College of MultiMedia

```
<head>
    <title>HTML5 Page Layout</title>
    <style type="text/css">
        body {
            color: #666666;
            background-color: #f9f8f6;
            background-image: url("images/background.jpg");
            background-position: center;
            font-family: Georgia, Times, serif;
            line-height: 1.4em;
            margin: 0px;
        }
        header {
            height: 160px;
            background-image: url("images/header.jpg");
        }
        footer {
            font-size: 80%;
            padding: 7px 0px 0px 20px;
        }
    </style>
</head>
<body>
    <div class="wrapper">
        <header>
            <h1>Programming Languages</h1>
            <nav>
                <ul>
                    <li><a href="" class="current">HTML</a></li>
                    <li><a href="">Javascript</a></li>
                    <li><a href="">Java</a></li>
                    <li><a href="">Python</a></li>
                    <li><a href="">C++</a></li>
                    <li><a href="">
                </ul>
```

HTML5 CSS3 JS

Front End Developer - Reader 2019 v1.2

College of MultiMedia

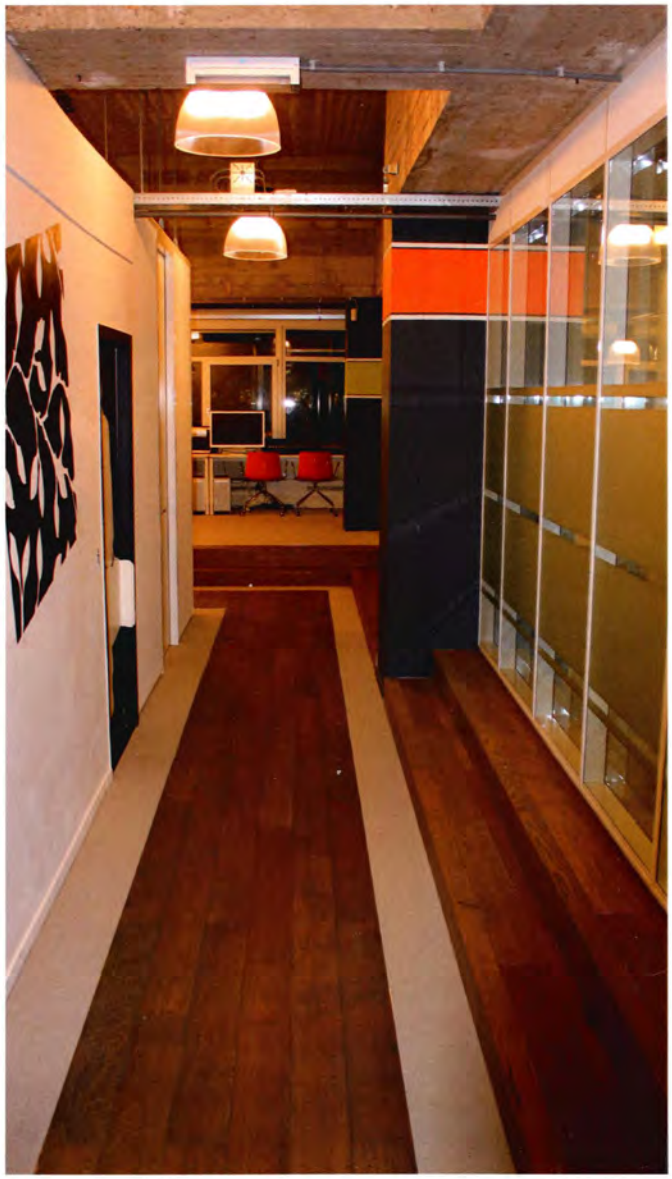
Informatie College of MultiMedia

College of MultiMedia
Egelenburg 150-152
1081 GK Amsterdam

Tel: 020 462 3939
Fax: 020 462 3930

www.cmm.nl
info@cmm.nl

MultiMedia Education BV
KVK Amsterdam 33291946
Rabobank NL12RABO0324675232



De inhoud van deze reader mag niet zonder uitdrukkelijke toestemming van College of MultiMedia worden gereproduceerd, gekopieërd of worden gebruikt voor andere doeleinden dan waarvoor College of MultiMedia deze heeft bedoeld.

Deze reader is samengesteld met eigen lesmaterialen van College of MultiMedia.
Bij gebruik van ander materiaal is de bron bij het desbetreffende onderwerp vermeld.

© 2019 - College of Multimedia

Niet Tot 1

Inhoudsopgave

Reglementen CMM	2
Belangrijke data	3
Studentenvoordeel	4
Cijferlijst	6
Boekenlijst	7
Boeken en magazines	8
Opdrachten	9
Video: CODEC Overzicht PT1	16 1
Video: HTML5 Video PT1	17 1
HTML:API's, Social Media, iFrames	20 1
HTML: Image Maps	21 1
CSS: Device vs Reference Pixel	22 1
CSS:Theorieën	24
CSS: Box Model	26
CSS:Typography	28
CSS: Graphics en Backgrounds	29
CSS: Gradients, Sprite images en Pseudo-classes	30
CSS: Pseudo Elements	31
CSS: Floats en Display Layout	32
CSS: Positioneren	33
CSS:Transitions	34
CSS: @keyframes	35
CSS: Responsive Web Design en Media Queries	36
CSS: Flexbox Layout	38
CSS: Grid Layout	39 IG
Abobe Illustrator CC: Interface	42
Adobe Illustrator CC:Tools	43
Adobe Illustrator CC: Lagen	44 1
Adobe Illustrator CC:Web Optimised SVG's	45
Adobe Animate CC: Geschiedenis	47 2
Adobe Animate CC: Interface	49
Adobe Animate CC:Timeline	50
Adobe Animate CC: Motion Editor	51
Adobe Animate CC: Objecten van Animate	52
Adobe Animate CC:Terminologie	53
JavaScript: Programming Theory	56
JavaScript: Programma Stroom Schema	59
JavaScript: Programma Structuur Diagram	61
JavaScript:Variables en Operators	62
JavaScript: Arrays	65
JavaScript: Control Statements en Loops	66
JavaScript: Functions,Arguments en Objects	68
JavaScript: BOM en DOM	70
JavaScript:Window-object	72
JavaScript: Document-object	73 2
JavaScript: Events	75
JavaScript: jQuery - Write less, do more	77
CSS: Preprocessors	79
Essentials:Webservers	84
Essentials:Virtual Host met MAMP PRO	85
CMS:WordPress Basics	87
CMS:WordPress Installeren	89
CMS:WordPress Back-ups en Verhuizen	90
CMS:WordPress Thema en Template Hierarchy	91
UNIX:Terminal APP	94
NPM: Package Managers	95

1 JS
Animate
RWD

2 GSAP
SCSS

3 NPM (WM-2)
jquery
Node js

Openingstijden
- Wij openen elke dag om 10:00 uur. De praktijkruimtes kunnen worden geboekt vanaf 10:30 uur.
- Maandag t/m donderdag sluiten de praktijkruimtes om 21:30 uur. De lessen duren tot 21:45 uur.
- Op vrijdag en zaterdag sluiten de praktijkruimtes om 17:00 uur. De lessen duren tot 17:15 uur.
- Op zondag is CMM de gehele dag gesloten.

Afmelden
- Tijdens kantooruren; ma. t/m vr. 09:30 - 18:00 uur: per telefoon op 020-462 39 39.
- Buiten kantooruren: alleen via info@cmm.nl.

Binnenkomen en verlaten
- In verband met de veiligheid en beveiliging van jullie en onze spullen, verzoeken wij je om bij het binnenkomen en het verlaten van het gebouw altijd even te controleren of de voordeur goed dicht zit.

Geen drugs en/of alcohol
- Het is in en rondom het gebouw niet toegestaan om drugs en/of alcohol te nuttigen.

Eten en drinken
- Eten en drinken is absoluut niet toegestaan in de praktijkruimtes of in nabijheid van apparatuur!
- Er mag gegeten en gedronken worden in de kantine en in de lobby (relaxruimte rechts naast receptie).
- Houd het gebouw alsjeblieft netjes en laat geen blikjes, papiertjes of andere troep slingeren.

Niet roken in het gebouw
- Overdag en in het weekend is roken in het gebouw niet toegestaan
- 's Avonds mag er in de rookruimte in de kantine worden gerookt. Peuken graag in de asbakken deponeren en niet uit het raam werpen.

De apparatuur en leslokalen
- Het is niet toegestaan apparatuur in het gebouw te verplaatsen of anders aan te sluiten.
- Als een apparaat stuk is of niet goed werkt moet dit onmiddellijk gemeld worden bij de receptie of aan een van de docenten. Als dit niet op tijd wordt gedaan, kan de student aansprakelijk worden gesteld.
- Na gebruik dienen alle werkstations en praktijkruimtes netjes te worden achtergelaten.
- Laat geen bestanden achter op de computer(s) en zorg regelmatig voor een goede back-up van je werk.

Software
- Het is niet toegestaan software van CMM te kopiëren.
- Het is niet toegestaan software te installeren op de computers van CMM.
- Verander geen instellingen van de systeemsoftware op de workstations. De software is met veel zorg geïnstalleerd en ingesteld. Het veranderen van systeeminstellingen kan gevolgen hebben voor de stabiliteit en werking van de programmatuur en/of de computer.

Boeken van Praktijkuren
- Per week kun je bij de receptie tijd boeken voor het werken achter een werkstation in een van de praktijkruimtes.
- Last-minute-boekingen kunnen alleen op de dag zelf worden aangevraagd.
- Je mag je alleen in een praktijkruimte begeven als je één van de werkstations hebt geboekt.

Examen en inleveren van projecten
- Deelneming aan examens is alleen mogelijk indien alle betalingen door CMM zijn ontvangen.
- Na het verstrijken van een deadline voor het inleveren van een praktijkopdracht wordt er 5% van het cijfer afgetrokken voor elke dag dat je te laat bent met inleveren. Een week na de deadline kun je maximaal nog maar 50% halen voor de opdracht en ontvang je geen nakijk-commentaar, bij 3D ontvang je ook geen extra punten.
- Progressietesten worden alleen op de geplande data gemaakt.
- Het inhalen van een progressietest is alleen mogelijk met een goede reden welke voor aanvang van de test bekend moet worden gemaakt. Inhalen van de test dient plaats te vinden vóór de eerstvolgende les na de oorspronkelijke test datum.
- Examens kunnen alleen op de gegeven datum worden gemaakt. Bij het missen van het theorie-examen ben je uitgesloten voor het herexamen. Bij het niet halen van een voldoende cijfer voor het theorie-examen bestaat er de mogelijkheid om een herexamen af te leggen mits je voor je praktijk een voldoende hebt.
  De kosten voor dit herexamen bedragen € 70,-.
- De uitslag van het theorie-examen volgt wanneer alle praktijkopdrachten zijn beoordeeld.
- Een herexamen wordt ± twee weken na het examen gehouden.
- Bij het niet af hebben van je opdrachten op de dag van het praktijkexamen, bestaat er bij bijzondere omstandigheden de mogelijkheid om binnen een maand na het praktijkexamen een herexamen af te leggen. De kosten voor dit praktijkherexamen bedragen € 135,-.
- Alleen in zeer uitzonderlijke gevallen en na tijdig overleg met je docent wordt er van deze regels afgeweken.
- Werkdruk, vakanties of computercrashes zijn geen geldige redenen voor het niet, of te laat, inleveren van praktijkopdrachten.
- Oefeningen en opdrachten dien je persoonlijk bij je docent in te leveren. Niet per e-mail of post.

Overige
- De keuken van CMM is niet toegankelijk voor studenten. Hiermee wordt het kookgedeelte, de koelkast en de koffieapparaten bedoeld, van de tafels en stoelen in de kantine kan wel gebruik worden gemaakt. In de lounge/relaxruimte is een koffie-apparaat voor studenten.
- Studenten worden uiteraard geacht de lessen te volgen. Eventuele absentie wordt, indien verzocht, doorgegeven aan instanties of personen die daar recht op hebben.
- Voor elke opleiding geldt dat je minimaal 80% van alle lessen moet hebben gevolgd om in aanmerking te komen voor het diploma. Wanneer je een les niet kunt bijwonen dien je dit op tijd te melden bij de receptie of je docent.
- Kom op tijd voor de lessen. Het is voor zowel de docent als studenten erg vervelend als een les wordt onderbroken door een student die te laat binnenkomt.
- In het gebouw van CMM is het is niet toegestaan om een ingeschakelde mobiele telefoon te dragen. Alleen tijdens de pauzes mag men gebruik maken van mobiele telefoons.
- In het gebouw van CMM zijn de docentenkamers en de keuken niet toegankelijk voor de studenten, behalve bij directe toestemming. Dit geldt ook voor de ruimte achter de balie van de receptie.
- CMM verzoekt je om de kantine netjes te houden en zaken zoals lege verpakkingen, theezakjes, fruitafval et cetera netjes in de daarvoor bestemde afvalbak te deponeren.
- Indien je je koffiebeker wilt omspoelen kun je dit doen in de wastafels van de toiletten.
- Op afgaande mobieltjes staat de drop-penalty.

Wegens feestdagen is CMM op de volgende dagen gesloten.

Feestdag	Dag van de week	Datum	Weeknummer
Tweede Paasdag 2019	maandag	22 april 2019	17
Koningsdag 2019	zaterdag	27 april 2019	17
Hemelvaartsdag 2019	donderdag	30 mei 2019	22
Tweede Pinksterdag 2019	maandag	10 juni 2019	24

Vakanties CMM

Wegens vakantie is CMM op de volgende dagen gesloten.

De lessen die door deze vakantiedagen komen te vervallen (uitgezonderd zomer- en kerstvakantie) worden aansluitend ingeroosterd of ingehaald en door CMM ingepland. Meestal gebeurt dit op een willekeurige dag in dezelfde week waarin de desbetreffende vakantiedag valt.

Vakantie	Van	Tot en met	Weeknummers
Zomervakantie 2019	maandag 22 juli 2019	zondag 11 augustus 2019	30, 31,32
Kerstvakantie 2019	maandag 23 december 2019	zondag 5 januari 2020	52, 1

Theorietoetsen

Vul hieronder de data van je theorietoetsen in, je krijgt deze van de docent.

Soort toets	Datum

Praktische inleveropdrachten

Vul hieronder de data van je inleveropdrachten in, je krijgt deze van de docent.

Soort opdracht	Inleverdatum

Soft- en hardware

Adobe
Je bent als student 65% voordeliger uit als je de studentenlicentie van Adobe Creative Cloud aanschaft.
https://www.adobe.com/nl/creativecloud/buy/students.html
Dit geldt voor een licentie van 1 jaar.

Maxon
De software voor Cinema 4D kun je aanschaffen bij Maxon of onze partner MediaMachine.
Op vertoon van je CMM-studentenpas of een bewijs van inschrijving kun je een speciale gratis studentenlicentie aanschaffen.
www.maxon.net (gratis studentenlicentie)
www.mediamachine.nl (betaalde studentenlicentie)

Audiovisuele Apparatuur

Camalot
Voor het huren van audiovisuele hardware krijg je als CMM-student 25% korting op de huurtarieven bij Camalot.
(www.camalot.nl)

Budgetcam
Ook bij Budgetcam krijgen studenten 20% korting op de huurtarieven!
(www.budgetcam.nl)

Cursussen

CMM studentenkorting
Wil jij je opleiding aanvullen met een extra vaardigheid of de kennis van een extra softwarepakket ?
Als je als student een opleiding volgt krijg je 20% korting op de meeste van onze cursussen.
Voor meer informatie verwijzen we je naar Algemene Voorwaarden Opleidingen van CMM.

Bovenstaande kortingen en afspraken tussen CMM en partners zijn onderhevig aan wijzigingen.
Hier kunnen dan ook geen rechten aan ontleend worden.

Algemene toetsencombinaties

Shift - ⌘ - N - nieuw document / nieuwe folder op de desktop
⌘ - Q - afsluiten van het programma (quit)
⌘ - W - sluiten van een scherm
⌘ - O - openen van een bestand / openen van geselecteerde folder
⌘ - S - bewaren van een bestand (save)
⌘ - Z - ongedaan maken van laatste actie in een programma (undo)
⌘ - X - knippen van een geselecteerd element (cut)
⌘ - C - kopiëren van een geselecteerd element (copy)
⌘ - V - plakken van een eerder gekopieerd of uitgeknipt element (paste)
⌘ - A - alles selecteren binnen een programma of op de desktop (select all)
⌘ - . - afbreken van een lopende actie
CTRL + klik - equivalent van de rechtermuisknop onder Windows

De meeste commando's zijn in Windows hetzelfde. De ⌘ - toets is dan vaak de CTRL-toets.
Alleen is ⌘ - Q (quit) in Windows meestal alt - F4.

Algemene regeltjes van Mac OS X

Als een bestand vanuit een bepaalde folder naar een andere folder wordt gesleept, dan wordt het bestand verplaatst. Echter, als de folder waarin je het bestand wilt slepen in een andere volume staat (harde schijf, USB-drive, etc), dan wordt het bestand gekopieerd en niet verplaatst.

Tijdens het slepen van een bestand naar een andere folder kun je de Option-toets (alt) ingedrukt houden om het bestand naar die folder te kopiëren. Zoals hierboven uitgelegd gebeurt dit automatisch als de folder op een andere volume staat.

Elk volume heeft zijn eigen bureaublad (desktop) en prullenmand (trash). Alleen zie je fysiek maar één desktop en één trash. Een bestand wordt pas écht verwijderd nadat je onder het Finder menu "Empty Trash" hebt gekozen.

Voordat je een removable volume (zoals een externe harde schijf of USB-drive) ontkoppelt, moet je deze drive uitwerpen (eject). Dit doe je door het naar de prullenmand te slepen. De prullenmand verandert tijdens het slepen in een eject-symbool.

Zet een Apple computer altijd uit met het 'Shut Down'-commando in het Apple menu.

Blacklist of (t)error:

A) Verboden tekens in bestandsnamen:
- /
- \
- &
- *
- ( )
- ILIKETOSCREAMFILENAMES
- URL of weblink

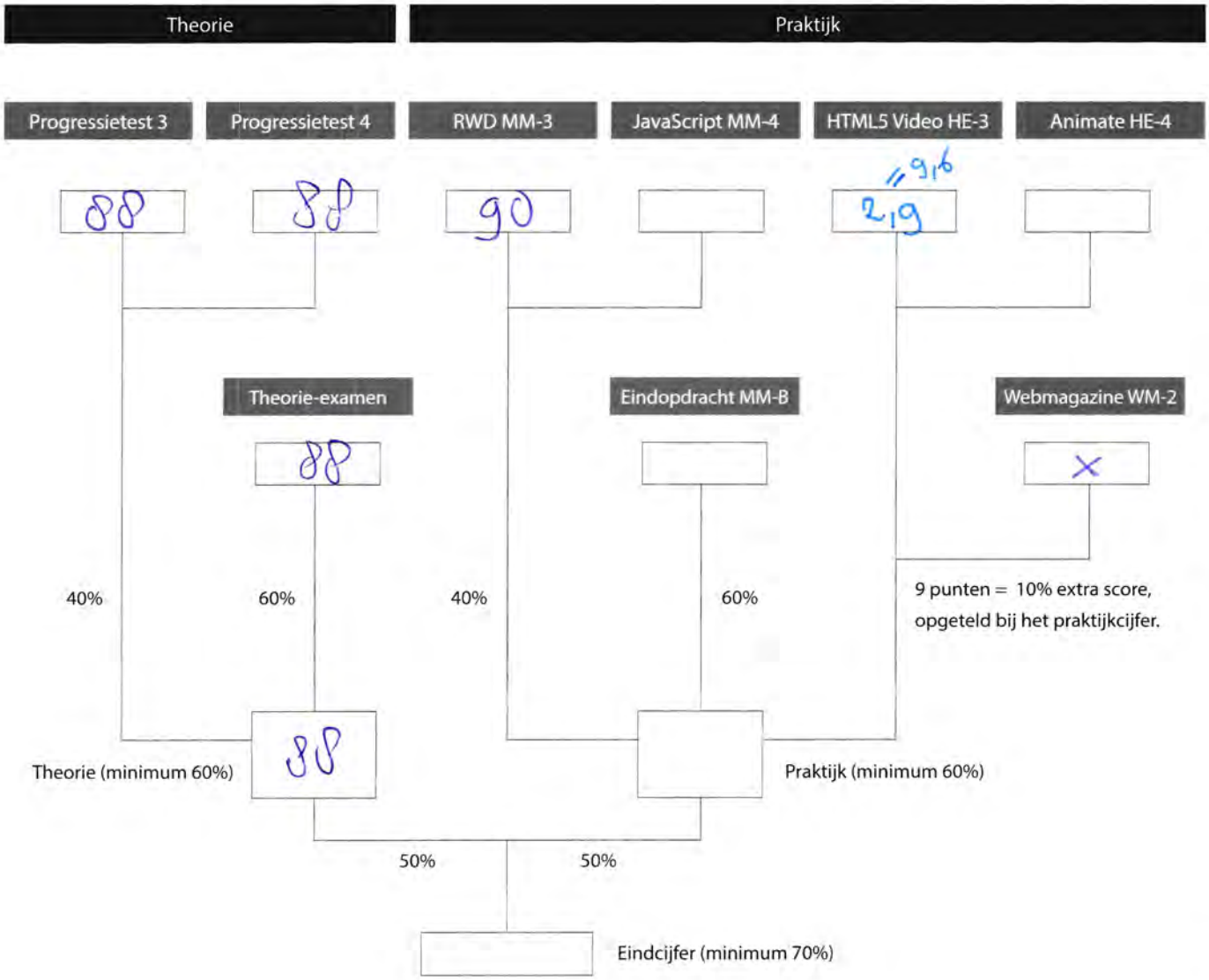
B) Foute bestandsnamen en extenties:
- myFile.jpeg
- myFile.tiff
- myFile

C) Foute terminologie:
- def / verydef / def2
- final / veryfinal / finaldef
- thingy / dingetje / stuff
- myName1. Name2

Werk ***nooit*** vanaf een removable medium (USB-drive). Sleep werkbestanden eerst naar de Desktop. Ga vervolgens aan het werk. Na afloop kopieër je de bestanden weer terug naar de USB-drive.

Algemene info

Cijferlijst



CSS: The Missing Manual (4th Edition)

Auteur: David Sawyer McFarland
Uitgever: O'Reilly
ISBN-13: 978-1-491-91805-0

CSS3 Web Development Library (Optioneel)

Auteurs: Peter Doolaard & Peter Kassenaar
Uitgever: Van Duuren Informatie
ISBN: 978-90-5940-809-8

Adobe Animate CC Classroom in a Book (2019 release)

Auteur: Russell Chun
Uitgever: Adobe Press
ISBN-10: 0-13-529984-5
ISBN-13: 978-0-13-529984-5

JavaScript & Jquery: Interactive Front-end Web Development

Auteur: Jon Duckett
Uitgever: John Wiley & Sons, Inc.
ISBN: 978-1-118-87165-2

Op bol.com wordt het boek vaak aangeboden in een voordelige set (Web Design with HTML, CSS, JavaScript, jQuery Set).
Het andere boek in de set is HTML & CSS - Design and Build Web Sites (nodig voor module 1 Webdesigner).

