geleidelijk laat opkomen.

Deze functie wordt na het laden van de pagina uitgevoerd en telkens na het voltooien van het geleidelijk verdwijnen (opkomen) van de foto. Zo maak je een continue overvloei-overgang van de ene naar de andere foto.

Om de overgangsduur op een eenvoudige centrale manier te kunnen aanpassen, plaats je vooraan het script een variabele waarin de overgangsduur wordt opgeslagen. Gebruik deze variabele op de desbetreffende plaatsen in het script.

Breidt het script uit zodat de niet zichtbare afbeelding door de volgende banner wordt vervangen. Tip: gebruik de attr() methode om de src eigenschap van de afbeelding aan te passen.

Zorg dat na het weergeven van de laatste afbeelding (banner28.png), terug de eerste afbeelding wordt geladen.

Breidt het script uit zodat na de overvloei-overgang de zichtbare foto 4 seconden blijft staan, voor de volgende overvloei-overgang start.

Gebruik voor de wachttijd tussen de overvloei-overgangen een globaal aan te passen variabele.

jQuery Sliding

De jQuery slide methoden worden gebruikt om tags uit- en in te vouwen.

jQuery slideDown() methode

De jQuery slideDown() methode wordt gebruikt om tags uit te vouwen.

Syntaxis

```
$(kiezer).slideDown(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag uitvouwt. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het uitvouwen wordt uitgevoerd.

Voorbeeld:

```
$("#paneel").slideDown();
```

jQuery slideUp() methode

De jQuery slideUp() methode wordt gebruikt om tags in te vouwen.

Syntaxis

```
$(kiezer).slideUp(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag wordt ingevouwen. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het invouwen wordt uitgevoerd.

Voorbeeld:

```
$("#paneel").slideUp();
```

jQuery slideToggle() methode

De jQuery slideToggle() methode wordt gebruikt om niet zichtbare tags uit te vouwen en zichtbare tags in te vouwen.

Syntaxis

```
$(kiezer).slideToggle(snelheid, terugroep);
```

Het facultatieve argument **snelheid** bepaald de snelheid waarmee de tag wordt in- of uitgevouwen. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het in- of uitvouwen wordt uitgevoerd.

Voorbeeld:

```
$("#paneel").slideToggle();
```

jQuery css() methode

De css() methode vraagt of past een CSS eigenschap van de geselecteerde tags op of aan.

Een CSS eigenschap opvragen

Syntaxis

```
css("css-eigenschap");
```

Het volgende voorbeeld vraagt de CSS achtergrondkleur van de alinea-tags op:

```
$("p").css("background-color");
```

Een CSS eigenschap aanpassen

Syntaxis

```
css("css-eigenschap","waarde");
```

Het volgende voorbeeld geeft alle alinea-tags een gele achtergrondkleur:

```
$("p").css("background-color","yellow");
```

Meerdere CSS eigenschappen aanpassen

Syntaxis

```
css({"css-eigenschap":"waarde","css-eigenschap":"waarde",...});
```

Het volgende voorbeeld geeft alle alinea-tags een gele achtergrond en een dubbele lettergrootte:

```
$("p").css({"background-color":"yellow","font-size":"200%"});
```

Opborrelende gebeurtenissen

Bij het klikken op een tag wordt geeft de browser het klikken door aan alle bovenliggende tags. Een voorbeeld: als je klikt op een knop in een formulier klikt, wordt het onclick event niet alleen aan de knop doorgegeven, maar ook aan het formulier (form tag) waarin de knop staat en de pagina (body tag) waarin het formulier staat.

Dit stoort niet zolang er geen onclick functies gestart worden die elkaar kunnen storen.

Het opborrelen van een onclick event op een p tag naar een hogere tag kun je met de volgende JavaScript code blokkeren:

```
$("p").click(function(event){
  event.stopPropagation();
  ... het vervolg van de onclick functie ...
});
```

De functie **event.stopPropagation()** stopt het doorgeven van de gebeurtenis (event). Let op de variabele **event** die als argument met de functie wordt meegegeven.