JavaScript & jQuery: The Missing Manual (3rd Edition) (Optioneel)

Auteur: David Sawyer McFarland
Uitgever: O'Reilly Media, Inc.
ISBN-13: 978-1-491-94707-4

Gratis e-books en PDF's staan op de MediaServer (MediaServer - Public - Boeken).

Je kunt bovengenoemde boeken verkrijgen via onder andere de volgende websites en winkels:
www.bol.com - www.amazon.com - www.amazon.co.uk - www.comcol.nl (www.managementboek.nl)
Je hebt de boeken op tijd nodig! Bestel zo snel mogelijk en voorkom lange levertijden. Tip: E-books worden direct geleverd.

Aanraders!

- Verdieping JavaScript:
  JavaScript Webdevelopment Library
  Auteur: Peter Kassenaar
  Uitgever: Van Duuren Informatica

- Verdieping JavaScript:
  Training Guide: Programming in HTML5 with JavaScript and CSS3
  Auteur: Glenn Johnson
  Uitgever: MicroSoft Press

- Verdieping JavaScript:
  Learning PHP, MySQL, JavaScript, CSS & HTML5 (4th Edition)
  Auteur: Robin Nixon
  Uitgever: O'Reilly Media (2014)

- Verdieping WordPress:
  Handboek WordPress 4 tweede editie
  Auteur: Dirkjan van Ittersum
  Uitgever: Van Duuren Informatice (2016)

- Verdieping WordPress:
  WordPress thema's bouwen
  Auteur: DirkJan van Ittersum
  Uitgever: Van Duuren Informatice (2015)

Interessante boeken

- Adobe Illustrator CC: Classroom in a Book (2018)
  Auteur: Jim Maivald
  Uitgever: Adobe Press

- Visual QuickStart Guide CSS3
  Auteur: Jason Cranford Teague
  Uitgever: Peachpit Press

- Visual QuickStart Guid JavaScript (Ninth Edition)
  Auteur: Tom Negrino & Dori Smith
  Uitgever: Peachpit Press

- jQuery Cookbook
  Auteur: jQuery Community Experts
  Uitgever: O'Reilly Media (2010)

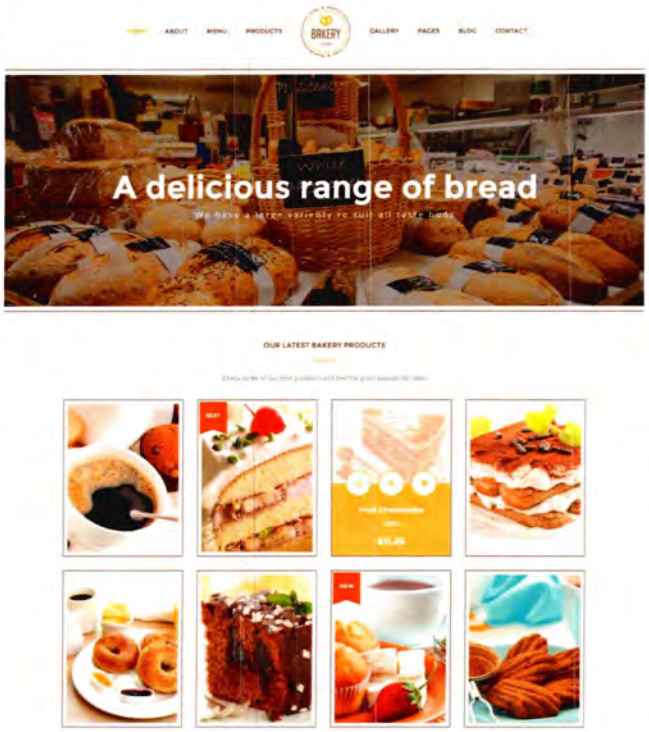
Interessante magazines

- Personal Computer Magazine (PCM)
  pcmweb.nl

- Advanced Photoshop
  advancedphotoshop.co.uk

- Webdesigner
  webdesignermagazine.nl

Opdrachten



Cruise Amsterdam

Iets te vieren of gewoon een keer genieten op het water, al varende kunt u vele mooie, leuke en historische bezienswaardigheden ontdekken. Of het nu gaat om een intiem boottochtje in een historische sloep/salonboot tot een vaarevenement op een partyschip voor grotere gezelschappen. Cruise Amsterdam verzorgt zowel reguliere als exclusieve borrel en dinner cruises door de grachten van Amsterdam en rondom Amsterdam.

ONTDEK CRUISE AMSTERDAM

Opdrachtomschrijving MM-B deel 2

Produceer de website van Cuisine aan de hand van het aangeleverde conceptboek.

RWD
JS
Animate

SCSS
jQuery
GSAP

Nodejs
NPM

Tips en werkwijze

- Gebruik je creativiteit om eventuele ontbrekende, onvolledige of onduidelijke onderdelen zelf in te vullen.
- Het project moet volledig responsive zijn uitgevoerd en goed werken in alle moderne webbrowsers (inclusief mobile en tablet devices).
- Het gebruik van een preprocessor zoals SCSS kan extra punten opleveren. Vergeet niet je bronbestanden aan te leveren, anders kan de CSS niet worden beoordeeld.
- De website moet een aantal JavaScriptonderdelen bevatten. Één daarvan is een uitgebreid contact- en bestelformulier, inclusef HTML5 browservalidatie (pattern en required attributes).
- Ontwikkel a.d.v. de prototypes interactieve UX-elementen met JavaScript of jQuery. Wees creatief; een eenvoudige pop-up of mouse-over leveren niet genoeg punten op.
- Het gebruik van third-party javaScript libraries of frameworks (zoals jQuery Easing of GSAP) is in principe toegestaan, maar alleen jouw JavaScript code wordt beoordeeld.
- De website moet twee Adobe Animate canvasanimaties bevatten. Één is het bedrijfslogo en de ander is het interactieve project "Waar is de koe?".
- De website bevat minimaal één HTML5 video. Alle videobestanden heb je zelf met Adobe Media Encoder of Apple Compressor gecomprimeerd.
- Het gebruik van NodeJS en NPM kan extra punten opleveren.

Inleveren van de opdracht

.md?

- De '_source'-map bevat een overzichtelijke ReadMe, waarin je toelicht in welke webpagina's de onderdelen van jouw website terug te vinden zijn.
- Vergeet niet de bronbestanden aan te leveren van Photoshop, Illustrator, Experience Design en Animate.
- Lever geen bronbestanden aan van video's.
- Lever geen node_modules map aan.
- Zip het hele project en stop het in de Drop Box van de docent. Upload de website met je favoriete FTP-programma naar je cmmstudents.nl subdomein.

Beoordelingscriteria

- File management
  - Correcte, geldige en consistente bestandsnamen, slimme bestandsnamen met aandacht voor SEO
  - Folderstructuur
- HTML
  - Correcte HTML5-syntax en semantiek
  - Opbouw, indeling en structuur van de bestanden
  - Usability en accessibility
- (S)CSS
  - Correcte syntax
  - Opbouw, indeling en structuur van de bestanden
  - Slim gebruik van selectors en pseudo classes
  - Gebruik van media queries en break points
  - Effectief gebruik van de cascade
  - Toepassing van RWD
  - CSS-transitions en/of animations
- JavaScript & jQuery
  - Correcte syntax
  - Opbouw, indeling en structuur van de bestanden
  - Gebruik van variabelen
  - Gebruik van functies
  - Gebruik de DOM / BOM
  - Gebruik van events
  - Gebruik van objecten en JSON
  - Correcte validatie van het formulier
  - Cross-browser compatibility
- Graphics
  - Gebruik van PS / AI / XD Assets generators
  - Optimale compressie, kwaliteit en kwantiteit van graphics
- HTML5 Video Deployment
  - Zelfgeproduceerde video's met logische toepassing
  - Kwaliteit, specificaties en compressie van de videobestanden
  - Correct gebruik van de HTML5 video-tag
- Canvasanimaties
  - Gebruik van Animate en het gebruik van de juiste exportsettings, inclusief RWD
  - Toepassing en bestandgrootte van de canvasanimatie
  - Interactie & creativiteit
  - Correcte implementatie van de animatie in HTML
- Interface en Design
  - Creativiteit en professionaliteit van het webdesign
  - Logica van de interface en leercurve
  - Consistente werking / positionering
  - Toepassing van JavaScript ter verrijking van de interface
  - Gracefull degradation / progressive enhancement / aanwezigheid van alternatieve (fall-back) media
- Algemeen
  - Commentaar in de HTML-, CSS- en JavaScript-code
  - Minimaal 12 webpagina's
  - Inhoudelijke waarde van de website en content
  - Op tijd ingeleverd, ontwikkeling van de student en invulling van de opdracht

Opdrachtomschrijving MM-3

Realiseer aan de hand van drie prototypes een responsive webdesign (RWD) voor de homepage van 'The Bakery'.

Tips en werkwijze

- Eerst denken, dan doen! Maak jezelf eerst bekent met de designs. Je wilt vooraf bepalen hoe je het gaat aanpakken.
- Bepaal een plan van aanpak door elke rij van alle drie de layouts apart te analyseren. Daarna bekijk je wat de overeenkomsten zijn tussen de rijen en de layouts.
- Bepaal de hoeveelheid kolommen per layout, breedtes, verhoudingen en vergeet de gutters niet.
- Het kan helpen om de drie layouts te printen en hier met potlood overheen te schetsen, of maak je eigen wireframes!
- Denk eraan: designers verzinnen soms iets wat in de praktijk lastig of zelfs onmogelijk te maken is! Hiervoor verzin je passende oplossingen die zo goed mogelijk lijken op het origineel.
- Begin met makkelijke gedeeltes, zoals de service en footer. Bewaar lastige gedeeltes voor later.
- Begin *niet* met het schrijven van CSS *vóórdat* je een compleet valide HTML-bestand hebt. Een gedeelte van de HTML5-code is al voor je geschreven. Schrijf overige elementen zelf.
- De HTML-code moet semantisch correct en zo toegankelijk mogelijk zijn opgesteld.
- Aangeleverde HTML-code mag je niet aanpassen door zelf id- of class-attributes toe te voegen. Gebruik hier advanced CSS-selectors zoals +, ~ en >.
- Je mag in de aangeleverde HTML-code wel je eigen code toevoegen (zoals een container div).
- Pas een CSS-reset toe. Overige code moet je zelf schrijven en mag niet overgenomen zijn van het internet.
- Gebruik minimaal twee breakpoints, zodat je CSS mobiel-, tablet- en desktopschermen goed ondersteunt.
- Werk uitsluitend met de aangeleverde fonts. Het gebruik van Google Fonts of Typekit is dus *niet* toegestaan. Genereer online de noodzakelijke fontkits.
- Gebruik Unicode of Font Awsome voor alle iconen. Iconen hoeven niet exact overeen te komen met de prototypes, maar gebruik dan wel goede alternatieven.
- HTML en CSS-code moet voorzien zijn van voldoende en logisch commentaar.
- De webpagina moet goed functioneren op alle populaire desktop browsers én moderne iOS en Android devices.
- Verwerk ter verrijking een aantal CSS-transitions en / of animations, bijvoorbeeld voor de hero image teksten.
- Animaties moeten een toevoeging zijn voor de userexperience.
- Het toevoegen van animatie kan tijdrovend zijn. Doe dit pas op het allerlaatst!
- Gebruik alléén HTML en CSS technieken; voeg géén JavaScript toe.
- Er zijn twee belangrijke gedeeltes waarvoor je interactiviteit moet toevoegen: het navigatiemenu (in mobile view) en mouseover effecten in het gedeelte 'our latest bakery products'. Je voegt interactiviteit toe met behulp van pseudo-classes en/of pseudo-elements.

Inleveren van de opdracht

- De complete website moet worden aangeleverd in een folder genaamd 'www.' Deze folder stelt de root-directory voor die je toegewezen hebt gekregen door je Internet Service Provider (ISP).
- Zip het hele project en hernoem de bestandsnaam met daarin klas- en opdrachtcode en jouw naam.
- Stop het zip-bestand in de Drop Box van de docent.

Beoordelingscriteria

- HTML5
  - Nette, schone en overzichtelijke code
  - Semantiek en toegankelijkheid
  - Commentaar en correcte syntax
- CSS
  - Opbouw, indeling en structuur van de bestanden
  - Slim gebruik van selectors en pseudo classes
  - Gebruik van media queries en break points
  - Effectief gebruik van de cascade (geen overbodige properties)
  - Toepassing van RWD in de typografie
  - CSS-transitions en/of animations
  - Correcte syntax
  - Commentaar
- File management
  - Consequent upper-, lower- of camelcase, geldige bestandsnamen en correcte folderstructuur
- Algemeen
  - Oog voor detail
  - Op tijd ingeleverd, ontwikkeling van de student en invulling van de opdracht

Opdrachtomschrijving MM-4

Maak een mini-website over computertechnologie. De website bestaat uit een homepage en twee vervolgpagina's.

De twee vervolgpagina's bevatten informatie over computeronderdelen, zoals de harde schijf of het moederbord.

JS, GSAP, RWD

Tips en werkwijze

- Het doel is het maken van een creatieve e-learning presentatie. Haal alles uit de kast en denk ook aan minder voor de handliggende events (zoals scroll of touch) en het gebruik van diverse animaties met GSAP.
- Eerst denken, dan doen! Werk je idee uit met rapid prototype technieken en / of een PSS of PSD schema.
- Haal inspiratie uit de lesson files en de voorbeelden uit je boek.
- Verzin creatieve GUI elementen zoals een tabsysteem, slider, scroller, carousel, gallery, pop-up (modal window), accordion of formulierelementen.
- Doe zo veel mogelijk met JavaScript en zo min mogelijk met CSS. Natuurlijk hou je het praktisch; de website moet tenslotte ook responsive zijn.
- Op de homepage kan de gebruiker middels jouw creatieve GUI kiezen uit twee onderdelen (weblinks naar de vervolgpagina's).
- Op de vervolgpagina presenteer je een onderwerp. De gebruiker kan middels jouw creatieve GUI meer informatie opvragen over een specifiek onderdeel van dat onderwerp.
- De gegevens van die specifieke onderdelen komen uit JavaScript objecten.
- Afbeeldingen moeten uiteraard worden gepreload.
- JavaScript moet extern worden ingeladen.
- Alle code moet door jou zijn geschreven. Het is dus niet toegestaan om frameworks & libaries te gebruiken, behalve GSAP!
- Gebruik voor deze opdracht géén jQuery!
- Zorg dat je code modulair en overzichtelijk is opgesteld. Vergeet niet je code van voldoende commentaar te voorzien. Zo maak je het jezelf makkelijk als je code wilt hergebruiken voor je MM-B eindopdracht.
- De GUI bevat op z'n minst een vrijstaand gemaakte afbeelding voorzien van een responsive image map. Je mag ook werken met meerdere afbeeldingen i.v.m. hotspots en events. Maak vervolgens hotspots, zodat (gedeeltes van) afbeeldingen en / of teksten interactief worden. Afhankelijk waar de gebruiker op klikt of het scherm aanraakt, verschijnt er bijvoorbeeld een pop-op met tekst en eventueel ook nieuwe graphics.

Inleveren van de opdracht

- De complete website moet worden aangeleverd in een folder genaamd 'www.' Deze folder stelt de root-directory voor die je toegewezen hebt gekregen door je Internet Service Provider (ISP).
- Zip het hele project en hernoem de bestandsnaam met daarin klas- en opdrachtcode en jouw naam.
- Stop het zip-bestand in de Drop Box van de docent.

Beoordelingscriteria

- HTML5
  - Correcte syntax, semantiek, toegankelijkheid en commentaar
- CSS
  - Responsive Web Design
  - Correcte syntax, nette opbouw, indeling & structuur, effectief gebruik van de cascade (geen overbodige properties), slim gebruik van selectors & pseudo classes en commentaar
- JavaScript
  - Correcte syntax
  - Commentaar
  - Nette opbouw, indeling en structuur
  - Gebruik van variabelen
  - Gebruik van functies
  - Gebruik van objecten
  - Gebruik van events (óók werkend op touch devices)
  - Manipulatie van de DOM / BOM
  - Cross browser compatibility ?
- File management
  - Consequent upper-, lower- of camelcase, geldige bestandsnamen, correcte folderstructuur en folderbeveiliging
- Algemeen
  - Kwaliteit webdesign / interface design / look & feel / creativiteit JavaScript toepassingen
  - Graceful degradation / progressive enhancement
  - Oog voor detail
  - Op tijd ingeleverd, ontwikkeling van de student en invulling van de opdracht

zie correct pagina 21 onderaan Image maps

Wanneer je over een deel van de harddisk of het moederbord rolt met de muis, verschijnt er een pop-up met tekst en uitleg hoe dat betreffende onderdeel van de harddisk of moederbord werkt.

Opdrachtomschrijving HE-3

Maak een trailerwebsite over een populaire film. De website bevat een homepage en drie vervolgpagina's. Op de homepage kan de gebruiker kiezen uit een kleine, medium of een large variant van de trailer.

De videobestanden moeten voldoen aan de volgende technische specificaties:

bitsnelheid? VBR 1 & 2

- Groot: max. 720p en max. 5000 kbit/sec incl. audio
- Medium: max. 480p en max. 2500 kbit/sec incl. audio
- Klein: max. 360p en max. 1000 kbit/sec incl. audio

Tips en werkwijze

- Rip een video van YouTube met een zo hoog mogelijke kwaliteit. Gebruik als bronmateriaal tenminste 1080p, maar 4K is natuurlijk beter! (garbage in = garbage out)
- Comprimeer de video met Adobe Media Encoder of met Apple Compressor.
- Maak voor de klein, medium en large varianten tenminste twee verschillende bestanden (mp4 en webm).
- Het maken van een ogg kan meer punten opleveren.
- Je maakt dus minimaal zes videobestanden (en maximaal negen).
- Als je de video's comprimeert met VBR moet je goed opletten dat de gemiddelde bitrate wel binnen de gestelde specificaties valt. Natuurlijk is het niet erg als je met VBR zo nu en dan boven de gestelde aanleverspecificaties uitschiet.
- Controleer de door jouw gemaakte mp4-bestanden met de QuickTime of VLC player), zodat je zeker weet dat de video's voldoen aan de aanleverspecificaties.
- Gebruik een JavaScript video libary, zoals video.js of mediaelements.js om de videoplayer een 'unified look' te geven. Je hoeft de player niet zelf te skinnen.
- Gebruik HTML5 en CSS3 technieken. Het toepassen van RWD is niet nodig. Werk bijvoorbeeld een design uit voor een desktopresolutie van 1200 pixels breed.
- Maak een indrukwekkende UX door slim om te gaan met de postermovies. Je kunt zo een design maken waarbij de postermovie naadloos opgaat in de grapics er omheen.

	Res
klein	360p, 1000 kbit/s
medium	480p, 2500 kb/s
large	720p, 5000 kbit/s

output:
mp4
webm
ogg

Inleveren van de opdracht

- Lever geen bronbestanden aan!
- De complete website moet worden aangeleverd in een folder genaamd 'www.' Deze folder stelt de root-directory voor die je toegewezen hebt gekregen door je Internet Service Provider (ISP).
- Zip het hele project en hernoem de bestandsnaam met daarin klas- en opdrachtcode en jouw naam.
- Stop het zip-bestand in de Drop Box van de docent.

Beoordelingscriteria

- HTML5 & CSS3
  - Creativiteit en professionaliteit van het webdesign en creatief gebruik van de postermovie
  - HTML syntax, commentaar, semantiek en toegankelijkheid, netheid code.
  - CSS syntax, commentaar, nette opbouw, indeling & structuur.
- HTML5 Video Deployment
  - Geproduceerde video's voldoen aan de technische specificaties (minimaal mp4 en webm)
  - Implementatie van de video-tag
  - Implementatie en gebruik van een JavaScript video library.

– video.js
– mediaelements.js

volgende
zie p 19, onderaan
1) MP4

nieuw snelter

w x H		kbit/s
1280	720	5000
853	480	2500
640	360	1000

bij CRF: CBR, VBR1, VBR2
Bitrate schuiven

Opdrachtomschrijving HE-4

Maak voor moederbordfabrikant ASRock een aantrekkelijke interactieve HTML5 canvas banner. Werk je project uit met Adobe Animate en gebruik het GreenSock Animation Platform.

- HTML 5 canvas
- Animate
- GSAP

Tips en werkwijze

- Analyseer eerst 'voorbeeld.jpg' en het aangeleverde materiaal. In de library van companyLogos.fla staan alvast een aantal symbols voor je klaar.
- Eerst denken, dan doen! Plan je animatie door het maken van een digitale of papieren storyboard.
- Maak in Adobe Animate een nieuw HTML5 canvas project. De stage wordt 1920 X 750 pixels @ 30fps.
- Importeer rechtstreeks in Animate een CDN link van de GSAP library. Uiteraard moet de createJS library ook middels een CDN zijn gehost.
- Bewerk de gepubliceerde HTML-file van Animate en het project moet responsive zijn uitgevoerd.
- Voeg voldoende HTML en CSS toe en test je project op diverse browsers en devices.
- Externalize de javaScript die Animate produceert. In de HTML staat dus géén embedded JavaScript.
- Maak een aantrekkelijke animatie met complexe elementen. Denk aan nested timelines, stagger cycles, diverse keyframe interpolaties, masking technieken, morphs, frame-by-frame animaties, etc.
- Je mag traditionele animatie technieken gebruiken en de GSAP library. Het gebruik van GSAP levert meer punten op (zie beoordelingscriteria).
- Voeg interactieve elementen toe. Bijvoorbeeld wanneer je onderin op één van de vier logo's klikt dat er een tekstbalonnetje met meer informatie in beeld verschijnt.
- Let op de filesize van je project! Je mag niet over de 2MB heen.
- Wees creatief! Het project inleveren zoals het in de les is gemaakt levert geen maximale punten op.

Inleveren van de opdracht

- Het project moet worden aangeleverd in een folder genaamd 'www.' Deze folder stelt de root-directory voor die je toegewezen hebt gekregen door je Internet Service Provider (ISP).
- Zip het hele project en hernoem de bestandsnaam met daarin klas- en opdrachtcode en jouw naam.
- Stop het zip-bestand in de Drop Box van de docent.

Beoordelingscriteria

- HTML5 & CSS3
  - Commentaar, overzichtelijke code, correcte syntax, semantiek en toegankelijkheid, embedding van het canvasproject in de HTML en RWD
- Animate
  - Gebruik van symbols en instances
  - Organisatie van de timelines(s)
  - Gebruik van de GSAP library
  - Interactiviteit en het gebruik van de code snippets
  - Organisatie van de library en de naamgeving van objecten
  - Correcte en optimale exportsettings. Optimale filesize van het project
- Algemeen
  - File management, bestandsnamen en folderstructuur
  - Creativiteit van de animatie en eigen invulling van de opdracht
  - Oog voor detail

- optimaliseren bestandsgrootte
<500 kb banner

*web~zine

MULTIMEDIA WEB MAGAZINE FOR THOSE IN THE KNOW

hi, we are michiel & sebastian - technology aficionados. we bring you the latest news about what's hot & what's not in the multimedia world.

Opdrachtomschrijving WM-2

NPM
SCSS

Maak in groepsverband een online magazine in de vorm van een e-mailnieuwsbrief. Het magazine moet de volgende onderdelen bevatten:

- Een review (met een duidelijke mening) van hardware, software of van een website.
- Een column met een mening over een in de multimediawereld actueel en relevant onderwerp (zoals 360 graden video, apps, rechtzaken tegen bedrijven als Microsoft, privacy op het internet, de toekomst van het internet, social media, cryptocurrency, etc).
- Een top vijf van mooi ontworpen, nuttige of grappige sites met een hele korte omschrijving en links naar deze sites.

Tips en werkwijze

- Schrijf je eigen materiaal. Je mag geen teksten van andere websites kopiëren. Plagiaat plegen betekent dat je voor deze opdracht bent gediskwalificeerd.
- Plaats bij alle onderdelen relevant beeldmateriaal, zoals foto's, iconen en andere keyvisuals. Let wel op auteursrechten.
- Graphics mogen niet van Google Images afkomstig zijn, maar wel van (gratis) stockfoto websites.
- Gebruik de Node Package Manager om je project te beheren en voor het installeren van dependencies.
- Gebruik de HTML5 syntax.
- De CSS moet gecompileerd zijn aan de hand van een SCSS bestand. Gebruik minimaal twee partials.
- Maak één webpagina die volledige responsive is uitgevoerd.
- Gebruik minimaal twee breakpoints en zorg ervoor dat de webmagazine ook op het kleinste scherm goed eruit komt te zien.
- Gebruik minimaal één jQuery plug-in of framework, zoals aniview.js.
- Zorg voor een correcte mappen- en filestructuur.

De taakverdeling

Node JS

Je mag zelf een team vormen. De docent geeft aan uit hoeveel leden een team bestaat.

Stel iemand aan als projectmanager. Hij of zij bewaakt de deadlines en de voortgang van het project (en werkt natuurlijk ook zelf mee aan het product).

Inleveren van de opdracht

- Lever de node_modules map ***niet*** aan! Wel moet de package.json aanwezig zijn.
- Vergeet niet de Photoshop, Illustrator en Experience Design documenten aan te leveren in de _source map).
- Zip het hele project en hernoem de bestandsnaam met daarin klas- en opdrachtcode en jullie voornamen.
- Stop het zip-bestand in de Drop Box van de docent.

Beoordelingscriteria

- HTML5
  - HTML syntax, commentaar, semantiek en toegankelijkheid, netheid code.
- SCSS
  - Responsive Web Design
  - Correcte syntax, nette opbouw, indeling & structuur, effectief gebruik van de cascade (geen overbodige properties), slim gebruik van selectors & pseudo classes en commentaar.
- Graphics
  - Goede compressie en kwaliteit van graphics en gebruik van Photoshop / Illustrator / Experience Design Assets generators
- File management
  - Consequent upper-, lower- of camelcase en geldige bestandsnamen / correcte folderstructuur
- Algemeen
  - Kwaliteit en informatieve waarde van content en alle verplichte onderdelen aanwezig
  - Design
  - Gebruik van NodeJS & NPM
  - Gebruik van jQuery plug-ins of frameworks

CODEC Overzicht

CODEC	JAAR	OMSCHRIJVING
Cinepak	1991	Een relatief eenvoudige CODEC met lage beeldkwaliteit. Tijdens het decoden is er weinig processorkracht nodig. Het is geoptimaliseerd voor een 1x speed CD-ROM speler (150 KBit/s) en een resolutie van 320 x 240. De CODEC werd geleverd bij QuickTime en Windows Media Player en was populair totdat in 1997 Sorenson werd geïntroduceerd.
MPEG-1	1993	MPEG-1 is geoptimaliseerd voor CD-ROM en VideoCD toepassingen (voorloper van de DVD) en ondersteunt maximaal 100 MBit/s.
MPEG-2	1996	Ook wel H.222 / H.262 genoemd. MPEG-2 is niet geschikt voor bitrates onder 1000 KBit/s en daarom niet geschikt voor het web. MPEG-2 wordt gebruikt voor DVD's, lage kwaliteit Blu-Ray en diverse streaming toepassing (inclusief KPN en Ziggo).
Sorenson	1997	De CODEC presteert iets beter dan Cinepak en is later sterk verbeterd (Sorenson 2 en 3). De CODEC was erg populair bij Apple en werd standaard geleverd bij QuickTime 3 (1998).
MPEG-4/2	1999	MPEG-4 Part 2 is ongeveer 50% efficiënter dan MPEG-2. Dit betekent de helft van de bestandsgrootte met vergelijkbare beeldkwaliteit! Hierdoor was MPEG-4/2 populair op file sharing platforms zoals KaZaA, IMesh, Gnutella en LimeWire. MPEG-4/2 was op deze platformen beter bekent onder de CODEC implementaties van Xvid, DivX en 3ivw.
Sorenson Spark	2002	Sorenson Spark is een implementatie van H.263 voor Flash Video. Na de lancering van MPEG-4 werd snel duidelijk dat Apple dit formaat omarmde en wegstapte van Sorenson CODECS. Sorenson was genoodzaakt hun CODEC elders te verkopen. Uiteindelijk was het Marcomedia (overgenomen door Adobe) die in 2002 Sorenson Spark omarmde voor hun Macromedia Flash MX software.
H.264 MP4	2003	H.264 wordt ook Advanced Video Coding (AVC) en MPEG-4 Part 10 genoemd. Het is ontwikkeld voor Blu-Ray Disc, smartphone en streamingtoepassingen en ondersteunt resoluties tot 4K. H.264 is de meest gebruikte CODEC voor video deployment op het web en wordt gebruikt in de MP4 video container. Let op: verwar de MP4 container niet met de MPEG4/2 CODEC).
VP6 en VP7	2003	De VP6 CODEC is ontwikkeld door On2 Technologies. Vanaf 2005 werd VP6 gebruikt door de Macromedia Flash Player als opvolger van de Sorenson Spark CODEC.
Theora – ogg	2004	Theora is ontwikkeld door de Xiph.Org Foundation en is (bijna) patentvrij en royalty-free. De CODEC wordt gebruikt in de Ogg en Matroska video container.
VP8 en VP9 webM	2008	In 2010 werd On2 Technologies overgenomen door Google. VP8 en VP9 CODECS (2013) worden het meest gebruikt in de videocontainer WebM van Google (gebaseerd op Matroska). De CODECS zijn bedoeld als royalty-free alternatief voor H.264. Ze zijn na H.264 het meest gebruikt voor video deployment op het web.
H.265	2012	H.265 wordt ook High Efficiency Video Coding (HEVC) en MPEG-H Part 2 genoemd. Het is ongeveer 20% efficiënter dan H.264 en ondersteunt resoluties tot 8K. Het is (nog) niet geschikt voor webdeployment, omdat niet alle browsers het ondersteunen.

H.264 MPEG-4/AVC

webm

Geschiedenis

Vóór de komst van HTML5 (oktober 2014) had je een third-party plug-in nodig om in de browser videocontent te kunnen afspelen. Een plug-in is add-on software waarmee de browser meer mogelijkheden krijgt.

Plug-in- en browserproducenten konden het niet eens worden over een video en audio standaard. Er waren verschillende plug-ins beschikbaar die je apart van elkaar moest installeren:

- Flash
- Silverlight
- QuickTime
- RealPlayer
- Windows Media Player

De ondergang van Flash video

Het is nu wellicht moeilijk voor te stellen, maar Flash van Macromedia (later Adobe) was lange tijd dè defacto standaard voor het afspelen van interactieve multimedia, games, animaties, video en audio in de browser.

In 2007 lanceerde Apple een smartphone die de markt compleet op de kop zette: de Iphone. Tot grote verbazing van de industrie weigerde Apple om de populaire Adobe Flash plug-in toe te laten, omdat de plug-in onveilig en ongebruiksvriendelijk zou zijn. Daarnaast zou Flash te veel processorkracht, geheugen, schijfruimte en batterij vragen.

De iPhone werd in eerste instantie belachelijk gemaakt omdat het geen Flash ondersteunde. Door de weergaloze populariteit van de smartphone stond de wereld al snel op z'n kop. Het bleek juist een nadeel dat Flashwebsites niet op de iPhone draaiden!

Adobe heeft in dezelfde periode (2007-2014) erg veel negatieve publiciteit gehad door de vele veiligheidslekken die werden ontdekt in de Flash plug-in. De reputatie van Flash heeft hieronder sterk geleden.

De laatste nagel aan de doodskist van Flash was de ontwikkeling van HTML5. HTML5 biedt open source en licentievrije video deployment oplossingen.

Voor YouTube stapte Google in 20015 af van Flash en ging over op HTML5 technieken.

* Adobe Media encoder
* Apple compressor
* video DVD ripper

Video deployment op het web

Sinds de komst van HTML5 kunnen webbrowsers video en audio afspelen zonder plug-in. De browser beschikt zelf over de noodzakelijke CODECS.

Helaas is er niet één formaat, maar drie: WebM, MP4 en OGG. Op dit moment wordt alleen MP4 door alle browsers ondersteund. Om het grootste deel van de browsers te ondersteunen heb je tenminste een MP4 en WebM video nodig.

WebM

VP8/VP9 + Vorbis

Google heeft zijn eigen containerformaat, WebM (.webm), afgeleid van de open source Matroska container (.mkv). Een WebM-container bevat een VP8 of VP9 video track en een Vorbis audio track.

Vorbis is een open source en royalty free audioformaat van Xiph, als antwoord op het MP3-formaat van het Fraunhofer instituut, dat in 1998 geld wilde zien voor haar patenten.

Vorbis audio wordt vaak omsloten met een Ogg container. De CODEC wordt daarom vaak OggVorbis genoemd, maar wordt ook gebruikt in WebM.

Android Phones kunnen Vorbis audio afspelen zonder extra software. Dit noemen we native ondersteuning.

Opera, FireFox, Chrome, Chromium en Edge ondersteunen allemaal WebM. Alleen Internet Explorer heeft een aparte plug-in nodig.

H264 + AAC video.js

MP4

Alle moderne webbrowsers kunnen omgaan met de MP4 container (niet te verwarren met de MPEG-4 *CODEC*). De MP4 container is gebaseerd op het QuickTime formaat van Apple.

De container bevat een H.264 video track en een AAC audio track. Deze twee compressiemethodes zijn uitstekend geschikt voor lage bandbreedtes en lage bitrates. Ze stellen weinig eisen aan moderne CPU's en dus perfect geschikt voor mobiele apparaten.

webm. (plugin) (Firefoxware.com)

OGG (optional) - MP4 & more

AAC staat voor Advanced Audio Coding en is de opvolger van de MP3 (MPEG-1 / MPEG-2 Layer 3). De geluidskwaliteit van AAC is veel beter dan het sterk verouderde MP3.

Om verwarring te voorkomen gebruikt Apple de .m4a extensie (in plaats van .mp4) voor hun audiobestanden. Toch is het gewoon een MP4 container.

In 2011 gaf Google aan H.264 support in Chrome te willen schrappen. Tot op heden hebben ze dit nog niet gedaan, omdat YouTube leunt op de MP4 container met de H.264 CODEC. In 2017 is gebleken dat hier wellicht nog verandering in komt.

OGG — Theora + Vorbis

OGG is een patentvrij open source containerformaat. Het is daarom door het W3C in 2007 aangemerkt als webvideo standaard. FireFox biedt native support. Men gebruikt soms de extensie .ogv en .oga in plaats van .ogg om onderscheid te maken tussen video- en audiobestanden.

Het videocomponent is de Theora CODEC, die minder efficiënt is dan VP8 en H.264. De beeldkwaliteit van Theora is dus minder goed. Theora kan omsloten worden door een WebM-, OGG- MP4- of MKV-container.

Bij nader inzien bleek Theora toch niet geheel patent vrij. Apple heeft aangegeven dat dit in de toekomst voor problemen kan zorgen. Microsoft, Google en Apple volgen daarom de W3C recommendation niet.

Het audiocomponent van OGG is net als bij WebM de Vorbis CODEC. .ogv, .oga, .ogg, Theora & Vorbis

Digital Right Management

De platen- en filmbazen willen dat het W3C DRM ontwikkelt, zodat videomateriaal door de gebruiker niet gekopieerd en verspreid kan worden.

Browser	MP4	WebM	Ogg
IE	Ja	Nee, aparte plug-in noodzakelijk.	Nee
Edge	Ja	Ja	Nee
Chrome	Ja	Ja	Ja
FireFox	Ja	Ja	Ja
Safari	Ja	Nee	Nee
Opera	Ja (vanaf versie 25)	Ja	Ja

DRM gaat in tegen het principe van een open en vrij internet. Het is daarom ook niet vreemd dat er binnen het W3C veel weerstand is tegen dit voorstel.

MIME-types

MIME staat voor Multipurpose Internet Mail Extensions en is oorspronkelijk bedoeld om niet-tekstbestanden via e-mail te kunnen opsturen. De MIME-type 'overruled' de bestandsextensie en bestaat uit twee delen: het hoofd- en subtype. Bijvoorbeeld: video/webm, multipart/form-data, text/html of image/png. Ze zijn oud, maar worden nog steeds toegepast. Voorbeelden:

```
<link href="styles.css" type="text/css" />
<source src="../media/ywcc.mp4" type="video/mp4">
```

Het is cruciaal dat source-tags in de video-tag de juiste MIME-types meekrijgen.

Video-tag

In 2007 kwam Opera met de enkel- en meervoudige video-tag. Het is tegenwoordig een W3C standaard.

Sommige oudere browsers tonen de video-tag niet goed zonder width en height attributes.

Source-tag

De video-tag wordt meestal meervoudig uitgevoerd en bevat dan meerdere enkelvoudige source-tags voor het aanbieden van meerdere videobronnen. De eerste source-tag moet altijd MP4 zijn, in verband met iOS ondersteuning.

Als je één bron wilt aanbieden kun je de video-tag enkelvoudig uitvoeren in combinatie met een src-attribute.

Track-tag

Met de enkelvoudige track-tag kan o.a. ondertiteling worden toegevoegd. Helaas werkt dit nog niet goed op alle browsers.

Als je IE9 wilt ondersteunen moet je de tag meervoudig maken. Dit is tegen de W3C regels en geeft uiteraard validation errors! We kunnen hiervoor JavaScript gebruiken.

magic number?

Legenda

(5)	HTML5	Het attribute is geldig in HTML-5.
|	óf	Er zijn meerdere mogelijkheden, gescheiden door het pipe-teken, "|".
="number"	verwachte waarde	Een omschrijving van de verwachte waarde voor het attribute. tabindex="number" betekent dat er een nummer verwacht wordt, zoals bijvoorbeeld tabindex="5".
[BOOL]	boolean	Boolean attributes.

Nota Bene!

W3Schools.com bevat een compleet overzicht, tutorials én praktische (interactieve) voorbeelden.

Hoofdwerk

Video- en Audio-tags

(5) <video> enkelvoudig | meervoudig	- HTML5 videoplayer element.
(5) <audio> enkelvoudig | meervoudig	- HTML5 audioplayer element.
(5) <source> enkelvoudig	- bronnen voor de audio-, video- of picture-tag.
(5) <track> enkelvoudig	- text tracks voor media elementen (zoals omschrijvingen, chapter markers en ondertiteling).

Atrributes Video-tag

(5) width="number" | height="number"	- breedte of hoogte van de video.
(5) controls [BOOL]	- een videocontroller laten zien. Elke browser heeft een unieke layout.
(5) autoplay [BOOL]	- direct afspelen zodra er voldoende gegevens zijn gedownload.
(5) loop [BOOL]	- herhalend afspelen.
(5) muted [BOOL]	- audio stil zetten.
(5) poster="URL"	- een afbeelding laten zien terwijl de video preload of totdat de gebruiker de video start.
(5) preload="auto|meta|none"	- auto begint direct de video te downloaden. 'Meta' download alleen metadata, maar geen videogegevens. 'None' download geen gegevens, totdat de gebruiker de video start.

Source-tag

(5) src="URL"	- verwijzing naar het mediabestand voor audio- en video-tags.
(5) srcset="URL"	- verwijzing naar afbeeldingen voor de picture-tag.
(5) media="media query"	- een afbeelding voor de picture-tag voor een specifieke media query.
(5) sizes="image sizes"	- afmetingen opgeven voor een afbeelding voor specifieke lay-outs
(5) type="type/subtype"	- de MIME-type opgeven van een bron.

Voorbeeld

toets

```
<video width="960" height="540" poster="../media/ywcc.jpg" controls muted loop preload="auto">
    <source src="../media/ywcc.mp4" type="video/mp4">
    <source src="../media/ywcc.webm" type="video/webm">
    <source src="../media/ywcc.ogg" type="video/ogg">

    <track src="subtitles-en.vtt" kind="subtitles" srclang="en" label="English (British)" default>
    <track src="subtitles-nl.vtt" kind="subtitles" srclang="nl" label="Nederlands">

    <p>Uw browser heeft geen video ondersteuning. U kunt de <a href="../media/ywcc.mp4" download>Yes We
    Can Clinics promo</a> downloaden en bekijken op uw computer middels andere videosoftware.
    Wij raden <a href="http://www.videolan.org/">de gratis VLC-speler</a> aan.</p>
</video>
```

Application Programming Interface

Een API biedt (beperkte) toegang tot het programma van een derde partij. Het is een set afspraken om informatie of instructies tussen de twee computerprogramma's uit te wisselen. De communicatie verloopt meestal via JSON (JavaScript Object Notation) of XML (eXtensible Markup Language).

Als je een API gebruikt hoef je alleen te weten welke gegevens moeten worden opgestuurd en/of welke gegevens worden teruggestuurd. Je hoeft niet te weten hoe die gegevens tot stand komen of hoe het bij de derde partij is geprogrammeerd. De derde partij levert jou natuurlijk documentatie aan, zodat jij de API kunt gebruiken in jouw project.

Veel websites en webapplicaties hebben tegenwoordig een API, gericht op het aan derden beschikbaar stellen van functies of informatie.

Open Graph Protocol

Het Open Graph Protocol is ontwikkeld door Facebook. Het procotol biedt webdevelopers één universele manier om informatie van webpagina's beschikbaar te stellen aan social media platformen. Het werkt met een meta-tag die door social media platformen uitgelezen kan worden:

```
<meta property="og:property" content="value">
```

De image, title, description en url staan onder het bericht van de Facebookgebruiker.

Als je een share of like button van Facebook in jouw webpagina wilt zetten dan moet je (behalve enkele platformspecifieke tags) tenminste meta-tags schrijven voor:

`og:title`	- de titel van het te delen object.
`og:type`	- het type object (bijvoorbeeld movie, blog, article, news, website)
`og:site_name`	- de naam van het te delen object of van de site (gebruik *géén* URL).
`og:description`	- een omschrijving van het te delen object.
`og:image`	- een afbeelding (gebruik een 1.91:1 aspect ratio en minmaal 1200X630 pixels).
`og:url`	- de URL van de te delen webpagina. Gebruik een absolute URL die gelijk is aan de te delen webpagina.

iFrame-tag

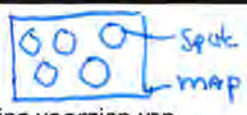
Een iframe-tag is een venster (frame) in de webpagina, waarin een andere webpagina wordt geladen. Het inladen van Google Maps of een YouTube video zijn veel gebruikte voorbeelden.

In HTML5 heb je de mogelijkheid om de iframe te voorzien van height-, width-, name-, src- en sandbox-attributes.

De sandbox-attribute beperkt de werking van het iframe. Soms wil je een webpagina inladen van een ander domein, maar dat wil niet zeggen dat die webpagina te vertrouwen is. Met sandbox kunt je de werking van bijvoorbeeld formulieren, script of pop-ups blokkeren.

sandbox attribute Iframe

Image maps



Met Image maps kun je een afbeelding voorzien van meerdere hotspots. Er zijn twee soorten image maps: server-side en client-side.

De server-side image map is een oude techniek, voor het eerst geïntroduceerd in 1993 met de komst van de Mosaic webbrowser.

De werking is simpel: een img-tag wordt voorzien van het ismap-attribute. Vervolgens krijgt de img-tag een parent a-tag met hyperlink reference naar een script die het klik-event verder afhandelt:

```
<a href="scripts/map-processor.asp">
      <img src="landkaart.gif" alt="Kaart van
      Nederland" width="800" height="600" ismap>
</a>
```

De browser herkent het ismap-attribute. Wanneer de gebruiker op de afbeelding klikt zal de browser X- en Y-coördinaten van de muisklikpositie toevoegen aan de URL (relatief gemeten vanaf de linker bovenhoek van de afbeelding). Dit is vergelijkbaar met de get-methode van de form-tag:

```
scripts/map-processor.asp?13,90
```

Het serverscript zorgt vervolgens voor de afhandeling.

Client-side Image Maps

Eind 1995 deed de client-side image map zijn intrede (Mosaic versie 2.1.0). In tegenstelling tot de server-side image map zijn hotspotcoördinaten niet op de server opgeslagen, maar client-side in de HTML.

De image-tag wordt voorzien van het usemap-attribute met als value een ID-verwijzing naar een map-tag. De map-tag bevat area-tags die de hotspotcoördinaten bevatten:

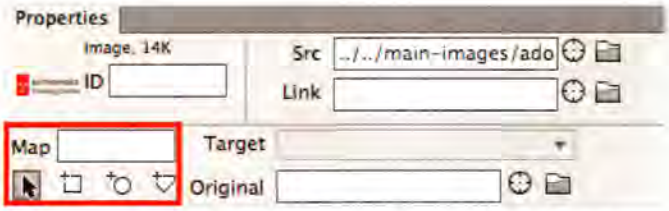
tool

```
<img src="landkaart.gif" alt="Kaart van Nederland"
width="800" height="600" usemap="#kaart-nederland">
<map name="kaart-nederland" id="kaart-nederland">
      <area shape="rect" coords="41,44,341,346"
      alt="Provincie Utrecht" href="utrecht.htm">
</map>
```

De hotspot in dit voorbeeld is een vierkant (shape="rect"). Er zijn slechts twee punten nodig om een vierkant te tekenen. De linker bovenhoek (pixelcoördinaat 41,44) en de rechter onderhoek (pixelcoördinaat 341,346).

Je kunt in Photoshop het info panel gebruiken om de pixelcoördinaten te achterhalen. Bij complexe vormen is dat niet praktisch, omdat de coords-attributes dan erg lang en ingewikkeld worden.

Gebruik daarom speciale software waarmee je met WYSIWYG de image map kan opstellen, zoals Abobe Dreamweaver.



De imag map tool in de Properties panel van Dreamweaver.

Image maps zijn NIET responsive!

Image maps zijn oude HTML technieken gebaseerd op pixels. Ze werden met name veel gebruikt vòòrdat Reponsive Web Design (RWD) zijn intrede deed. Zonder extra hulp werkt de image map dus ook niet correct in een responsive omgeving.

De jQuery plug-in van Matt Stow biedt de oplossing.