Oefening

Boomstructuur

De HTML code voor deze opdracht:

```
1   <!doctype html>
```

```
2   <html>
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Boom</title>
6   <style type="text/css">
7   html {
8       height: 100%;
9   }
10  body {
11      margin: 0px;
12      height: 100%;
13  }
14  .beschikbarehoogte {
15      height: 100%;
16  }
17
18  #boom {
19      padding-left: 24px;
20      margin-left: 0px;
21  }
22  #boom ul {
23      padding-left: 24px;
24      margin-left: 0px;
25  }
26  #boom li {
27      /* Selecteren beletten */
28      -webkit-touch-callout: none;
29      -webkit-user-select: none;
30      -khtml-user-select: none;
31      -moz-user-select: none;
32      -ms-user-select: none;
33      user-select: none;
34      list-style-type: none;
35      cursor: pointer;
36  }
37  </style>
38  </head>
39  <body>
40  <table width="100%%" border="0" class="beschikbarehoogte">
41    <tr>
42      <td width="200" valign="top">
43        <ul id="boom">
44          <li class="submenu">Webdesign
45            <ul>
46              <li class="submenu">Dreamweaver
47                <ul>
48                  <li class="item"><a
     href="boom/webdesign/dreamweaver/deel1.html" target="inhoud">Deel
     1</a></li>
49                  <li class="item"><a
     href="boom/webdesign/dreamweaver/deel2.html" target="inhoud">Deel
     2</a></li>
50                  <li class="item"><a
     href="boom/webdesign/dreamweaver/eindwerk.html"
     target="inhoud">Eindwerk</a></li>
51                </ul>
52              </li>
```