Image maps gebruiken tijdens de opleiding

Gebruik image map technieken voor je MM-4 JavaScript opdracht om één enkele afbeelding te voorzien met meerdere JavaScript events.

Map-tag

```
<map id="mapNaam" name="mapNaam">
```

- Meervoudige tag met daarin minimaal één area-tag.

Area-tag

```
<area shape="rect|circle|poly|"
coords="pixelcoördinaten" href="link.html"
alt="alt-text">
```

- Enkelvoudige tag. Rectangles hebben twee punten (X1,Y1,X2,Y2). Circles hebben drie punten (X1,Y1 en de straal van de cirkel). Polyvormen mogen zo veel punten bevatten als je wilt. De alt-attribute mag nooit ontbreken (anders heb je een lege linktekst).

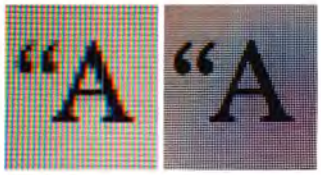
Img-tag

Extra attributen voor de img-tag : `usemap="#mapNaam"`. Gebruik `usemap="pages/map.htm#mapname"` om naar een map-tag van een extern webdocument te verwijzen (niet aangeraden).

Device pixel

Het woord pixel is een samenvoeging van de twee eerste lettergrepen van het Engelse 'picture element'. De device pixel, ook wel fysieke pixel of hardware pixel genoemd, is één enkel punt op het beeldscherm.

Vroeger hadden beeldschermen jaren lang dezelfde resolutie van ongeveer 72ppi tot 96ppi. Tegenwoordig krijgen beeldschermen steeds meer pixels per inch, want dat betekent meer detail en scherper beeld!



Device pixel vs Reference pixel

Het screenshot onderin deze pagina is gemaakt op een Apple MacBook Pro met een schermresolutie van 221ppi. Je ziet dezelfde afbeelding in twee verschillende programma's; Photoshop en Chrome. Beide programma's zijn ingesteld op 100% zoom, maar de afbeelding is niet even groot! ?

device pixel / Fysieke

Device pixel

Photoshop toont afbeeldingen aan de hand van hardware pixels. Daarom is in Photoshop een afbeelding van 1000 X 1000 pixels @ 300ppi op 100% zoom exact even groot als een afbeelding van 1000 X 1000 pixels @ 150ppi op 100% zoom. Pas bij het afdrukken zijn er verschillen zichtbaar (zie het kopje 'PPI en drukwerk' op de volgende pagina).

Pixel als relatieve eenheid

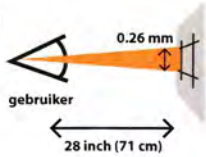
Pixels hebben geen vaste afmeting. Het is afhankelijk van het device. Maar, als devices door de jaren heen meer pixels per inch krijgen, waarom worden websites dan niet steeds kleiner afgebeeld?

Browsers kunnen dan ook niet uitgaan van de hardware pixel om op al die verschillende beeldschermen websites consistent te tonen!

Reference pixel (CSS Pixel)

Als we in CSS de pixel-unit gebruiken, gebruik je de zogenaamde 'reference pixel' (ook wel CSS pixel genoemd). De fysieke afmeting van deze puur hypothetische pixel is een W3C standaard.

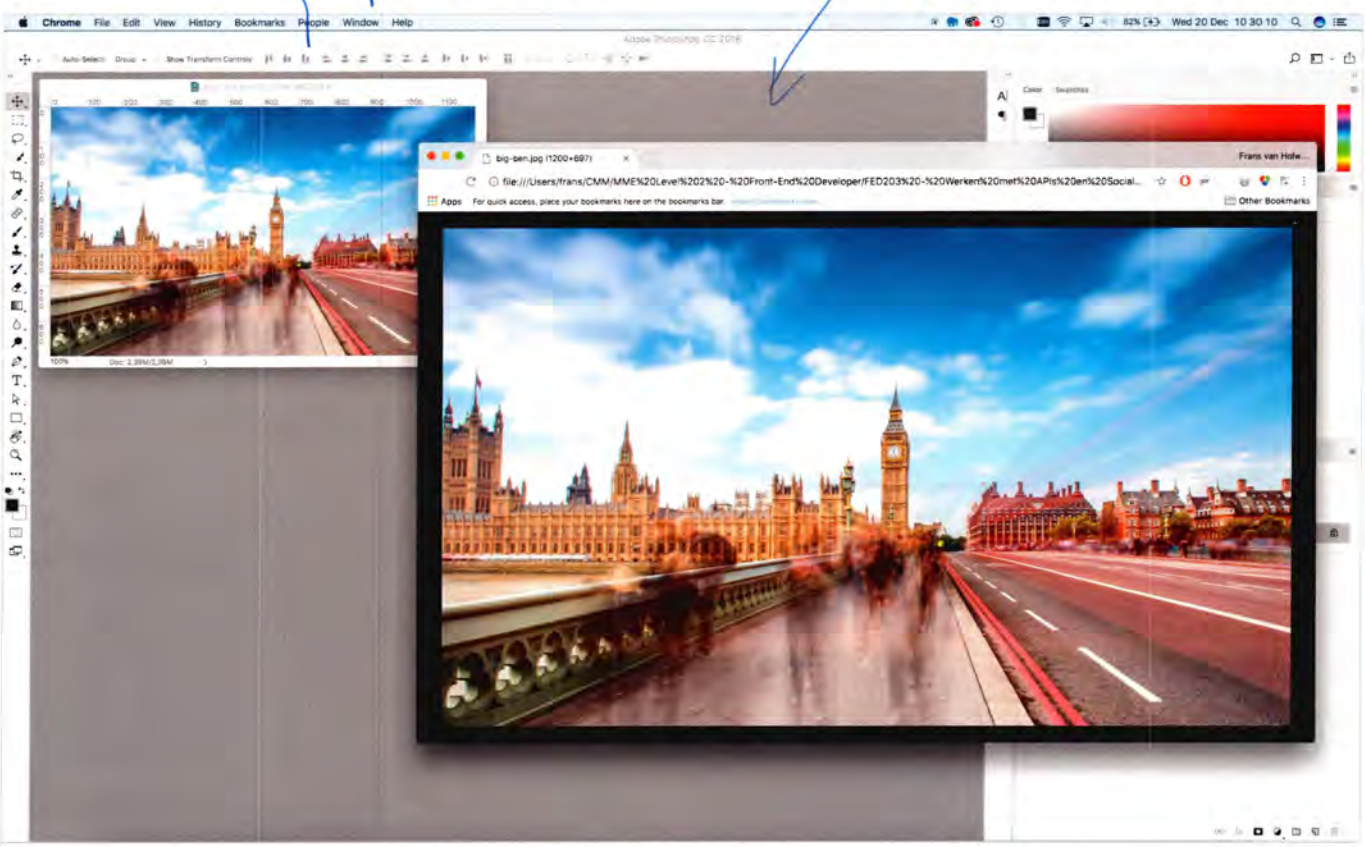
Het is de afmeting van één pixel, gemeten op een device van 96PPI, op armafstand (28") gehouden. De reference pixel is dan 1/96 inch of te wel 0.26 mm.



De reference pixel is een abstract concept, maar door de bepaling van het W3C optisch consistent gemaakt.

PS: Hardware pixel

Chrome reference pixel

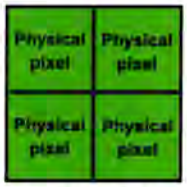


Device Pixel Aspect Ratio

Voor de browser zijn twee resoluties van belang: de device pixel resolutie en de CSS reference pixel.

De browser rekent uit hoeveel hardware pixels overeenkomen komen met de reference pixel. Deze verhouding noemen we de device pixel aspect ratio.

De iPhone 7 heeft in de breedte 750 hardware pixels en een pixel aspect ratio van 2. Dit betekent dat de breedte van het scherm overeenkomt met 375 CSS pixels. `width:375px;` komt daarom overeen met `width:100%;`.



4:1 of 2:1?

Een compleet en up-to-date overzicht van device pixel aspect ratio's vind je op mydevice.io/devices/

- **1** klassieke desktop computers
- **1,5** oudere of low-end Android telefoons
- **2** Android en Apple Retina schermen
- **3** iPhoneX, iPad Pro, HTC One, Samsung Galaxy S4
- **4** high-end device: Samsung Galaxy S8 en LG G5

Assets maken voor hi-res schermen

Hou rekening met de device aspect ratio, zodat graphics scherp worden getoond in browsers van hoge resolutie beeldschermen.

Stel, je wilt een afbeelding mooi scherp tonen. De afbeelding moet in de browser 300 x 300 pixels groot worden. In Photoshop maak je de volgende afbeeldingen:

- Normaal (1:1) 300 X 300 - image.jpg
- iPhone 7 (2:1) 600 X 600 - image@2x.jpg
- iPhone X (3:1) 900 X 900 - image@3x.jpg
- LG G5 (4:1) 1200 X 1200 - image@4x.jpg

Gebruik vervolgens de picture-tag en media queries om per device de juiste afbeelding te tonen.

Picture-tag voorbeeld

opdracht?

```
<picture>

<source media="(-webkit-min-device-pixel-ratio: 3), (-o-min-device-pixel-ratio: 3/1), (min-resolution: 3dppx)" srcset="image@3x.gif">

<source media="(-webkit-min-device-pixel-ratio: 2), (-o-min-device-pixel-ratio: 2/1), (min-resolution: 2dppx)" srcset="image@2x.gif">

<img src="image.gif" alt="Default graphic, ook voor browsers die de picture-tag niet ondersteunen.">

</picture>
```

CSS Media Query voorbeeld

?

```
#mediaQueries{
  background-image: url("image.gif");
}

@media (-webkit-min-device-pixel-ratio: 3), (-o-min-device-pixel-ratio: 3/1), (min-resolution: 3dppx){
  #mediaQueries{
    background-image: url("image@3x.gif");
  }
}

@media (-webkit-min-device-pixel-ratio: 2), (-o-min-device-pixel-ratio: 2/1), (min-resolution: 2dppx){
  #mediaQueries{
    background-image: url("image@2x.gif");
  }
}
```

PPI en drukwerk

Om graphics op papier scherp te kunnen afdrukken, kijken we naar de resolutie van de afbeelding; het aantal pixels per inch (of centimeter).

In drukwerk is een graphic van tien bij tien centimeter met 300ppi twee keer zo gedetaileerd (en dus scherper!) dan een graphic van tien bij tien centimeter met 150ppi.

Je kunt ook andersom denken, al is dat niet praktisch en erg verwarrend. Een graphic van 1000 X 1000 @ 300ppi wordt twee keer zo klein afgedrukt dan een graphic van 1000 X 1000 @ 150ppi.

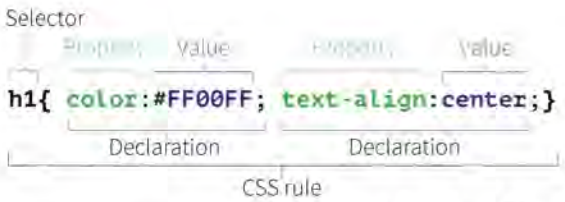
Conclusie

Bij het maken van webgraphics zorg je ervoor dat de graphic genoeg pixels heeft, zodat het op hi-res schermen scherp is. De pixeldichtheid (ppi) is voor websites niet relevant.

Maak graphics aan met 72ppi, 24 bitdiepte (8 bit per kanaal) en gebruik de sRGB color space.

CSS rule

Een CSS instructie noemen we een CSS-rule. CSS-rules bestaan uit selectors, declarations, properties en values.



Selector

les: [ATTR^$*]
+ , > (spc) ~ ! * ␣

De selector bepaalt welk HTML-element, of welke combinatie van HTML-elementen wordt aangesproken.

Declaration, Property en Value

Declarations komen na de selector(s) en staan tussen curly braces ({ }). Met declarations bepaal je *wat* (property) en *hoe* (value) HTML-elementen worden gestyled.

first-of-type

CSS: Theorieën

Cascading Style Sheets kent vier principes die bepalen hoe vormgeving van een HTML-element tot stand komt; de Cascade Order:

- Order & Proximity
- Priority
- Specificity
- Inheritance

les! → opbouw CSS rule.

Proximity

Met proximity bedoelen we de fysieke plek waar CSS-code voorkomt ten opzichte van de HTML. Van laag (als eerste toegepast en ver van het HTML-element vandaan) naar hoog (als laatst toegepast en dichtbij het HTML-element):

1. **De browser defaults** zijn de standaard stijlen die browsers (de user-agents) meegeven aan HTML-elementen. Zo heeft de body een marge, paragrafen zijn 16 pixels groot en van het Times lettertype. Kopteksten zijn vet gedrukt, hyperlinks zijn underlined en blauw en lijst items hebben bullets en zijn ingesprongen. Deze vormgeving is eenvoudig te overschrijven met jouw eigen stijlen.

   De W3C omschrijft hoe browsers HTML-elementen moeten vormgeven. Desondanks zijn er kleine verschillen (met name met Webkit- en iOS-browsers). Daarom is het belangrijk dat je jouw eigen stijlen niet baseert op de browser defaults en een 'css-reset' toe past (ook normalizer genoemd). Dit is een zelfgeschreven CSS-bestand waarmee je browser defaults opheft.

   Er zijn veel voorbeelden te vinden van goede CSS-resets, waarvan Eric Meyer's toepassing het meest wordt gebruikt.

2. **External & Embedded styles** zijn stijlen die je zelf toevoegt aan een webdocument. ~~Embedded styles staan tussen style-tags van de head-tag.~~ External styles worden gekoppeld aan de HTML met de link-tag.

   External styles hebben de voorkeur omdat de vormgeving voor de hele website op één centrale plek te beheren is.

```
<head>
   <title>Pagina titel</title>
   <link href="css/external-styles.css"
     rel="stylesheet">
   <style>
      /* Embedded styles */
   </style>
</head>
```

les

3. **De @import rule** koppelt een external stylesheet aan een ander stylesheet. CSS-code kan zo modulair worden onderverdeeld in verschillende bestanden, terwijl er in de head-tag van de HTML toch maar één CSS-bestand wordt gekoppeld. Complexe websites zijn op deze manier overzichtelijk en makkelijker te beheren.

```
// Onderstaande CSS-code plaats je bijvoorbeeld
in een CSS-bestand met de bestandsnaam main.css:

@import url('css-reset.css');
@import url('layout.css');
@import url('typography.css');
@import url('homepage.css');
@import url('ie-hacks.css');
```

   Toch gebruiken we de @import rule in de praktijk niet, omdat ze het laden van de webpagina aanzienlijk vertagen. Browsers kunnen gegevens parallel downloaden (meerdere bestanden tegelijk), maar bij de @import-rule gebeurt dat in serie (stapsgewijs na elkaar).

   Ook is deze methode niet compatible met oude webbrowsers en sommige Internet Explorer browsers hebben problemen met het lezen van media queries die in @import voorkomen.

   Vanwege de performance nadelen gebruiken we de @import-rule in de praktijk alleen voor uitzonderingen, bijvoorbeeld om Google fonts te koppelen.

les: google chrome inspector oefeng.
les: Font Squirrel

CSS, FONT SHORTHAND. CLEARFIX: OVERFLOW: hidden;

4. **@media rules** (media queries) worden gebruikt om CSS-rules selectief toe te passen voor bepaalde devices of schermgroottes (Responsive Web Design). Voorbeeld:

```
@media screen and (orientation:landscape){
    body{
      background-color:red;
    }
}
```

Hierboven staat: **als** het een screen device betreft **en** de schermoriëntatie **is** landscape, **dan** wordt de achtergrond van de body rood.

5. **Inline declaraties** staan in een style-attribute van een HTML-element. Inline styles bieden geen duurzame oplossing, omdat structuur (HTML) en vormgeving (CSS) strict gescheiden moet zijn. Gebruik inline styles alleen als het niet anders kan, zoals bijvoorbeeld bij e-mail nieuwsbrieven. Voorbeeld:

```
<p style="font-size:15px;">inhoud</p>
```

6. **User-stylesheets** zijn styleheets van de gebruiker zelf. Veel browsers bieden de mogelijkheid om eigen stylesheets te maken. Mensen met een zichtbeperking (slechte ogen, kleurenblindheid, etc) maken hier gebruik van.

De user-stylesheet heeft altijd de hoogst mogelijke positie binnen de cascade order. Ondanks dat deze stijlinformatie fysiek gezien niet het vlakbij het HTML-element zit, overschrijft het toch alle bovengenoemde stijlen.

Order

Order (volgorde) bepaalt hoe identieke selectors met identieke declarations worden opgeteld of overschreven:

```
h1{ color: grey; font-family: Arial; }
h1{ color: red; }
```

Het eindresultaat van deze CSS rules is een rode tekstkleur met Arial als lettertype.

De proximity (plaats) van stylesheets bepaalt ook order (volgorde). Bijvoorbeeld:

```
<head>
  <link src="externalSheet.css">
  <!--  In externalSheets.css staat de CSS-rule:
        h1{ color: grey; font-family: Arial; }
  -->
  <style>
        h1{ color:red }
  </style>
</head>
```

les: ZI de CSS link!

<link rel="stylesheet" HREF="" media="screen">

Het resultaat is een rode tekstkleur met Arial als lettertype, omdat de style-tag een hogere proximity heeft dan de external stylesheet (het staat dichterbij het HTML-element).

Priority

Niet elke CSS-selector heeft hetzelfde gewicht. Sommige selectors tellen zwaarder mee dan andere. Dit principe noemen we priority (prioriteit). Van laag naar hoog:

- Universal-selector (*)
- Tag-selector (h1)
- Class-selector (.className)
- Id-selector (#idName)
- !Important.

gebruik meer selectors IPV classes!

Een class-selector wint het van de tag-selector. Een id-selector wint het weer van de class-selector. Onderstaand voorbeeld maakt de H1-tag rood:

```
<h1 id="mainHeader" class="belangrijk">Tekst</h1>

h1#mainHeader { color: red; }
h1{ color: grey; }
h1.belangrijk { color: orange; }
```

Bovenstaand voorbeeld laat ook zien dat priority belangrijker is voor het bepalen van de cascade order dan order & proximity!

les: :nth-child(x);

Specificity

Deze theorie omschrijft dat HTML-elementen die direct of nauwkeurig worden aangesproken voorrang krijgen in de cascade order. Anders gezegd: een specifieke CSS-selector wint het van een algemene selector.

In dit voorbeeld wordt de tekst van de h1 rood omdat die selector nauwkeuriger is:

```
<div><h1 id="mainHeader">Tekst</h1></div>

body div h1{ color: red;}
div h1{ color: grey; }
```

In dit voorbeeld wordt de tekst van de h1 rood omdat die selector directer is:

```
<div><h1 id="mainHeader">Tekst</h1></div>

#mainHeader{ color: red; }
body div h1{ color: grey; }
```

les: @Import CSS rule

Inheritance

Inheritance (overerven) houdt in dat bepaalde CSS-properties (eigenschappen) overgaan van parent naar child. Meestal zijn dit teksteigenschappen. Voorbeeld:

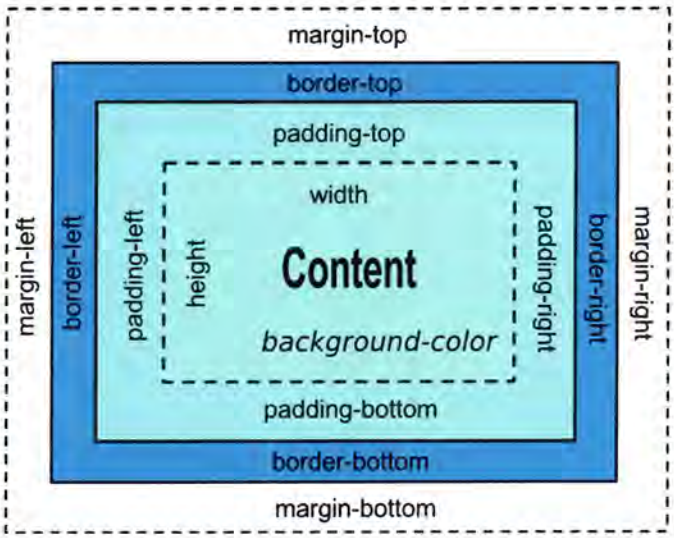
```
<div><h1 id="mainHeader">Tekst</h1></div>
div{ color: red; font-family: Arial; }
```

CSS box model (les)

CSS ziet elk HTML-element als een doos. Dit noemen we de CSS box model. Het bestaat uit:

- **Content:** Dit is de inhoud van een HTML-element. Het kan teksten of andere HTML-elementen (children) bevatten en heeft width, height en background-color eigenschappen.
- **Padding:** De ruimte om de content heen. Padding krijgt dezelfde kleur als de background-color van de content.
- **Border:** De rand om de padding. Borders kunnen een eigen dikte, stijl en kleur krijgen. Gebruik 'border-image' om de border te voorzien van een afbeelding. Dit is overigens een shorthand voor border-image-source, -slice, -width, -outset en -repeat.
- **Margin:** De ruimte aan de buitenkant van de border. De margin is transparant en krijgt de kleur van het onderliggende (parent) element.
- **Outline:** Een rand aan de buitenkant van de border. De outline ligt bovenop het element, waardoor het geen fysieke ruimte in beslag neemt en bepaalde delen kan overlappen. CSS, outline. (les)

  Het wordt gebruikt om een element op te laten vallen. Klik bijvoorbeeld op een input tekstveld, waardoor het oplicht met een zachte blauwe glow.



TRouBLe!

Margin, border en padding hebben vier kanten (top, right, bottom en left), die apart kunnen worden vormgegeven.

Gebruik het "TRouBLe" ezelsbruggetje, zodat je bij shorthand notaties geen fouten maakt. Shorthand notaties maak je met de klok mee van top, naar right, naar bottom naar left:

```
h1{
   padding: 1rem 2rem 3rem 4rem;
   /* Is hetzelfde als: */

   padding-top: 1rem;
   padding-right: 2rem;
   padding-bottom: 3rem;
   padding-left: 4rem;
}
```

Block vs Inline (les)

HTML-elementen worden grofweg ingedeeld in twee verschillende categorieën: block en inline.

- **Block:**
  - Een blockelement wordt even breed als zijn parent.
  - De hoogte wordt bepaald door de content.
  - Een block element forceert boven en onder een line break. Hierdoor kunnen twee block elementen niet op dezelfde regel staan.
- **Inline:**
  - Een inline element wordt even breed en hoog als de content.
  - Twee of meer inline elementen kunnen op dezelfde regel staan (inline).
  - Een inline element met een grote padding of border drukt zichzelf buiten de parent (overflow).
  - Top en bottom margins worden genegeerd. Left en right margins worden wel getoond.
  - Als je een inline element float, gaat het zich als een block element gedragen.

Margin collapse (les)

Als twee elementen met een margin tegen elkaar liggen wordt alleen de grootste margin van de twee gebruikt. De kleinere margin wordt genegeerd. Dit principe noemen we margin-collapse.

De margin collapse leidt tot één enkele marge.
De kleinere marge wordt als het waren opgeslokt door de grotere.

Box model berekeningen (les)

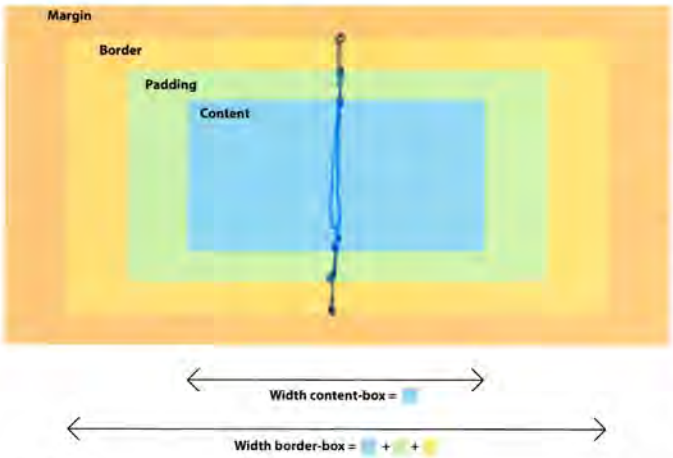
Berekenen hoeveel ruimte een HTML-element in beslag neemt door de onderdelen van de box model bij elkaar op te tellen:

De width van een element = width + padding-left + padding-right + border-left + border-right. Hou verder ook nog rekening met margin-left en margin-right.

De height van een element = height + padding-top + padding-bottom + border-top + border-bottom. Hou verder ook nog rekening met margin-top en margin-bottom.

Box-sizing: border-box

Bij het maken van complexe RWD lay-outs is het soms wenselijk dat de width van een element wordt berekend *inclusief* padding en border. Gebruik hiervoor de border-box value van de box-sizing property.




```
.card{
   box-sizing: border-box;
   width: 25%;
   border: 1px solid black;
   padding: 1rem;

   /* De width is 25% inc. border en padding. */
}
```

Overflow

De overflow property bepaalt wat er gebeurd als content groter is dan de afmetingen van de box. Het is mogelijk de content af te maskeren (clipping) of scrollbars toe te voegen.

Let op: Sinds Mac OS X Lion worden scrollbars automatisch verborgen. Ze worden alleen getoond wanneer ze worden gebruikt, zelfs bij `overflow: scroll;`

`overflow`	- property voor het bepalen van text overflow in de hoogte én breedte.
`overflow-y`	- property voor de hoogte as.
`overflow-x`	- property voor de breedte as.
`visible`	- overflow niet clippen, maar laten overlopen buiten het element (default).
`hidden`	- overflow clippen; de rest van content is onzichtbaar.
`scroll`	- overflow clippen, maar een scrollbar is beschikbaar om de rest van de content zichtbaar te maken. De scrollbar is ook zichtbaar als er geen overflow is.
`auto`	- hetzelfde als scroll, behalve dat er geen scrollbar zichtbaar is als er geen sprake is van overflow.

Vendor prefixing (= work in progress)

Browserproducenten implementeren nieuwe CSS3 functies vaak geleidelijk. Soms duurt het even voordat een nieuwe functie betrouwbaar werkt of totdat het W3C zijn recommendations uitbrengt. In het laatste geval zouden de specificaties tussentijd kunnen wijzigen.

Soms is het mogelijk deze geavanceerde functies voortijdig te gebruiken, door zogenaamde prefixes toe te voegen aan de CSS. Vendor prefixing is een tijdelijke oplossing voor een tijdelijk probleem.

- Android: -webkit-
- Chrome: -webkit-
- iOS: -webkit-
- Safari: -webkit-
- Firefox: -moz-
- IE en Edge: -ms-
- Opera: -o-

-webkit-
-moz-
-ms-
-o-

```
/* Vendor prefixing voor linear-gradient: */

div{
    background: -webkit-linear-gradient(red, yellow);
    background: -o-linear-gradient(red, yellow);
    background: -moz-linear-gradient(red, yellow);
    background: -ms-linear-gradient(red, yellow);
    background: linear-gradient(red, yellow);
}
```

Les: errett:
- text-decoration-color: red
+ TAB
↓
verden prefixes
/
of met Sass

Web Safe Fonts

Web safe fonts zijn lettertypes die op de meeste computersystemen zijn geïnstalleerd. Je kunt ze meestal zonder veel problemen toevoegen aan een font-stack. Voorbeelden:

- **Monospace:** Courier, Courier New, Lucida Console.
- **Serif:** Georia, Times, Times New Roman, Palatino, Garamond.
- **Sans-Serif:** Arial, Arial Black, Arial Narrow, Verdana, Helvetica, Comic Sans MS, Trebuchet MS, Tahoma, Geneva.

Web Fonts

Google Fonts, Adobe Typekit en Adobe Edge Web Fonts zijn populaire online platformen die lettertypes aanbieden voor gebruik in je eigen website.

Je voegt een link-tag toe aan je webpagina, zodat de browser lettertypes direct van het platform kan downloaden. De gebruiker hoeft zelf dus geen lettertypes te installeren.

Bij het gebruik van web fonts moet je goed opletten dat je website niet te zwaar wordt, want elk lettertype voegt tientallen kilobytes laadtijd toe!

@font-face

Sommige lettertypes zijn niet beschikbaar via Google Fonts of andere online platformen, of je wilt een lettertype gebruiken dat specifiek voor jouw klant is ontworpen. Je gebruikt dan de @font-face rule en host lettertypes zelf.

@font-face werkt alleen betrouwbaar als je vijf verschillende bestandsformaten koppelt aan je CSS:

```
@font-face{
    font-family: 'Montserrat Regular';

    src: url("montserrat.eot"); /* IE9 */
    src: url("montserrat.eot?#iefix") format("embed-
    ded-opentype"), /* IE6-IE8 */

    url("montserrat.woff2") format("woff2"),
    url("montserrat.woff") format("woff"),
    url("montserrat.ttf") format("truetype"),
    url("montserrat.svg") format("svg");
}
```

Font Squirrel generator

- **EOT:** is ontwikkeld door Microsoft en wordt alleen ondersteund door Internet Explorer en Edge browsers.
- **WOFF2 en WOFF:** zijn de W3C recommendations. Het is eigenlijk een OpenType of TrueType font met compressie en extra meta data. Het is daarom geschikt in situaties waar weinig bandbreedte beschikbaar is. Het wordt ondersteund door praktisch alle browsers (inclusief Internet Explorer vanaf versie 9).
- **TrueType:** ontwikkeld in 1991 door Apple en Microsoft, als tegenhanger van Adobe's PostScript. Dit is het meest voorkomende fontbestand op zowel Mac als PC. TrueType bestanden zijn geoptimaliseerd voor beeldschermweergave.
- **SVG:** wordt o.a. ondersteunt Android browsers.

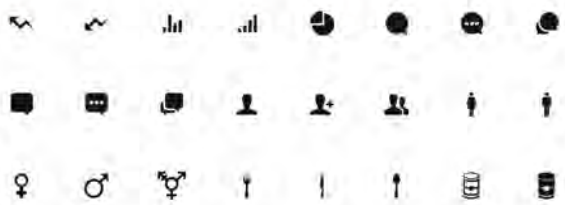
Icon fonts

Lettertypes zijn gebaseerd op vectoren en bestaan uit honderden karakters die we glyphs noemen.

Steeds meer webdesigners ontdekken de kracht van icon fonts. Dit zijn lettertypes die bestaan uit iconen en pictogrammen in plaats van letters, cijfers en leestekens.

Het voordeel van icon fonts is dat de browser één font downloadt, waarmee vervolgens honderden iconen kunnen worden getoond. Icon fonts zijn te combineren met CSS-opmaak. Hierdoor zijn complexe stijlen mogelijk zoals diverse kleuren, slagschaduwen, 3D-effecten, het roteren en schalen van de iconen en zelfs animaties.

Bekende icon fonts zijn Font Awesome, Ionicons en Glyphicon. Daarnaast bevat Unicode honderden glyphs die gebruikt kunnen worden bij het maken van websites.



Icon fonts van Ionicons.

	EOT	WOFF	WOFF2	TrueType	SVG
Chrome	Nee	5.0	36.0	4.0	4.0
IE / Edge	6.0	9.0	Nee	9.0	Nee
FireFox	Nee	3.6	35.0	3.5	Nee
Safari	Nee	5.1	Nee	3.1	3.2
Opera	Nee	11.1	26.0	10.0	9.0

Legenda

`"unit"`		Gebruik rem, em, px, %, ex, vh, vw, cm, mm, in, pc of pt.
`"color"`		Gebruik een HTML color name, hexadecimale, rgb, rgba, hsl of hsla waarde.
`|`	óf	Er zijn meerdere mogelijken, gescheiden door het pipe-teken, "|".
`,`	én	Er is meer dan één waarde mogelijk, gescheiden door de komma.
`[selector]`		Vul op tussen de haakjes een geldige CSS-selector in.

Graphics

`object-fit`

- img of video uitvullen, afsnijden of schalen met behoud van aspect ratio.

fill (default): De afbeelding wordt geschaald naar de afmeting van zijn content box. Indien nodig wordt de afbeelding opgerekt of ingedrukt.

contain: De afbeelding wordt teruggeschaald met behoud van de aspect ratio, zodat het past in zijn content box.

cover: De afbeelding wordt uitgevuld met behoud van de aspect ratio, zodat het in de hoogte of breedte past in zijn content box. De rest van de afbeelding wordt afgeknipt.

none: De afbeelding wordt niet teruggeschaald.

scale-down: De afbeelding tonen zoals none of contain opgegeven zouden worden.

Backgrounds

`background-position`

- De positie van de background-image t.o.v. de padding-box (tenzij anders ingesteld met background-origin).
  0% 0% (default) | top | center | bottom | left | right | xpos ypos (cijfer + "unit")

`background-size`

- Bij cover en contain wordt de afbeelding geschaald zodat het in het element past. Cover schaalt over de breedte en contain over de lengte.
  auto (default) | width height (cijfer + "unit") | cover | contain

`background-attachment`

- De manier hoe een background-image beweegt tijdens het scrollen t.o.v. de viewport:
  scroll (default): bij het scrollen beweegt de achtergrond niet mee met de content.
  local: de content én de achtergrond bewegen bij het scrollen.
  fixed: de achtergrond is vastgezet t.o.v. het browserscherm. Dit is pas zichtbaar als de body scrollt. Dit geeft het populaire 'kijkdoos' paralax effect.

`background-origin`

- Uitgangspunt waar de background-image voor het eerst wordt getoond in de boxmodel.
  padding-box (default) | border-box | content-box

`background-clip`

- Uitgangspunt waar de background-color of background-image clipped in de boxmodel.
  border-box (default) | padding-box | content-box

`background-blend-mode`

- De manier waarop twee backgrounds (bijvoorbeeld een kleur en een afbeelding) onderling met elkaar worden gemend.
  normal (default) | multiply | screen | overlay | darken | lighten | color-dodge | saturation | color | luminosity

Gradients

Gebruik gradients als value van de background shorthand property of als value van background-image. Vanwege browser compatibiliteit is het gebruik van vendor prefixes aan te raden. Gebruik gradients niet in combinatie met een body-tag-selector en gebruik rgba-waarden voor maximale flexibiliteit.

`linear-gradient()` - direction, color-stop-1, color-stop-2, etc. Voorbeelden:
linear-gradient(red, yellow);
linear-gradient(to right, red, yellow, green);
linear-gradient(to right, rgba(255,0,0,0), rgba(255,0,0,1));

`radial-gradient()` - shape size at position, color-stop-1, color-stop-2, etc. Niet compatible met alle browsers.
Voorbeeld: radial-gradient(rgba(255,255,0,1.0) 50%, rgba(255,0,0,1.0) 75%);

`repeating-linear-gradient()`
`repeating-radial-gradient()` - repeterende verlopen maken. De syntax is vergelijkbaar met bovenstaande voorbeelden.

Sprite images

Een sprite image is een verzameling afbeeldingen die samen in één enkele afbeelding zijn gestopt. Een website met veel afbeeldingen kan traag laden omdat elke afbeelding door de browser moet worden opgevraagd met een http-request aan de webserver. Het gebruik van één enkele sprite image is dus voordelig voor de snelheid van je website. Verder maken sprite images de bestandsmanagement eenvoudiger.

Je bepaalt met CSS welk gedeelte van de sprite image je wilt tonen. Voorbeeld:

```
#home {
    width: 46px;
    height: 44px;
    background: url(img_navsprites.gif) 0 0;
}
```

```
#next {
    width: 43px;
    height: 44px;
    background: url(img_navsprites.gif) -91px 0;
}
```

Pseudo-classes

Pseudo-classes gebruiken we om een bepaalde 'state' van een element vorm te geven. Er zijn erg veel pseudo-classes, waarvan onderstaande het meest populair zijn:

`a:link` - selecteert een normale, nog niet bezochte weblink.
`a:visited` - selecteert een reeds bezochte weblink.
`[selector]:hover` - 'hover'-state van een bepaalt element (mouse-over).
`a:active` - 'active'-state van een link (mouse-down).

`[selector]:first-child` - selecteert alle eerste children van een parent.
p:first-child selecteert alle paragrafen die het eerste child zijn van hun parent.
`[selector]:last-child` - selecteert de laatste child van een parent.
`[selector]:first-of-type` - selecteert een child die het eerste van zijn type is, in een parent.
`[selector]:last-of-type` - selecteert een child die het laatste van zijn type is, in een parent.
`[selector]:nth-child(n)` - selecteert het zoveelste child van een parent, waarbij `n` een getal moet zijn.
Er kunnen ook children worden geselecteerd aan de hand van een serie.
`3n+2` selecteert elk tweede element in een serie van drie (de 2de, 5de, 8de, 11de, etc).

`[selector]:focus` - selecteert elementen met focus (een aangeklikt / actief input-veld bijvoorbeeld).
`input:checked` - selecteert een aangevinkte checkbox of actieve radio button.
`input:invalid` - selecteert input-elementen met ongeldige waarde (werkt i.c.m. het pattern-attribute).
`input:valid` - selecteert input-elementen met geldige waarde (werkt i.c.m. het pattern-attribute).

Pseudo Elements

Met pseudo elements kunnen specifieke gedeeltes van een element worden vormgegeven.

`::first-letter`	- De eerste letter van een element vormgeven. Hiermee kunnen zogenaamde 'initials' en 'drop cap' worden gemaakt. Dit werkt alleen bij block tags.
`::first-line`	- De eerste regel van een element vormgeven (werkt alleen bij block tags).
`::selection`	- Het gedeelte vormgeven dat geselecteerd is door de gebruiker. Gebruik enkel color, background, cursor en outline properties.
`::before, ::after`	- Content invoegen vóór of na een element. Gebruik deze selector i.c.m. de content property. Ook is het vaak zinvol om deze pseudo elementen te voorzien van display: inline-block of block properties.

WKH

Voorbeeld Pseudo Elements

In onderstaand voorbeeld wordt bij een hyperlink (met rel="external"), onmouseover, een tooltipje gezet.

De tekst van de tooltip wordt door CSS uit de zelfverzonnen 'data-title'-attribute gehaald. Voor de hyperlink wordt een icoon uit unicode (een wereldbol) geplaatst, zodat duidelijk te zien is dat men de website verlaat.

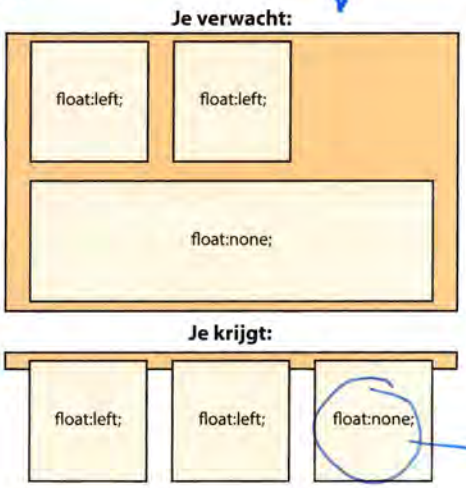
```
<a rel="external" data-title="U verlaat onze website. Jammer!" href="http://www.google.com">google</a>

a[rel="external"]::before{ /* Wereldbol Unicode character voor de a-tag zetten. */
    display: inline-block;
    padding: 0 0.6em 0 0;
    content: "\1f30d";
    font-size: 0.7em;
}

a[rel="external"]:hover::after{ /* Hover tooltip, met als tekstcontent de data-title attribute. */
    position: absolute;
    top: 1.7em;
    left: 1.3em;

    display: block;
    width: 160px;
    padding: 0.5em;
    background-color: #f5f773;
    border: 1px solid black;
    box-shadow: 2px 2px 2px rgba(0,0,0,0.5);

    color: black;
    font-size: 0.7em;

    content: attr(data-title);
}
```

secret source

Float (les)

Met de float property kun je block tags visueel naast elkaar zetten in je webpagina. Elementen voorzien van float properties geeft een paar eigenaardigheden:

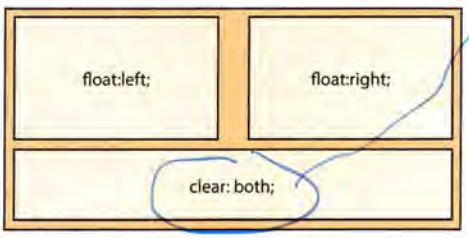
```
<section>
    <div style="float:left;"></div>
    <div style="float:left;"></div>
    <div></div>
</section>
```



Clearing floats met een child element (les)

In bovenstaand voorbeeld zie je dat de derde child geen float property heeft, maar desondanks toch float! Dit komt omdat de derde child in de HTML direct onder een gefloate tag staat en daarom ook gaat floaten.

Je gaat dit ongewenste gedrag tegen door de derde child te voorzien van de clear property:



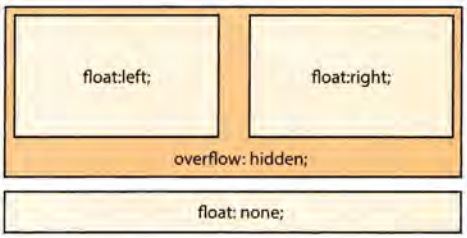
Bij links floaten gebruik je `clear:left;` en andersom gebruik je `clear:right;`. In de praktijk wil je hier niet over nadenken. Gebruik daarom `clear:both;`

Clearing floats met een parent element (les)

De overflow property bepaalt wat er gebeurd als content groter is dan de afmetingen van de box. Je kunt er ook floats mee opheffen door de parent te voorzien van een overflow property.

Het maakt overigens niet uit welke waarde van overflow je gebruikt. In de praktijk wordt `overflow:hidden;` gebruikt.

```
<section style="overflow:hidden;">
    <div style="float:left;"></div>
    <div style="float:left;"></div>
</section>
<div></div>
```



Height collapse (les)

Als children in een parent floaten, dan verliest de parent zijn hoogte eigenschappen. Het is dan alsof de parent helemaal geen content bevat! Dit principe noemen we 'height collapse'.

Ook dit probleem ga je tegen door de parent een overflow property te geven. & clearfix.

Display: inline-block

Een alternatief voor float is `display:inline-block;`. We maken een soort hybride element dat het beste van twee werelden biedt:

1. Het element gedraagt zich inline; in de zin dat twee elementen naast elkaar op dezelfde regel mogen staan.
2. De boxmodel van het element gedraagt zich echter niet inline, maar juist als block element.

Inline en inline-block elementen worden op de tekstbaseline van hun parent gezet. Gebruik `vertical-align:top;` om ze bovenin de parent te positioneren.

Je weet dat white spaces in de HTML-code worden getoond als één enkele spatie. Dit betekent dat er tussen twee inline-block elementen meestal ook een spatie staat. Dit kan problemen geven want 50% + spatie + 50% ≠ 100%!

Los het op door de parent `font-size:0;` te geven. Let op: de inline-block elementen zullen natuurlijk die `font-size:0;` erven!

Position: static (les)

Normaal gesproken worden elementen door de browser gerenderd in volgorde van de HTML-code ('document flow' genoemd). `position:static;` is de browser default voor alle tags.

Position: absolute (les)

`position:absolute;` haalt het element uit de normale document flow. Het is een beetje alsof je een aparte Photoshop laag aanmaakt. Het element wordt gepositioneerd t.o.v. de eerste ancestor die niet static gepositioneerd is.

Met de properties top, right, bottom en left kun je het element pixelprecies positioneren. `top:0; left:0;` zet het element in de linker bovenhoek van het webdocument.

Position: relative (les)

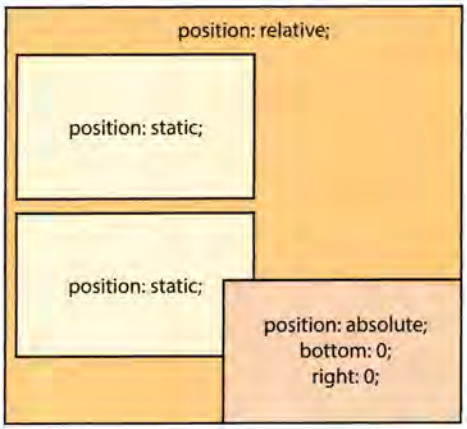
Met `position:relative;` kun je een element verschuiven t.o.v. zijn normale positie in de document flow. Met andere woorden: je geeft een offsetpositie, zonder de elementen er omheen te beïnvloeden. Voorbeeld: `top:-10px; left: 50px;` zet het element tien pixels omhoog en vijftig pixels naar rechts t.o.v. zijn normale static positie.

Absoluut positioneren t.o.v. een parent

Vaak wil je een element absolute positioneren t.o.v. zijn parent. De oplossing is de parent `position:relative;` te geven en de child `position:absolute;`. De parent is nu de eerste anchestor die niet static is. Onderstaand voorbeeld zet een child in de rechter onderhoek van zijn parent:

```
<section class="parent">
    <div class="child"></div>
    <div class="child"></div>
    <div class="absoluteChild"></div>
</section>

.parent{
    position: relative;
}

.child{
    position:static; /* browser default value. */
}

.absoluteChild{
    position: absolute;
    bottom: 0px;
    right: 0px;
}
```

!! Toets: popup (les: 7)
↳ position.



Position: fixed (les)

`position:fixed;` positioneert het element t.o.v. de viewport. Dit betekent dat het element niet meescrollt met het document en 'gefixeerd' in beeld blijft staan. Handig voor een navigatiemenu dat bovenin de viewport zichtbaar moet blijven, ongeacht de positie van de scrollbalk.

Position: sticky (les)

Dit is een nieuwe value die nog niet in alle browsers werkt. Het element gedraagt zich relatief ***of*** fixed, afhankelijk van de scrollbalkpositie.

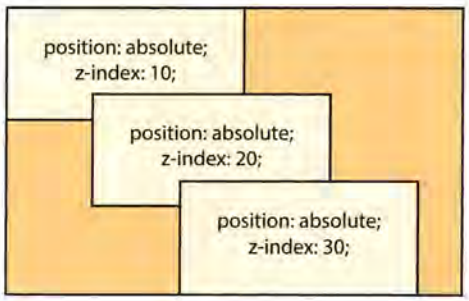
Het is relatief, totdat het element door het scrollen uit de viewport dreigt te verdwijnen. Daarna wordt het fixed en blijft het gefixeerd staan.