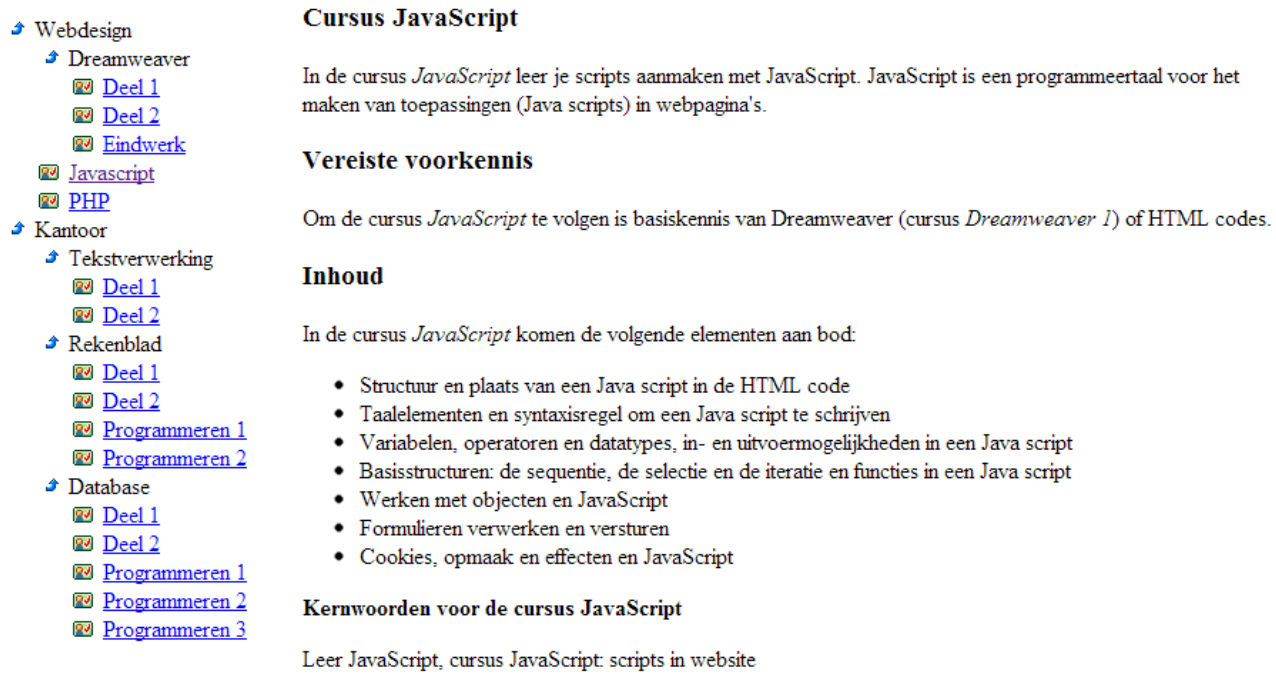
```
53            <li class="item"><a
    href="boom/webdesign/javascript/javascript.html"
    target="inhoud">Javascript</a></li>
54            <li class="item"><a href="boom/webdesign/php/php.html"
    target="inhoud">PHP</a></li>
55          </ul>
56        </li>
57        <li class="submenu">Kantoor
58          <ul>
59            <li class="submenu">Tekstverwerking
60              <ul>
61                <li class="item"><a
    href="boom/kantoor/tekstverwerking/deel1.html" target="inhoud">Deel
    1</a></li>
62                <li class="item"><a
    href="boom/kantoor/tekstverwerking/deel2.html" target="inhoud">Deel
    2</a></li>
63              </ul>
64            </li>
65            <li class="submenu">Rekenblad
66              <ul>
67                <li class="item"><a
    href="boom/kantoor/rekenblad/deel1.html" target="inhoud">Deel
    1</a></li>
68                <li class="item"><a
    href="boom/kantoor/rekenblad/deel2.html" target="inhoud">Deel
    2</a></li>
69                <li class="item"><a
    href="boom/kantoor/rekenblad/programmeren1.html"
    target="inhoud">Programmeren 1</a></li>
70                <li class="item"><a
    href="boom/kantoor/rekenblad/programmeren2.html"
    target="inhoud">Programmeren 2</a></li>
71              </ul>
72            </li>
73            <li class="submenu">Database
74              <ul>
75                <li class="item"><a
    href="boom/kantoor/database/deel1.html" target="inhoud">Deel
    1</a></li>
76                <li class="item"><a
    href="boom/kantoor/database/deel2.html" target="inhoud">Deel
    2</a></li>
77                <li class="item"><a
    href="boom/kantoor/database/programmeren1.html"
    target="inhoud">Programmeren 1</a></li>
78                <li class="item"><a
    href="boom/kantoor/database/programmeren2.html"
    target="inhoud">Programmeren 2</a></li>
79                <li class="item"><a
    href="boom/kantoor/database/programmeren3.html"
    target="inhoud">Programmeren 3</a></li>
80              </ul>
81            </li>
82          </ul>
83        </li>
84      </ul>
85    </td>
```

```
86      <td class="beschikbarehoogte">
87          <iframe src="" name="inhoud" width="100%" height="100%"
    scrolling="auto" frameborder="0"></iframe>
88      </td>
89    </tr>
90  </table>
91  </body>
92  </html>
```

Zorg dat je op deze pagina jQuery kunt gebruiken.

Na de laden van de webpagina

- worden in alle tags met de klasse .submenu alle ul tags verborgen,
- krijgen alle tags met de klasse .submenu de afbeelding boom/openen.gif als opsommingsteken (via de css eigenschap **list-style-image** die je de waarde **url(boom/openen.gif)** geeft),
- krijgen alle tags met de klasse .item de afbeelding boom/lezen.gif als opsommingsteken,
- bij het klikken op een tag met de klasse .submenu wordt een functie uitgevoerd die:
  - ✓ om de eerste in de tag voorkomende ingevouwen ul tag uit te vouwen of omgekeerd,
  - ✓ bij een ingevouwen submenu gebruik je als opsommingsteken de afbeelding boom/openen.gif, bij een uitgevouwen submenu gebruik je als opsommingsteken de afbeelding boom/sluiten.gif,
  - ✓ zorg dat bij het sluiten van een submenu of het klikken op een item het volledige menu niet wordt ingevouwen.



jQuery Animatie

Met de jQuery animate() methode kun je animaties maken.