`position:sticky;` is vergelijkbaar met hoe headercellen zich gedragen in Exceldocumenten. Ze staan altijd bovenin beeld, ongeacht de scrollbalkpositie. Andere cellen schuiven tijdens het scrollen als het ware onder de headercellen door.

Z-index (les)

De z-index property bepaalt de stapelvolgorde van een absolute, relative of fixed gepositioneerd element met een (negatief) nummer als value.

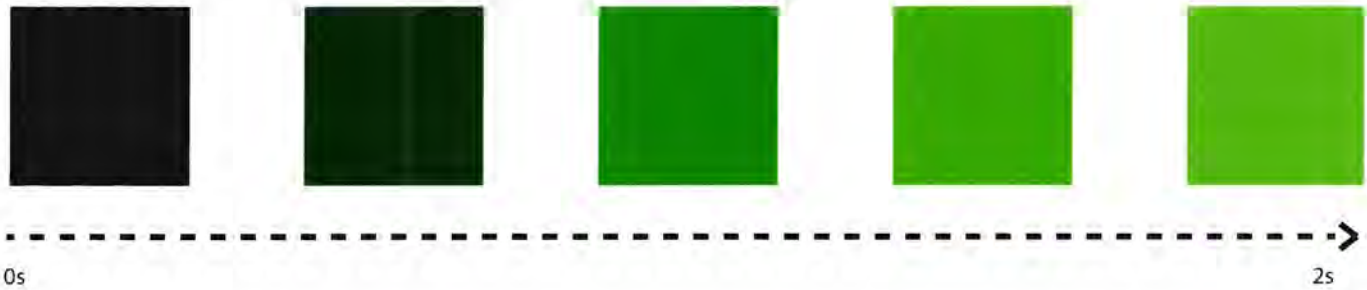
De default is `z-index:auto;`. Dit stapelt het element op gelijke hoogte met zijn parent.



CSS Transitions (les)

Met CSS3 transitions kun je in een bepaalt tijdsbestek de value van een property geleidelijk veranderen. Een transition start door een event, zoals een mouse over. In CSS gebruik je voor de events bepaalde pseudo-classes zoals :hover, :checked, :valid, :invalid of :focus. Shorthand voorbeeld:

```
.blackToGreen {
    background-color: black;
    transition: background-color 2s linear;
}

.blackToGreen:hover{
    background-color: green;
}
```



De belangrijkste properties en values zijn als volgt. Een complete lijst vind je op w3schools.com:

`transition-property` - Komma gescheiden propertylijst met te animeren properties.

`transition-delay` - Komma gescheiden lijst om voor elke property een startvertraging in te stellen. Gebruik seconde of milliseconde (bijvoorbeeld 2s of 250ms). Logischerwijs is 0s geen startvertraging; de animatie zal direct starten bij het event.

`transition-duration` - Komma gescheiden lijst om voor elke property duur in te stellen (in s of ms).

`transition-timing-function` - Komma gescheiden lijst om voor elke property de animatie curve in te stellen.
ease: de default value: trage start, dan snel, dan een traag einde.
linear: de animatie verloopt lineair.
ease-in: trage start, dan versnellen.
ease-out: snel begin, dan vertragen.
ease-in-out: trage start, traag einde.

`transform` - 2D en 3D transformaties toevoegen aan een element zoals translate(), rotate(), scale() en skew().

transition-timing-functions

@keyframes animaties (les)

Met de @keyframes CSS-rule maak je complexe animaties. Keyframes bepalen een verandering van een eigenschap op een bepaalt moment. Voor een animatie heb je daarom minimaal twee keyframes nodig. Een startpunt (from) en een eindpunt (to). Je kunt natuurlijk ook meerdere keyframes gebruiken voor complexere animaties. Je gebruik dan percentages. 0% is hetzelfde als 'from' en 100% is hetzelfde als 'to'.

@keyframes heeft geen event of pseudo class nodig: de animatie start direct nadat het webdocument door de webbrowser is geladen. Shorthand voorbeeld:

```
@keyframes infiniteSpinning {
    from {
      transform: rotate(0deg);
    }
    to {
      transform: rotate(360deg);
    }
}

div{
    animation: infiniteSpinning 2s linear infinite;
}
```

`@keyframes animatieNaam{}` - De @keyframes rule bevat eerst de naam van jouw animatie. Tussen de curly braces staan weer twee rules (genest dus), waarmee je het begin- en eindpunt van de animatie opgeeft (from, to en/of percentages).

`from{} | to{} | percentage%{}` - In de @keyframes rule staan minimaal twee keyframe rules. Het beginpunt van de animatie is from{} of 0%{}. Het eindpunt van de animatie is to{} of 100%{}.

Je kunt zo veel keyframes gebruiken als je wilt. Bijvoorbeeld op het begin, op een kwart (25%{}), halverwege (50%{}) en aan het eind van de animatie.

`animation-name` - In de css-rule van het element dat je wilt animeren zet je de animation-name property. De value wordt de betreffende naam van jouw @keyframes-rule. Meerdere animatie koppelen kan door een kommagescheiden lijst op te geven. Voorbeeld:

```
@keyframes fadeout{
    from{ opacity: 0; }
    to{ opacity: 1; }
}

@keyframes moveLeft{
    from{ left: 0px; }
    to{ left: 1000px; }
}

#balletje{
    animation-name: fadeout, moveLeft;
    animation-duration: 250ms, 2s;
}
```

`animation-delay` - Zie transition-delay.

`animation-duration` - Zie transition-duration.

`animation-timing-function` - Zie transition-timing-function. Bekijk op w3schools.com de handige steps() method!

`animation-iteration-count` - Geef aan hoe vaak de animatie moet worden herhaald. Oneindig herhalen is de waarde infinite.

Viewport

De viewport is het gedeelte van het browservenster waarin de website te zien is. De titel-, adres, bookmark-, tab- en statusbar vallen niet onder de viewport.

Responsive Web Design (RWD)

Responsive Web Design (RWD) is een CSS-techniek om de layout van een webpagina aan te passen, aan de hand van viewportafmetingen.

Adaptive Web Design → JS

JavaScript verstuurt tijdens het laden van de webpagina eerst browsereigenschappen naar de server. De server antwoordt met een geoptimaliseerde versie van de webpagina a.d.v. de vastgestelde viewportafmetingen.

Adaptive design is minder flexibel en technisch moeilijker te realiseren dan RWD. In de praktijk biedt het geen voordelen (het is vaak trager en leunt op JavaScript), waardoor het steeds minder voorkomt.

Waarom Responsive Webdesign?

- **Geen dubbel werk:** Voor de komst van RWD was er adaptive web design: we maakten speciale mobiele versies van een website. Dit geeft dubbel werk tijdens het ontwikkelen, testen en het beheer.

- **Betere leesbaarheid:** Met RWD wordt de typografie nauwkeurig afgestemd voor verschillende schermgroottes. Dit maakt pinchen en inzoomen grotendeels overbodig. Ook afbeeldingen passen zich aan, zodat ze scherp worden getoond op hoge resolutie schermen.

- **Gebruikersvriendelijker:** De user experience is afgestemd op het device en het scherm. Websites op kleine schermen zijn daarom voorzien van uitklapbare navigatiemenu's en andere ruimtebesparende GUI-elementen. Mouse click events werken niet op een touch screen. Een goede ontwikkelaar ontwerpt websites voor alle devices, schermen en browsers.

Breakpoints

Een breakpoint is een drempelwaarde in RWD, waarbij de browser kiest tussen twee CSS-rules. Meestal is een breakpoint gebaseerd op een bepaalde schermbreedte (in pixels), maar breakpoints op basis van pixelratio of schermoriëntatie (portait of landscape) komen ook voor.

De meeste websites hebben minimaal twee breakpoints, waarmee drie verschillende schermtypes worden ondersteund. Tussen de breakpoints moet het ontwerp zich dynamisch en vloeiend kunnen aanpassen.

De positie en de hoeveelheid breakpoints wordt bepaald door de complexiteit van het webdesign, door user research en door te kijken naar CSS frameworks. Bootstrap hanteert bijvoorbeeld de volgende breakpoints:

- Extra small devices < 576px
- Small devices: >= 576px
- Medium devices: >= 768px
- Large devices: >= 992px
- Extra large devices: >= 1200px

Bootstrap

Mobile first!

Android devices worden online het meest gebruikt: méér dan Windows computers! In de toekomst zal het aandeel mobiele telefoons alleen maar toenemen. Websites moeten daarom worden ontwikkeld met een 'mobile first' mentaliteit.

Een goede designer bepaalt aan de hand van user research welke schermen de doelgroep gebruikt.

Vervolgens worden *éérst* wireframes en (interactive) prototypes gemaakt voor de kleinste schermen.

Websites ontwikkelen voor kleine schermen is lastig omdat er weinig ruimte beschikbaar is, maar als hiervoor alle GUI-, usability- en accessibilityproblemen worden opgelost zal het design probleemloos opschalen naar grotere schermen!

Bij het opstellen en het testen van de CSS wordt er ook éérst gekeken naar de kleinste schermen. Men werkt vervolgens naar de grotere schermen toe.

Deze manier van denken heeft ook invloed op de breakpoints. Onderzoek en test of voor jouw webdesign de volgende breakpoints van toepassing zijn:

bes: < 480 px met gedefinieerd technieken

- Small phones: < 320 pixels
- Phones: < 480 pixels
- Large phones / small tablets: < 512 pixels
- Tablets: < 768 pixels
- Large tablets / small desktops: < 992 pixels
- Desktops: > 1200 pixels

Mobile 1st

Let op: meer breakpoints betekent meer testen én meer CSS-code!

Test je website op zo veel mogelijk verschillende browsers, besturingssystemen, devices en schermen!

Meta name="viewport" (les)

Met een speciale meta-tag geven we de browser een aantal instructies. Zo zetten we de breedte van de webpagina gelijk aan de breedte van het device, omdat we het uiterlijk en de grootte van content hoofdzakelijk bepalen aan de hand van de breedte van het (device)scherm.

Een webpagina kan niet responsive worden uitgevoerd zonder de meta name="viewport" tag. Het is *onmisbaar* in de RWD workflow! Voorbeeld:

```
<meta name="viewport" content="width=device-width,
initial-scale=1.0, minimum-scale=1.0, maxi-
mum-scale=3.0, user-scalable=yes">
```

- `width=device-width` - de browser schaalt de breedte van de website terug, of op, naar de breedte van het device (meestal is dat de viewport). Deze setting is onmisbaar in een RWD workflow!

- `initial-scale` - de zoomsetting van de webpagina na het inladen. 1.0 betekent 100% zoom, of te wel: De website niet in- of uitzoomen na het laden. Dit is de aangeraden setting.

Omrekt:

meta: vp <tabs>

- `minimum-scale` - hoe ver de gebruiker kan uitzoomen door te pinchen. Vanuit accessibility oogpunt is aan te raden dat de gebruiker de website niet verder kan verkleinen dan 100%. 1.0 is dus de aangeraden setting. Deze setting mag niet ontbreken, anders werkt de CSS min-width property mogelijkerwijs niet goed.

- `maximum-scale` - hoe ver de gebruiker kan inzoomen door te pinchen. Vanuit accessibility oogpunt is het sterk aan te raden dat de gebruiker de website minimaal een factor 2 kan vergroten (2.0).

  De gebruiker moet echter ook weer niet te veel kunnen inzoomen, omdat hij anders het overzicht kan verliezen. Een waarde boven 3.5 gebruiken we in de praktijk niet.

- `user-scalable` - aangeven of de gebruiker mag pinchen. Gebruik yes of no (1 en 0 werkt echter ook).

Media Queries

Media Queries zijn kleine CSS wrappers. Ze zijn er in drie varianten:

1. Binnen de head-tag. Het media attribute van de link-tag zorgt voor een specifiek stylesheet per scherm of device:

   ```
   <link rel="stylesheet" media="(min-width:
   768px)" href="css/tablet.css">
   ```

2. Door gebruik te maken van de @import methode in de CSS stylesheets zelf:

   ```
   @import url(xlarge.css) and (min-width: 1200px);
   ```

3. Direct geschreven in de CSS stylesheet zelf. Dit is de aangeraden methode:

   ```
   @media (max-width: 320px){
     body{
       font-size:0.75rem;
     }
   }
   ```

Voorbeeld

Je kunt de Media Queries eenvoudig interpreteren.

```
@media only screen
and (min-device-width : 320px)
and (max-device-width : 480px) {
  body {
    background: red;
  }
}
```

Hier staat: als het een screen device betreft en de breedte van dit device is tussen 320px en 480px dan wordt de achtergrond van de body rood.

Flexbox Layout

Met de CSS3 Flexbox Layout module maak je complexe pagina layouts. Het vervangt grotendeels de float-, display- en position-technieken. De module is oorspronkelijk bedoeld voor het opstellen van lay-outs in één dimensie, of te wel een lay-out in één richting (de breedte óf hoogte), zoals applicatiecomponenten, cards en UI-onderdelen.

Flexbox is hard op weg om een W3C Recommendation te worden, maar zo ver is het nog niet. Gelukkig is de browser-ondersteuning op het moment van schrijven erg goed. Als je doelgroep moderne browsers gebruikt kun je deze techniek zonder al te veel problemen gebruiken.

`display:`

`flex|inline-flex;`

Block Inline

- Een flex-container die zich t.o.v. omliggende elementen gedraagt als block (flex) of inline (inline-flex). De directe descendants (children) van de flex-container noemen we de flex-items. Ze worden uitgelijnd op de denkbeeldige 'flex-line' van de flex-container. Er zijn twee assen voor de flex-line: de horizontale 'main axis' en de verticale 'cross axis'.

`direction:`

`ltr|rtl;`

- Direction is eigenlijk bedoeld om de leesrichting van tekst aan te geven. Normaal gesproken definieer je `direction` met het dir-attribute in het root-element van de HTML en niet met CSS.

  Direction is het uitgangspunt voor de `flex-direction` property. In 99,99% van de gevallen defineer je deze property niet en gaan we uit van ltr.

1 `flex-direction:`

`row|row-reverse|column|column-reverse;`

- Het uitlijnen van de flex-line op één van de twee axis. `row` betekent uitlijnen op de main axis van links naar rechts (de x-as) en `column` betekent uitlijnen op de cross axis van boven naar beneden (de y-as).

2 `flex-wrap:`

`nowrap|wrap|wrap-reverse;`

- Als flex-items samen breder zijn dan de flex-container, dan zullen de flex-items bij nowrap smaller worden, zodat ze als nog passen op één flex-line. Bij `wrap` springen de flex-items die niet meer passen naar een volgende flex-line.

3 `justify-content:`

`flex-start|flex-end|center|space-between|space-around`

- Uitgaande van `flex-direction:row;` is `justify-content` te vergelijken met de horizontal alignment van flex-items: het uitlijnen op de x-as van links naar rechts. `space-between` vult de beschikbare ruimte tussen flex-items op. `space-around` vult de beschikbare ruimte vóór, tussen en na flex-items op.

4 `align-items | align-self:`

`stretch|flex-start|flex-end|center|baseline`

default

- Uitgaande van `flex-direction:row;` is `align-items` te vergelijken met de vertical alignment van flex-items: het uitlijnen op de y-as van boven naar beneden. Standaard wordt de hoogte van flex-items uitgerekt tot de hoogte van de flex-container (`stretch`).

  De `align-items` property pas je toe in de flex-container. Gebruik `align-self` in een flex-item om deze te laten afwijken van de rest.

`align-content:`
`stretch|flex-start|flex-end|center|space-between|space-around`

- bepaalt hoe flex-lines onderling uitgelijnd worden in de flex-container. Uiteraard werkt dit alleen i.c.m. `flex-wrap:wrap;`. Uitgaande van flex-direction:row is de uitlijning verticaal over de y-as.

`flex-basis:`
`auto|%|content`

- bepaalt de afmeting van één bepaalde flex-item, voordat er geen ruimte meer beschikbaar is in de flex-container.

`order:`
`number`

- bepaalt de onderlinge volgorde van flex-items op de flex-line. De default is 0.
  -1 = naar links
  -2 = nog meer naar links

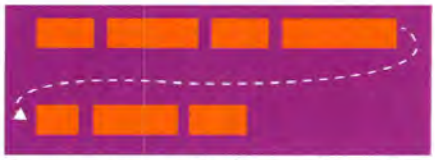
Voorbeelden

Onderstaande voorbeelden zijn overgenomen van 'A Complete Guide to Flexbox', gepubliceerd door Chris Coyier op css-tricks.com.

1

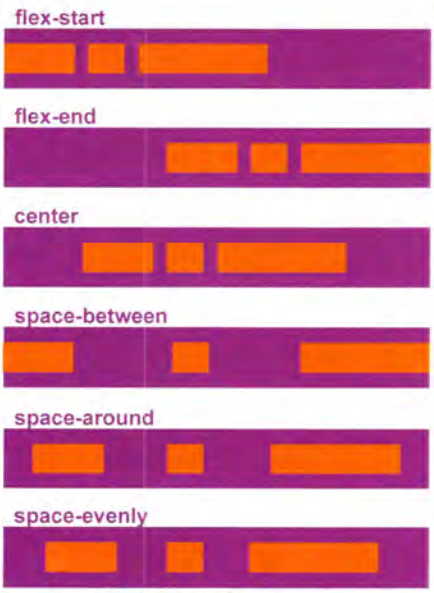
flex-direction: row, row-reverse, column en column-reverse.

2



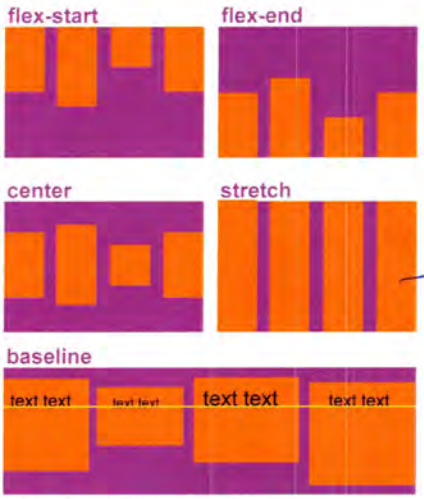
flex-wrap: wrap zet flex-items springen bij gebrek aan ruimte naar een nieuwe flex-line.

3



justify-content uitgaande van flex-direction:row.

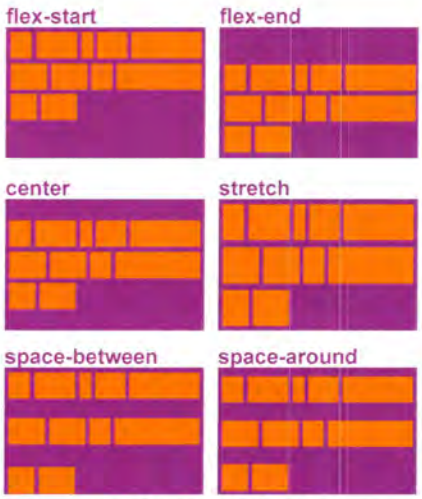
4



default

align-items uitgaande van flex-direction:row.

5



align-content bepaalt de uitlijning van flex-lines. Het werkt alleen i.c.m. flex-wrap:wrap.

Grid Layout

Niet gebruiken voor opdrachten

Grid Layout is het broertje van Flexbox. Het is ontwikkeld voor lay-outs in twee dimensies. Dat wil zeggen een layout in twee richtingen (breedte én hoogte), zoals de gehele webpagina.

Ondanks dat Firefox, Safari en Chrome in het voorjaar van 2017 Grid Layout zijn gaan ondersteunen, blijft de Grid Layout nog even toekomstmuziek.

De browserondersteuning is op het moment zo'n 70 à 85%. Microsoft Edge en de mobiele browsers lopen achter op het moment.

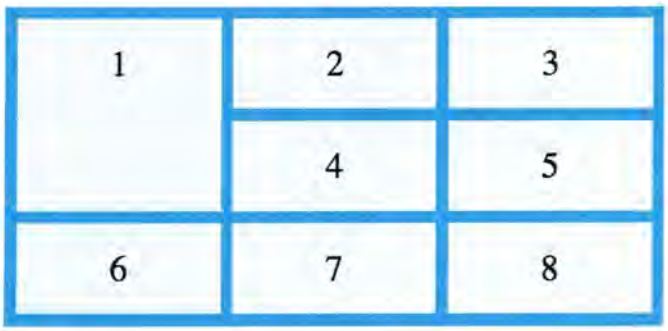
Gebruik dus nog even 'klassieke' gridsystemen zoals Bootstrap, Bourbon Neat of Suzy.

Terminologie

Een parentelement met `display:grid;` is de grid-container. De children van de grid-container noemen we grid-items.

Codevoorbeeld

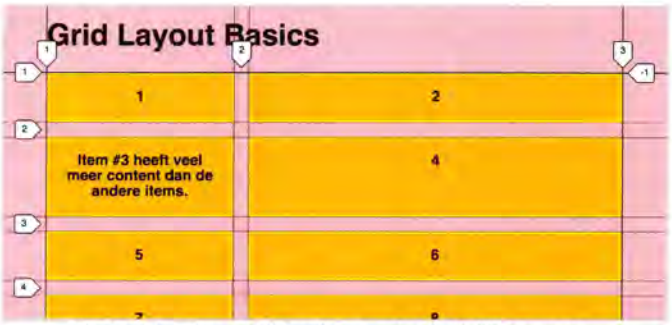
```
.grid-container {
  display: grid;
  grid-template-columns: auto auto auto;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}

.grid-container > div {
  background-color: rgba(255, 255, 255, 0.8);
  text-align: center;
  padding: 20px 0;
  font-size: 30px;
}

.item1 {
  grid-row-start: 1;
  grid-row-end: 3;
}

<div class="grid-container">
  <div class="item1">1</div>
  <div class="item2">2</div>
  <div class="item3">3</div>
  <div class="item4">4</div>
  <div class="item5">5</div>
  <div class="item6">6</div>
  <div class="item7">7</div>
  <div class="item8">8</div>
</div>
```



Tracks

Werken met Grid Layout is lastig omdat browsers weinig tot geen visuele feedback geven over het grid.

FireFox Developer Edition loopt op het moment van schrijven voorop en kan 'tracks' (de collectieve naam voor je kolommen en rijen) tonen in de inspector-window van de developer tools. Gebruik hiervoor de layout-tab.



FireFox Developer Edition kan tracks duidelijk laten zien en voorziet ze van handige tabjes met tracknummers.

Explicit en Implicit tracks (les)

Bewerk het codevoorbeeld van deze pagina als volgt:

```
.grid-container {
  display: grid;
  grid-template-columns: 200px 400px;
  grid-gap: 10px;
  background-color: #2196F3;
  padding: 10px;
}
```

FireFox toont een grid met twee kolommen en drie tracks. Tracks die je zelf hebt opgegeven noemen we explicit tracks. De derde track aan de rechter kant van de tweede kolom heeft FireFox automatisch voor ons gemaakt: een implicit track.

Fractional Unit (les)

Speciaal voor grid-layout is er de speciale unit genaamd 'fr': de fractional unit. 1fr is de ruimte die overblijft na het plaatsen van explicit grid items.

De fr-unit houdt rekening met grid-gaps, margins, borders, etc. Je hoeft dus niet meer zelf te rekenen om de lege ruimte van een parent op te vullen!

Gebruik fr in plaats van procenten. Grid-items worden dan gelijkmatig in de grid-container geplaatst.

```
grid-template-columns: 200px 200px 1fr;
/* Als de browser window 1200px breed is, dan is
1fr 800px. */

grid-template-columns: 1fr 1fr;
/* Twee gelijke kolomm. */
```

Vaststaat: Free space

Grid-container

```
grid-template-columns: 100px 200px auto;
```

grid-template-columns bepaalt de hoeveelheid en de breedte van kolommen. De afmeting van de kolom 'auto' wordt bepaald door de container en de afmeting van de grid-items.

```
grid-template-rows: 25vh 1fr;
```

grid-row-start
grid-row-end

grid-template-rows bepaalt de hoeveelheid en de hoogte van de rijen. Dit code voorbeeld maakt een grid-container met daarin twee rijen: eentje van 25% van de viewport height en de andere rij neemt de overgebleven ruimte in beslag.

```
grid-template-columns: repeat(5,1fr);
grid-template-rows: repeat(3,500px);
```

De repeat function maakt het schrijven van korte code makkelijk. `repeat(5,1fr)` is hetzelfde als `1fr 1fr 1fr 1fr 1fr`.

Repeat 12 gelijke kolommen:
`grid-template-columns: repeat(12, 1fr);`

```
grid-row-gap: 1rem;
grid-column-gap: 20px;
grid-gap: 1rem 20px;
```

De ruimte tussen de grid-items. grid-gap is shorthand voor respectievelijk grid-row-gap en grid-column-gap.

```
grid-auto-flow: row | column;
```

Bepaalt waar implicit gemaakte grid-items komen te staan: in een rij onder de explicit grid-items, of in een kolom naast de explicit grid-items.

```
grid-auto-columns: 500px;
grid-auto-rows: 200px;
```

Bepaalt hoe groot de implicit toegevoegde kolommen / rijen moeten worden.

Grid-items

```
grid-column-start: 2;
grid-column-end: 3;
grid-row-start: 1;
grid-row-end: 2;
grid-column: span 3;
grid-row: span 2;
```

Een grid-item exact plaatsen op een bepaalde track in de grid-container.

Grid-column en grid-row zijn shorthand properties voor grid-column-start, grid-column-end, grid-row-start en grid-row-end.

Gebruik grid-column en grid-row properties om grid-items een explicit width en height te geven. Als er in een rij niet genoeg ruimte beschikbaar is, dan komt het grid-item op de volgende rij terecht en laat een gat achter.

Grid-items short-hand voorbeelden

```
grid-column: 2 / 5;          /* start on 2, end on 5 */
grid-column: span 2 / 5;     /* span 2, end at 5 */
grid-column: 1 / span 2;     /* start at 1, span 2 */
grid-column: 1 / -1;         /* start at 1, end op het einde */
grid-column: 1 / -2;         /* start at 1, end op het twee vanaf het einde */
grid-row: 2 / span 3;        /* start op track 2 en wordt drie hoog. */

grid-row: 1/ -1;             /* werkt niet totdat je grid-template-rows definieert in de grid-container. De
                             browser moet weten waar jouw explicit grid eindigt! */
```

- Firefox developer edition
- Template areas (les)

Adobe Illustrator CC - Interface

6) Properties panel.
In Photoshop staat bovenin de option bar met daarin gerelateerde opties en parameters voor de actieve tool. Illustrator gebruikt hiervoor de properties panel. Ook hier wisselt inhoud op basis van context; zoals de actieve tool of de actieve selectie.

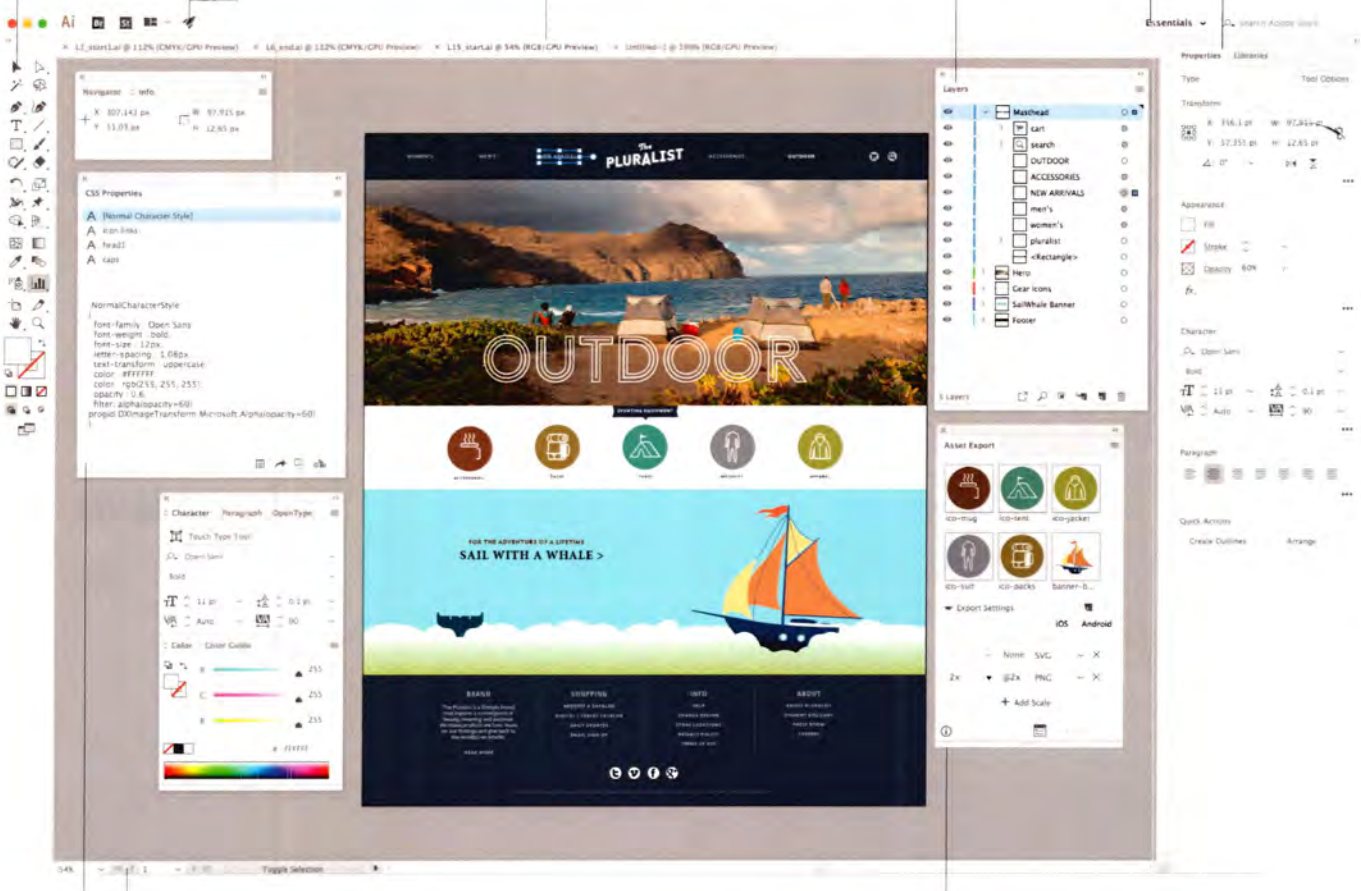
1) Tools panel.
De column graph tool is de actieve tool.

2) GPU performance.
Instellingen voor de snelle GPU-accelerated preview mode.

3) File tabs.
Toont geopende bestanden. Het huidige bestand is lichtgrijs gekleurd.

5) Het Workspace menu.

4) Layer panel.



8) Zoom en Artboards.
Zoomsettings (links) met daarnaast Artboard Navigation.

9) Asset Export panel.
Drag artwork naar dit paneel om assets te genereren. Elke groep van het geselecteerde artwork wordt één asset. Option-drag het artwork naar dit panel om één enkele asset te maken van meerdere groepen. Je kunt assets in de panel hernoemen en vervolgens exporteren. Dit is vergelijkbaar met de Photoshop 'Export As...' optie van het layer panel.

7) CSS Properties.
Indien mogelijk (zoals bij tekstobjecten) worden hier CSS properties getoond van het geselecteerde item. Deze gegevens kun je kopiëren en plakken naar je eigen CSS-bestand. Vergeet niet de code op te schonen en onnodige code weg te halen.

College of Multimedia

* Deze tools zijn onmisbaar voor webdesigners. Zorg dat je goed oefent!

- * Pen Tool (P)
- * Add Anchor Point Tool (+)
- * Delete Anchor Point Tool (–)
- * Anchor Point Tool (Shift+C)

- * Type Tool (T)
- Area Type Tool
- Type on a Path Tool
- Vertical Type Tool
- Vertical Area Type Tool
- Vertical Type on a Path Tool
- Touch Type Tool (Shift+T)

- * Rectangle Tool (M)
- * Rounded Rectangle Tool
- * Ellipse Tool (L)
- * Polygon Tool
- * Star Tool
- Flare Tool

- * Shaper Tool (Shift+N)
- * Pencil Tool (N)
- * Smooth Tool
- Path Eraser Tool
- Join Tool

- * Rotate Tool (R)
- Reflect Tool (O)

- * Width Tool (Shift+W)
- * Warp Tool (Shift+R)
- Twirl Tool
- Pucker Tool
- Bloat Tool
- Scallop Tool
- Crystallize Tool
- Wrinkle Tool

- * Shape Builder Tool (Shift+M)
- Live Paint Bucket (K)
- Live Paint Selection Tool (Shift+L)

- * Eyedropper Tool (I)
- Measure Tool

- Symbol Sprayer Tool (Shift+S)
- Symbol Shifter Tool
- Symbol Scruncher Tool
- Symbol Sizer Tool
- Symbol Spinner Tool
- Symbol Stainer Tool
- Symbol Screener Tool
- Symbol Styler Tool

- * Hand Tool (H)
- Print Tiling Tool

- * Direct Selection Tool (A)
- Group Selection Tool

- * Line Segment Tool (\)
- Arc Tool
- Spiral Tool
- Rectangular Grid Tool
- Polar Grid Tool

- Paintbrush Tool (B)
- Blob Brush Tool (Shift+B)

- Eraser Tool (Shift+E)
- Scissors Tool (C)
- Knife

- Scale Tool (S)
- Shear Tool
- Reshape Tool

- Puppet Warp Tool
- * Free Transform Tool (E)

- Perspective Grid Tool (Shift+P)
- Perspective Selection Tool (Shift+V)

- Column Graph Tool (J)
- Stacked Column Graph Tool
- Bar Graph Tool
- Stacked Bar Graph Tool
- Line Graph Tool
- Area Graph Tool
- Scatter Graph Tool
- Pie Graph Tool
- Radar Graph Tool

- Slice Tool (Shift+K)
- Slice Selection Tool

Adobe Illustrator CC - Lagen

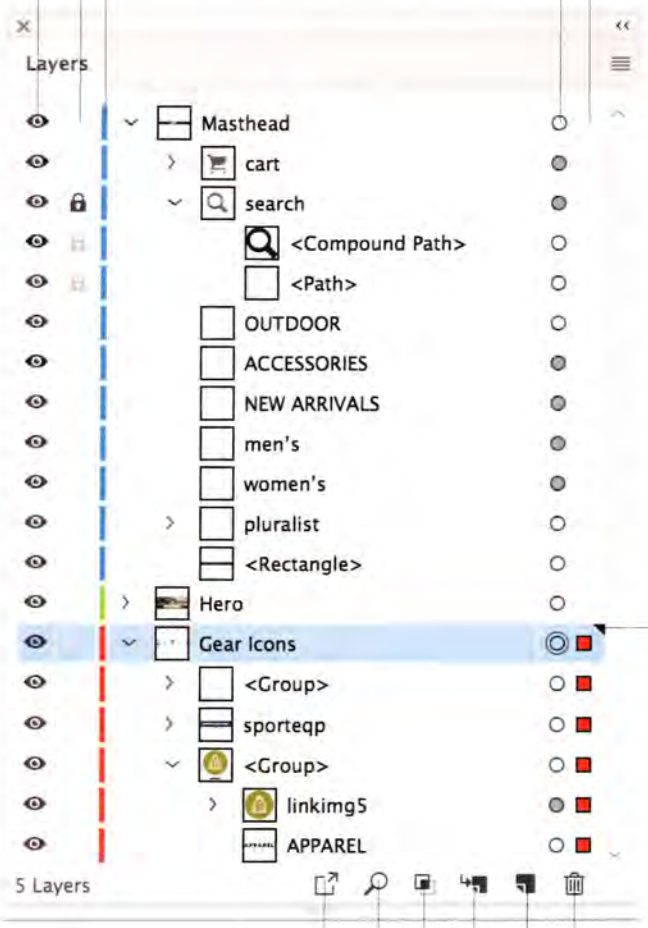
Layer color: Instelbaar via het hamburger menu - Options for "laagnaam"...

Edit column (lock / unlock): een laag (tijdelijk) beschermen tegen het editen met een slotje.

Visibility column: Verberg of toon lagen.

Target column

Selection column



Current Layer Indicator.

Delete Selection: De geslecteerde lagen (aangegeven met de blauwe balk) weggooien.

Create New Layer: Option-klik dit icoon voor meer opties.

Create New Sublayer: Voor het aanmaken van geneste lagen.

Collect for Export: De selectie klaarzetten in de Asset Export panel om daarna snel te kunnen exporten.

Locate Object: Laat Illustrator de laag aanwijzen in de layer panel, horende bij het object dat je hebt aangeklikt in de canvas. Dit kan helpen de laagstructuur beter te begrijpen.

Make / Release Clipping Mask: Stop twee objecten in één laag. Selecteer daarna de laag en klik op dit icoon. Er wordt nu een clipping mask aangemaakt. Het ene object in de laag dient als masker (bovenliggende laag) voor de ander (onderliggende laag).

Scalable Vector Graphics (SVG's)

Het bestandsformaat .svg is een W3C standaard, volledig open source en het enige geschikte vectorbestand voor webpublicatie.

Een SVG is een tekstbestand, waarin vectorinformatie staat omschreven met de XML-syntax. XML en SVG hebben veel overeenkomsten met HTML. De syntax is vergelijkbaar en er zijn tags en attributes die structuur en gegevens omschrijven. XML is echter niet beperkt tot een vooraf gedefinieerde set tags; je kunt je eigen tags en attributen verzinnen.

Adobe Illustrator kan vectorbestanden, zoals .pdf, .eps en .ai, omzetten naar web optimized svg's.

Een SVG met een artboard van 64 X 64 pixels, met daarin een cirkel (rode fill, zwarte stroke):

```
<?xml version="1.0" encoding="utf-8"?>
<svg version="1.1" xmlns="http://www.w3.org/2000/
svg" xmlns:xlink="http://www.w3.org/1999/xlink"
viewBox="0 0 64 64">
    <circle fill="#FF0000" cx="32" cy="32" r="31.5"
    stroke="black" stroke-width="1" />
</svg>
```

Voorbereidingen in Illustrator

Voordat je in Illustrator artwork exporteert naar .svg, moet je het document eerst opruimen en optimaliseren. Maak altijd een back-up van het origineel *vòòrdat* je optimaliseert!

1. Verwijderen ongebruikte lagen en haal onnodige objecten weg.

2. Vereenvoudig artwork door het aantal anchor points in artwork terug te brengen. Ga naar Object - Path - Simplify...

3. Tekstlagen moeten worden omgezet naar outlines (Type - Create Outlines). De vormgeving van teksten is nu op alle devices gegarandeerd, omdat er in de SVG geen typografische informatie of verwijzingen naar externe fonts worden ingesloten.

4. SVG's zijn vectorbestanden. Zet daarom alle bitmaps om naar vectoren met de Image Trace functie van Illustrator. Haal bitmaps weg als dit niet mogelijk is.

5. Als je met de Pathfinder panel werkt, zorg dan dat gaten in je artwork ook echt gaten zijn en niet objecten met dezelfde kleur als de achtergrond.

6. Hernoem alle lagen en objecten. Deze stap is alleen belangrijk als je naderhand de SVG-code handmatig wilt bewerken of als je de SVG interactief wilt maken met JavaScript.

7. Selecteer de Artboard-tool. In de Properties panel, selecteer bij 'Artboards Preset' voor 'Fit to Artboard Bounds'.

8. Exporteer de SVG.

Exporteren in Illustrator

Illustrator kent verschillende technieken en mogelijkheden:

- **File - Save As - Format: SVG**
  Deze mogelijkheid bevat de meeste exportsettings. Kies in het save as dialoog *nooit* voor SVG Compressed (svgz). Deze bestanden zijn niet compatible met browsers.

- **File - Export - Export As... - Format: SVG**
  Deze mogelijkheid biedt alle noodzakelijke exportsettings en is iets eenvoudiger.

  De 'Minify' optie is erg handig omdat het alle white spaces uit de SVG-code haalt. Dit scheelt soms tientallen kilobytes per svg!

- **File - Export - Export for Screens...**
  Kies voor deze mogelijkheid als je meerdere artboarts tegelijk wilt exporteren, of objecten hebt klaargezet met de 'Asset Export' panel (aanrader!).

  Helaas heb je met deze optie minder controle over hoe de uiteindelijke SVG-code wordt.

Illustrator exportsettings (File - Save As - Format: svg).

SVG Options

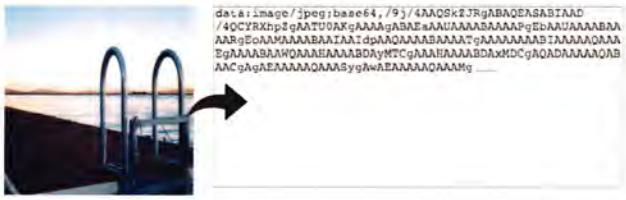
Het kiezen van de juist options bij het exporteren is cruciaal voor een compatible en web optimized svg (zie afbeelding op de vorige pagina)_:

- **SVG Profile: SVG 1.1:** Kies SVG 1.1 omdat dit de W3C recommendation is. SVG Tiny was oorspronkelijk bedoeld voor de eerste generatie Blackberry telefoons en PDA's.

- **Type:** Convert to outline. Je hebt tijdens je voorbereidingen alle tekstlagen reeds omgezet naar outlines. Eventueel kun je hiermee vergeten tekstlagen alsnog omzetten van glyphs naar vectorpaden.

- **Image Location:** Zorg ervoor dat je geen bitmaps in je artwork hebt voordat je gaat exporteren naar SVG. Deze setting is dan ook niet relevant.

  Eventuele bitmapinformatie wordt met 'Embed' in de SVG weggeschreven als Base64-code (binary to text encoding). De 'Link' optie maakt een external reference naar de bitmap.



Voorbeeld van een jpeg omgezet naar Base64 code.

- **Preserve Illustrator Editing Capabilities**: *NOOIT* aanvinken! Dit zorgt voor problemen bij complexe SVG's.

- **CSS Properties:** Presentation Attributes is de aangeraden optie omdat het de kleinste SVG's maakt. Exportvoorbeeld:

  ```
  <path fill="#f00"></path>
  ```

  Style Attributes (Entity Reference): Deze optie rendert niet goed in sommige browsers. Exportvoorbeeld:

  ```
  <path style="&st0;"></path>
  ```

  Style Attributes kan soms grotere SVG's geven dan bij presentation attributes, omdat stijlinformatie per tag inline gezet wordt. Exportvoorbeeld:

  ```
  <pathstyle="fill: #f00;"></path>
  ```

  Style Elements geeft elk onderdeel in de SVG een class-name en voorziet de SVG van embedded CSS-rules. Deze optie gebruik je als je met JavaScript de svg wilt manipuleren. Deze optie geeft meestal grotere bestanden dan presentation attributes. Exportvoorbeeld:

  ```
  <style>.st0 { ... }</style>
  <path class="st0"></path>
  ```

- **Decimal Places:** Bepaal hoe nauwkeurig anchorpoint posities worden vastgelegd. Meestal is één of twee decimalen voldoende om een vorm nauwkeurig te kunnen omschrijven. Kies meer decimalen als text outlines niet mooi rond worden.

- **Encoding:** Uiteraard kies je UTF-8.

- **Responsive:** Uiteraard wil je dat de SVG correct schaalt in de browser.

RTS-game: They Are Billions

Voorbeelden van SVG's.

FutureWave (1995 - 1996)

Adobe Animate kent een lange geschiedenis. Midden jaren 90 lanceerde het bedrijf 'FutureWave Software' het teken-programma 'SmartSketch'. SmartSketch was zijn tijd ver vooruit en kwam met een peninterface en een eigen besturingssysteem.

FutureWave Software zag potentie in webanimaties gebaseerd op vector graphics. SmartSketch werd daarom aangepast met frame-by-frame animatiemogelijkheden en werd opnieuw in de markt gezet als FutureSplash Animator. Het pakket moest concurreren met de Shockwave technologie van Macromedia.

Eind 1996 werd FutureWave Software overgenomen door Macromedia en werd de software hernoemt naar Macromedia Flash. Saillant detail: Adobe deed ook een bod op de technologie, maar FutureWave Software sloeg dat af.

FutureSplash Animator: de voorloper van Adobe Animate. De timeline is na al die jaren nog steeds herkenbaar.

Macromedia Flash (1996 - 1999)

In de periode 1996 - 1999 is Flash veranderd van een animatieplatform naar een webplatform. De Flash authoring environment produceerde .swf-bestanden, die in de webpagina werden ingesloten met embed- en object-tags. De browser gebruikte vervolgens de Flash plug-in om het project af te spelen.

Flash veroorzaakte een ware revolutie. Voor het eerst in de geschiedenis van het internet was het mogelijk om complexe interactieve multimedia applicaties in de browser te draaien, zoals games met audio en video.

ActionScript (2000)

Flash werkte met de object geörienteerde programmeertaal ActionScript. Het is een ECMAScript dialect en lijkt daarom op JavaScript.

Tegenwoordig schrijf je in Adobe Animate met ActionScript 3 (gelanceerd in 2007) complete Android en iOS apps.

Macromedia Flash 8 (2000 - 2005)

Flash 8 is de laatste grote versie van Macromedia. Het bracht diverse grafische updates, zoals filters (blur, drop shadows, etc), blending modes en FLV-video.

Flash Video was een belangrijke ontwikkeling voor het internet en heeft bijgedragen aan de populariteit van Flash.

In de periode 2000 - 2005 bleef Flash buitengewoon populair. Wereldwijd was het geïnstalleerd op 95% tot 98% van de computers!