De animate() methode

Syntaxis

```
$(kiezer).animate({CSS_eigenschappen}, snelheid, terugroep);
```

Het argument **CSS_eigenschappen** bevat de te animeren CSS eigenschap(pen).

Het facultatieve argument **snelheid** bepaald de snelheid waarmee animatie wordt uitgevoerd. Je gebruikt als waarden **slow**, **fast** of een tijd in milliseconden.

Het facultatieve argument **terugroep** is de naam van een functie die na het voltooien van de animatie wordt uitgevoerd.

Onderstaande voorbeeld verplaatst een div tag naar rechts tot de CSS eigenschap left 250 px bereikt:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Animatie</title>
6   <script src="jquery.js"></script>
7   <script>
8   $(document).ready(function(){
9     $("button").click(function(){
10      $("div").animate({left:'250px'});
11    });
12  });
13  </script>
14  </head>
15  <body>
16  <button>Animatie starten</button>
17  <p>Standaard hebben alle HTML tags een statische positie en kunnen
     ze niet verplaatst worden.
18  Om de positie te manipuleren, moet je de CSS eigenschap position van
     de tag aanpassen naar relative, fixed of absolute!</p>
19  <div style="background:#98bf21; height:100px; width:100px;
     position:absolute;">
20  </div>
21  </body>
22  </html>
```

Meerdere eigenschappen animeren

Voorbeeld

```
$("button").click(function(){
  $("div").animate({
    left:'250px',
    opacity:'0.5',
    height:'150px',
    width:'150px'
  });
});
```

Opmerkingen

De namen van de CSS eigenschappen gebruiken de camel-case schrijfwijze. M.a.w. in de animate() methode gebruik je paddingLeft voor de CSS eigenschap padding-left, marginRight voor margin-right, enz.

Kleurenanimatie zit niet in de standaard jQuery bibliotheek. Voor kleuranimaties heb je de Color Animation plugin nodig (deze kun je downloaden van jQuery.com).

Relatieve waarden

Het gebruik van relatieve animatiewaarden laat toe animaties de maken die oorspronkelijke CSS eigenschappen aanpassen (laten toenemen += of laten afnemen -=).

Voorbeeld

```
$("button").click(function(){
  $("div").animate({
    left:'250px',
    height:'+=150px',
    width:'+=150px'
  });
});
```

Voorgedefinieerde waarden

De voorgedefinieerde animatie waarden zijn: show, hide en toggle.

Voorbeeld

```
$("button").click(function(){
  $("div").animate({
    height:'toggle'
  });
});
```

Wachtrij

Animaties op één en dezelfde tag kunnen in een wachtrij geplaatst worden om na elkaar uitgevoerd te worden.

Voorbeeld

```
$("button").click(function(){
  var div=$("div");
  div.animate({height:'300px',opacity:'0.4'},"slow");
  div.animate({width:'300px',opacity:'0.8'},"slow");
  div.animate({height:'100px',opacity:'0.4'},"slow");
  div.animate({width:'100px',opacity:'0.8'},"slow");
});
```

Het volgende voorbeeld verplaatst een div tag naar rechts en vergroot nadien de tekst:

```
$("button").click(function(){
  var div=$("div");
  div.animate({left:'100px'},"slow");
  div.animate({fontSize:'3em'},"slow");
});
```

Tip

Gebruik een tweede gelijkaardige animate() methode om in de animatie een pauze in te lassen:

```
$("button").click(function(){
  var div=$("div");
  div.animate({left:'100px'},"slow");
  div.animate({fontSize:'3em'},"slow");
  div.animate({fontSize:'3em'},"slow");
});
```

Om animaties op verschillende tags na elkaar uit te voeren, maak je gebruik van een terugroepfunctie:

Syntaxis

```
$(kiezer).animate({CSS_eigenschappen}, snelheid, function(){
  $(kiezer).animate({CSS_eigenschappen}, snelheid);
});
```

De tweede animate() methode wordt pas uitgevoerd na het voltooien van de eerste animate() methode. Daarbij mag de kiezer van de twee animate() methoden verschillen.

Oefeningen

Voorbeeld afwerken

Werk het voorbeeld verder af zodat alle hierboven vermelde animatie voorbeelden op één pagina met elk een eigen startknop werken (zie oplossing op het internet).

Tip: gebruik relatieve positionering.

Reclamebanner

We maken een geanimeerde reclamebanner op basis van de volgende HTML code:

```
1   <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2   <html xmlns="http://www.w3.org/1999/xhtml">
3   <head>
4   <meta http-equiv="Content-Type" content="text/html; charset=utf-
     8" />
5   <title>Reclame banner</title>
6   <style type="text/css">
7   body {
8       font-family: Verdana, Geneva, sans-serif;
9   }
10  #reclame th {
11      color: #8b921e;
12      border-bottom-width: medium;
13      border-bottom-style: solid;
14      border-bottom-color: #4a2f1e;
15  }
16  #reclamekop {
17      font-size: 300%;
18  }
19  #reclame {
20      color: #4a2f1e;
21      font-size: small;
22  }
23  </style>
24  </head>
25  <body>
26  <table border="0" id="reclame" style="position:relative;">
27    <tr>
28      <td width="118" height="142"><img id="reclameafbeelding"
     src="reclame/koffer.png" width="118" height="142"
     style="position:relative;" /></td>
29      <td>
30          <table width="100%" border="0">
31            <tr>
```

```
32              <th><span id="reclamekop"
    style="position:relative;">SNT</span><span id="reclamekoptekst"
    style="position:relative;"> haal hier uw bagage!</span></th>
33            </tr>
34            <tr>
35              <td align="center">Volwassenenonderwijs overdag en 's
    avonds<br />
36                Arsenaalstraat Brugge | Blankenberge</td>
37            </tr>
38          </table>
39      </td>
40    </tr>
41  </table>
42  </body>
43  </html>
```

Zorg dat je jQuery kunt gebruiken.

SNT haal hier uw bagage!

Volwassenenonderwijs overdag en 's avonds
Arsenaalstraat Brugge | Blankenberge

Gebruik voor de animatiesnelheid voor alle animate() methoden een variabele. Daarmee kun je later op een eenvoudige manier (de variabele een andere waarde geven) de snelheid van de animatie aanpassen.

De automatisch startende animatie bestaat achtereenvolgens uit:

1. Een pauze (Het ondoorzichtig maken van de tags met het id #reclamekop en #reclamekoptekst.)

   Het doorzichtig maken van de tag met het id #reclamekoptekst.

   Het doorzichtig maken van de tag met het id #reclamekop.
2. De tekst in de tag met het id #reclamekop aanpassen naar **Webdesign** met de opdracht:

   $("#reclamekop").html("Webdesign");

   De tekst in de tag met het id #reclamekoptekst verwijderen.

   Het gelijktijdig ondoorzichtig maken van de tags met de id #reclamekop en #reclamekoptekst.

   Het doorzichtig maken van de tag met het id #reclameafbeelding.
3. Het aanpassen van de reclameafbeelding naar reclame/html_logo.svg.

   Het ondoorzichtig maken van de reclameafbeelding.

   Pauze.

   Het ondoorzichtig maken van de reclameafbeelding.
4. Het aanpassen van de reclameafbeelding naar reclame/css_logo.svg.

   Het ondoorzichtig maken van de reclameafbeelding.

   Pauze.

   De hoogte van de reclameafbeelding verbergen.
5. Het aanpassen van de reclameafbeelding naar reclame/javascript_logo.svg.

   De hoogte van de reclameafbeelding weergeven.

   Pauze.

   De hoogte van de reclameafbeelding verbergen.
6. Het aanpassen van de reclameafbeelding naar reclame/PHP_logo.svg.

De hoogte van de reclameafbeelding weergeven.

Pauze.

7. De reclamekoptekst 100 beeldpunten naar boven en 150 beeldpunten naar rechts verplaatsen.

   De reclamekop 100 beeldpunten naar boven verplaatsen.

8. De reclameafbeelding 150 beeldpunten naar links verplaatsen.
9. In deze laatste animatieblok herstel je de oorspronkelijke situatie.
10. Herstart de animatie.