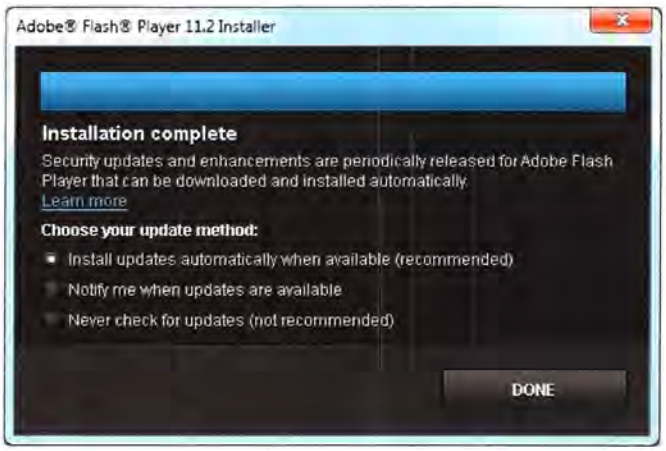
Adobe (2005 - 2007)

In 2005 werd Macromedia overgenomen door Adobe. Door de overname kreeg Adobe een aantal zeer populaire softwarepakketten in handen zoals Dreamweaver, Flash en Director (voor het maken van applicaties op CD-ROM).

Ze hadden hiermee werkelijk een goudmijn in handen. Illustrator, Encore (voor DVD authoring), FireWorks (het maken van webgraphics) werkte allemaal intensief samen met Flash.

Flash leek niet te stoppen, maar flashwebsites waren ook slecht toegankelijk, scoorden niet goed in zoekmachines, hadden lange laadtijden en een matige UX door irritante splash screens en onzinnige introducties die niet waren over te slaan.

Adobe werd na de overname van Macromedia zelf-ingenomen. De Flash plug-in kwam steeds vaker negatief in het nieuws door problemen op het gebied van veiligheid, bedrijfszekerheid en browsercompatibiliteit.



De vele updates gaven frustratie, waardoor mensen hun computer slecht of helemaal niet meer bijwerkten: koren op de molen van hackers!

Apple iPhone (2007)

Adobe kreeg het tij echt tegen met de komst van de Apple iPhone.

In 2007 lanceerde Apple de iPhone samen met het iOS besturingssysteem. De smartphone werd in eerste instantie bekritiseerd omdat het geen Flash ondersteunde.

In 2010 publiceerde de CEO van Apple, Steve jobs, het document "Thoughts on Flash", waarin hij uitlegde waarom Apple de razend populaire Flash plug-in niet ondersteunde.

De vele security updates waren klantonvriendelijk, de plug-in nam kostbaar geheugen in beslag en het afspelen van .swf-bestanden was intensief voor de mobiele processor, waardoor het ook nog eens een aanslag was op de batterii.

Er was nog een ander argument voor Apple. Flash vormde een directe bedreiging voor iOS en de App Store!

De smartphone bleek razend populair, waardoor de publieke houding tegenover Flash snel veranderde. Het was geen nadeel dat de iPhone geen Flash afspeelde, maar een nadeel dat websites geen iPhone ondersteunde!

Microsoft SilverLight (2007)

Microsoft kwam in 2007 met een Flash concurrent: het Silverlight platform. Met Silverlight kon, je net als met Flash, games en multimedia- en videoplatformen schrijven. Het is korte tijd intensief gebruikt door o.a. de KNVB en Netflix.

Microsoft Silverlight

Angular & React (2011)

Door de snelle ontwikkeling van HTML5, CSS3 en JavaScript is Flash technisch overbodig geworden:

- *Animaties worden gemaakt met JavaScript en de HTML5* canvas-tag.
- Video's spelen in de browser direct af zonder trage plug-ins.
- JavaScript frameworks (zoals Angular en React) staan tegenwoordig centraal bij het ontwikkelen van webapps.
- Adobe zelf heeft in 2011 aangekondigd te stoppen met Flash om zich te richten op het ontwikkelen van HTML5-tools.

YouTube en Edge Animate (2015)

In 2015 stopte YouTube met Flash Video ondersteuning.

Adobe had naast Flash een ander animatieprogramma: Edge Animate. Met Edge Animate maak je animaties met HTML5, CSS en JavaScript technieken. Er is dus geen Flash plug-in nodig om ze af te spelen.

Ondanks de vele creatieve beperkingen die CSS-gebaseerde animaties hebben ten opzichte van Flash, bleek Edge Animate zeer populair bij webdevelopers. Des te groter was de verbazing toen Adobe in 2015 ineens stopte met Edge Animate! Het idee was Edge Animate te integreren met Flash, maar daar zien we vandaag de dag erg weinig van terug.

Adobe Animate (2017)

Door de dalende populariteit van Flash en door de integratieplannen met Edge Animate heeft Adobe het pakket hernoemt naar Animate. In feite is Animate nog gewoon Flash met een andere naam.

Het verschil zit hem nu in de output van het programma. Het leunt al een tijdje niet meer op de verouderde Flash technieken, maar op de moderne HTML5 canvas-tag en het open-source createJS framework.

Een voorbeeld hoe JavaScript een circel tekent in een canvas tag:

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95,50,40,0,2*Math.PI);
ctx.stroke();
```

Adobe Animate is dus een nog altijd een uitstekende tool voor het maken van interactieve (reclame-)banners, complex geanimeerde hero images, webgames en iOS / Android apps. Ook kun je er ook volledige tekenfilms mee maken, zoals Bob's Burgers van Comedy Central!

Adobe Animate CC - Interface

1) Properties panel.
Vergelijkbaar met de Photoshop option bar en de Illustrator properties panel. De panel verandert op basis van context; zoals de actieve tool of de actieve selectie.

2) Stage.
De canvas / workarea van Animate heet de Stage. Net als bij theater is de Stage het zichtbare gedeelte dat de gebruiker ziet. Het bevat de 'actors' zoals teksen, afbeeldingen en video.

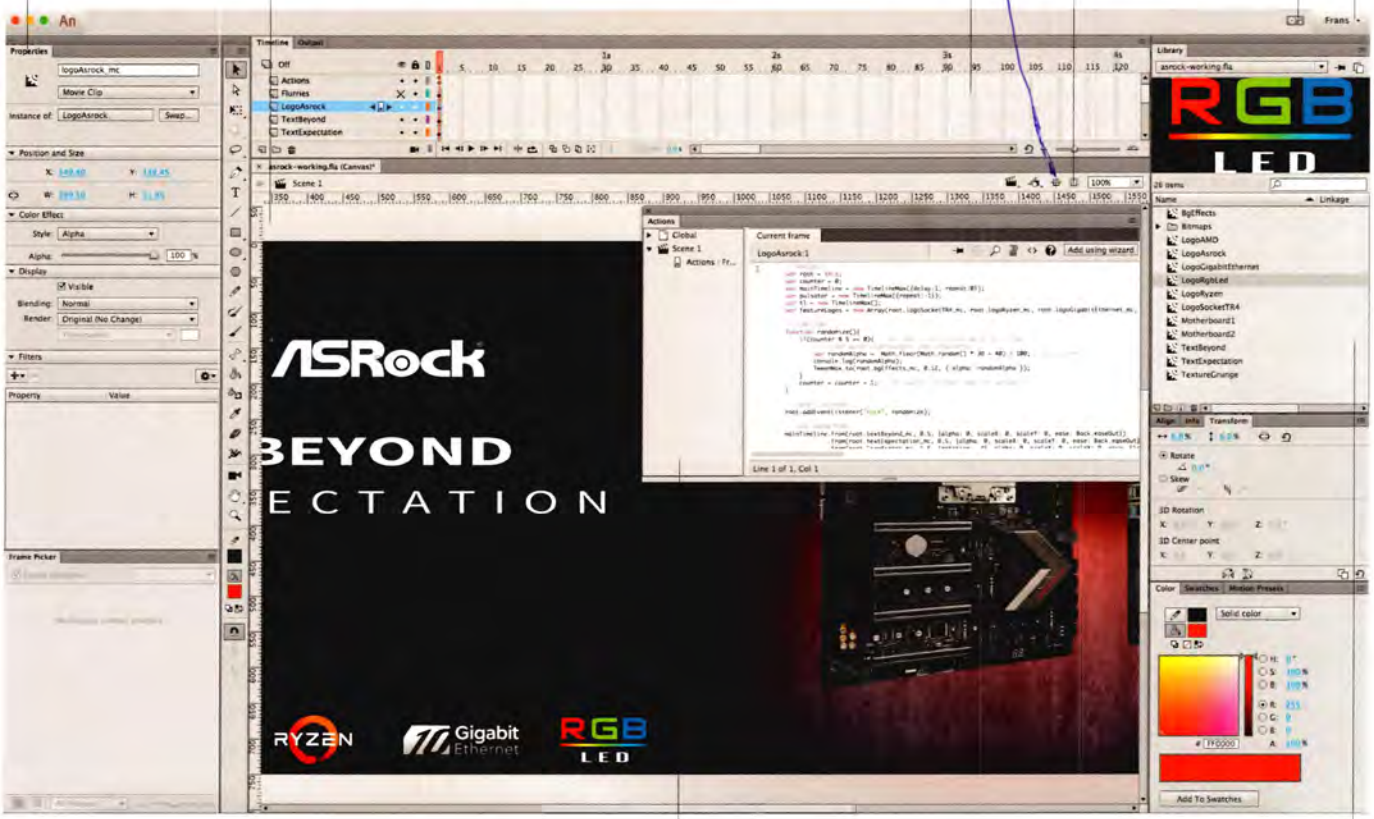
3) Clip The Content Outside Stage.
Bepaal of beeldelementen buiten de Stage zichtbaar zijn. Dit is alleen bedoelt als authoring tool in de authoring environment. In de run-time environment zijn deze elementen niet zichtbaar.

center stage

4) Timeline.

5) Creative Cloud Sync Settings.

6) Het Workspace menu.



7) Action panel.
Schrijf hier JavaScript of ActionScript code. Het linker gedeelte heet de Script Navigator. Animate plaatst code in timeline keyframes, dus dit gedeelte is erg handig als code over verschillende keyframes en timelines is verdeeld. Het rechter gedeelte heet de Script Pane. Schrijf hier je code.

8) Library panel.
Met de library panel organiseer je assets (zoals bitmaps, symbols, video en audio) in mappen. Het laat zien hoe vaak je een bepaalde asset in je project gebruikt.

Animate projecten zijn self-contained: assets worden bij het importeren fysiek gekopiërd naar de library. Er is dus geen verwijzing naar een bestand op de harde schijf.

- EXE X
- APP
- HTML ..... HTML5 CANVAS
- AIR
- Apk
- ISW
- SWF X

"Zwakke"
&
"Sterke" Stappen (?)

problem publish settings:
✓ Hosted libraries

URL? ... = Force
Hard Refresh

TINY PNG
TINY JPG

13) Empty keyframe.
Een leeg bolletje geeft aan dat er niets op de stage staat voor deze laag.

14) Keyframe.
Een keyframe geeft een verandering van content op de stage weer. Een zwart bolletje is de beginkeyframe van een frame span. Een diamandje is ook een keyframe, maar nooit de eerste van een tween span. Een keyframe laat niet zien voor welke property of properties de verandering geldt.

Dubbel klik daarom op de tween span (of rechter muisklik - Refine Tween) om de motion editor te openen.

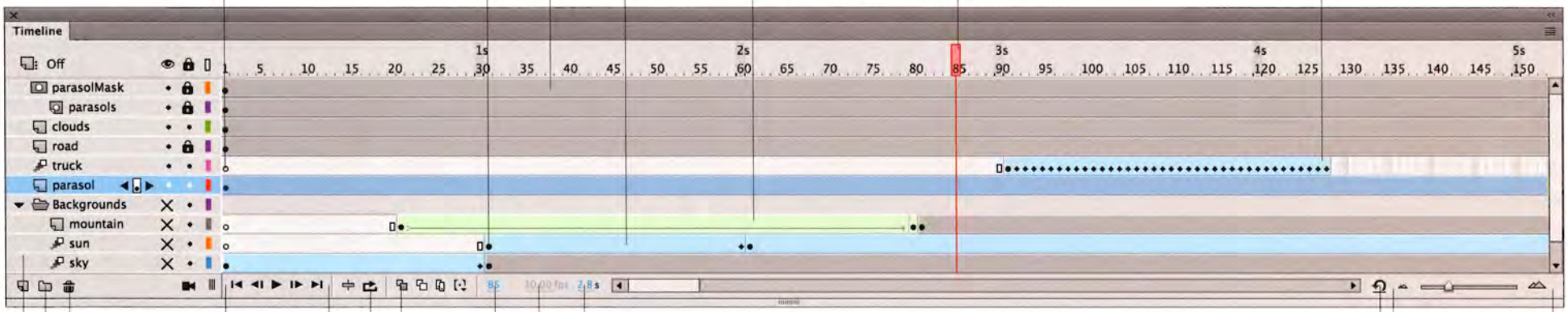
1) Playhead.
Bij het afspelen van het project beweegt de rode verticale playhead over de frames in de timeline. Drag & drop de label van de playhead om de content van een bepaald frame te tonen.

15) Normale frame span.

16) Motion tween frame span (tween span genoemd)

17) Shape tween span.

18) Frame by frame animatie.



4) Delete Layer.

3) New Folder.

2) New Layer.

7) Union skinning.

6) Loop.

10) Elapsed time (in seconden).
In de authoring environment kan een project meestal niet real-time worden afgespeeld. De elapsed time is een handig hlaat zien hoeveel tijd er normaal gesproken gepasseerd is ter hoogte van de playhead positie.

12) Resize timeline view.

11) Resize timeline zoom to the default level

5) Playback Controls.
Van links naar rechts:
- Go to first frame (shift ,),
- Step back one frame (,),
- Play (enter),
- Step forward one frame (.),
- Go to last frame (shift .)

9) Frame rate.
De frame rate van het project, gemeten in frames per second (fps). Het is hier niet mogelijk om de frame rate aan te passen, omdat dit de gehele timing van het project veranderd.

8) Current frame.
Het nummer van het huidige frame dat nu wordt getoond. Je kunt hierop dubbel klikken of scrubby scrollen.

Nota Bene!

Voor canvasprojecten adviseert Adobe het gebruik van Classic Tweens in plaats van Motion Tweens! Bij het publiceren van je project worden Motion Tweens door de CreateJS library omgeschreven naar frame-by-frame animaties. JavaScript-bestanden van complexe projecten kunnen hierdoor onnodig groot worden!

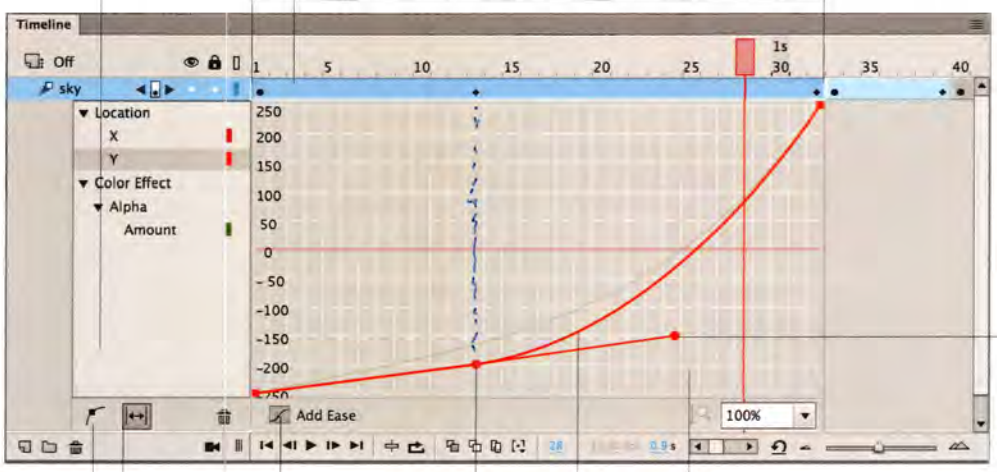
In tegenstelling tot Motion Tweens worden Classic Tweens bij het publiceren omgeschreven naar optimale CSS-keyframe animaties. Helaas is er geen motion editor beschikbaar voor Classic Tweens, maar de property panel biedt wel uitkomst.

1) Motion Editor panel.
Gebruik de Motion Editor bij het maken van complexe animaties. Dubbel klik op een tween span om de motion editor te openen (of rechter muisklik op een tween span en kies voor 'Refine Tween'). Er is altijd maar één Motion Editor panel zichtbaar.

2) Tween Span.
Motion tweens worden getoond als blauwe balk. Shape tweens zijn groen van kleur.

3) Tween Span scheidingslijn.
Click & drag de scheidingslijn (aan het begin of einde van een tween span) om het start- of eind-punt van een tween span aan te passen. Keyframes worden in verhouding mee verplaatst.

Shift-click & drag op een scheidingslijn om de duur aan te passen, zonder hierbij keyframes te verplaatsen.



10) Direction lines.
De hoek van twee direction lines uit een keyframe bepalen grotendeels het karakter van de animatie.

Een haakse property curve voor en achter een keyframe betekent een abrupte verandering.

Een vloeiende property curve door een keyframe betekent een geleidelijke verandering.

11) Apply ease for selected property.
Easing toevoegen zonder keyframes te zetten of direction lines te tekenen.

7) Remove tween for selected property.
De animatie van de geselecteerde property wissen.

Voor de Location property betekent dit dat je zowel de X- en Y-keyframes weghaalt.

4) Motion Editor graph.
De grafiek laat overzichtelijk zien welke properties over tijd veranderen en op welke manier. In dit voorbeeld zijn er keyframes voor Location (X en Y pixelposities) en voor de Alpha Color Effect (Amount %).

De Y-property is geselecteerd en zichtbaar als property curve.

8) Restore view.
De breedte van de motion graph aanpassen.

5) Property Curve.
De verandering over tijd van een property wordt getoond als een uniek gekleurde lijn (de property curve).

Bij het aanklikken van een property, in de lijst aan de linkerkant van de grafiek, licht de betreffende property curve op en worden bijhorende keyframes zichtbaar.

6) Keyframe.
Een blokje op de property curve betekent een keyframe. Merk op dat de keyframe ook in de tween span bovenin is afgebeeld als zwart bolletje (de beginkeyframe van een frame span) of als zwart diamandje. Een diamandje is ook een keyframe, maar nooit de eerste van een tween span.

9) Add anchor point on graph.
Je kunt met een pentoolachtig gereedschap property curves wijzigen. Met deze knop voeg je een keyframe toe op de positie van de playhead.
- Command-klik op een keyframe om het weg te halen.
- Option-klik op een keyframe om de direction lines te wissen.
- Option-klik & drag op een keyframe om een nieuwe direction line uit de keyframe te draggen.

Shapes & Objecten				
Eigenschap	**Merge Shape**	**Drawing Object**	**Primitive**	**Group**
Type vector shape?	Zwak, met een losse fill en stroke.	Sterk (soort groep)	Sterk (soort groep)	nvt
Smelten twee objecten samen?	Ja	Nee	Nee	Nee
Vrij te vervormen?	Ja	Ja	Nee	Nee
Ronde hoeken?	Nee	Nee	Ja, achteraf aan te passen	Nee

Symbols & Instances			
Eigenschap	**Graphic Symbol**	**Movieclip**	**Button**
Herbruikbaar?	Ja	Ja	Ja
Tijdlijn?	Ja, maar gekoppeld aan de parent's tijdlijn.	Ja, volledige onafhankelijke tijdlijn.	Ja, met vier frames voor de button states: up, over, down en hit.
Instance Name (ID)?	Nee, niet aan te spreken met scripts.	Ja, aan te spreken met scripts.	Ja, aan te spreken met scripts.
Filters of Blending modes mogelijk?	Nee	Ja	Ja
Overigen?	Heeft diverse looping options.		Krijgt automatisch een mouse-pointer.

Code Snippets		
Method	**Code voorbeeld**	**Opmerking**
.stop()	this.bal_mc.stop();	De timeline van de instance genaamd 'bal_mc' moet stoppen met afspelen. 'this' refereert naar de timeline waar 'bal_mc' opstaat. Dit kan de stage's timeline zijn, maar ook een timeline van andere symbol (nested timeline).
.play()	this.bal_mc.play();	De timeline van de instance genaamd 'bal_mc' moet afspelen.
.gotoAndPlay()	this.bal_mc.gotoAndPlay(5);	Verplaatst de playhead van 'bal_mc' naar het 6de frame, en begin met afspelen. (!)
.gotoAndStop()	this.bal_mc.gotoAndStop("chapter 2");	Verplaatst de playhead van 'bal_mc' naar het frame met het framelabel "chapter 2" en stop daar.

(!) Let op:

CreateJS tijdlijnen beginnen op frame #0 en niet op #1! Bij het publiceren van je Canvasproject geeft de output panel de volgende melding:

```
"Frame numbers in EaselJS start at 0 instead of 1. For example, this affects gotoAndStop and gotoAnd-
Play calls. (2)"
```

'this.bal_mc.gotoAndPlay(5)' verplaatst de playhead van de bal_mc instance naar het *zesde* frame. Het eerste frame is logischerwijs frame #0.

ActionScript 3.0
ActionScript is een objectmatige programmeertaal van Adobe voor ActionScript- en AIR-projecten. De taal is vergelijkbaar met JavaScript. Voor canvas-projecten schrijf je geen ActionScript, maar JavaScript.

Actors
De objecten op de stage worden soms ook wel eens actors (acteurs) genoemd; een verwijzing naar het theater.

AIR
Rond 2009 werd Adobe AIR gelanceerd. Dit is een cross-platform runtime applicatie, waarmee Flashprojecten buiten de browser als stand-alone programma kunnen draaien op Apple, PC, BlackBerry OS, Android, iOS en Linux (Linux ondersteuning is gestopt).

Authoring Environment
Een authoring environment is software waarmee je andere software schrijft. Voorbeelden zijn Animate voor het ontwikkelen van HTML5 Canvasprojecten, Dreamweaver voor het ontwikkelen van websites en Apple Xcode voor het ontwikkelen van iOS apps.

CreateJS
CreateJS is een verzameling van vier JavaScript libraries. Adobe Animate publiceert je project naar HTML, CSS en JavaScript. De JavaScript is gebaseerd op CreateJS:

1. **EaselJS** - een API om makkelijker met de HML5 Canvas-tag te kunnen werken.
2. **TweenJS** - een library voor het tweenen en animeren van HTML en CSS properties. Deze library kan zonder EaselJS stand-alone worden gebruikt.
3. **SoundJS** - een API om te werken met geluid in de browser.
4. **PreloadJS** - een library voor het laden van assets en data.

Drawing Object
Één van de tekenobjecten van Animate. Het drawing object is in feite een soort groep om de losse fill en stroke heen. Net als de merge shape is het vrij te vervormen.

Easing
Een term voor animaties die langzaam beginnen en daarna versnellen (de 'ease in') en voor animaties die snel beginnen en langzaam vertragen (de 'ease out').

Frame
Een frame is één beeldje van een animatie. Animaties spelen voor het oog vloeiend af wanneer er tenminste 15 à 20 frames per seconde zijn. De frequentie noemen we frames per second (fps).

GreenSock Animation Platform (GSAP)
Een eenvoudige JavaScript library voor het maken van complexe (timeline) animaties. Het is een veel gebruikt alternatief voor TweenJS.

Met het GSAP-platform gebruik je JavaScript code voor het beschrijven van de animatie. In Animate hoef je dan ook niet meer te werken met het Timeline panel.

GSAP kan uiteraard ook buiten Adobe Animate worden gebruikt door HTML-elementen en CSS-eigenschappen aan te spreken met JavaScript.

Group
Met de group kun je meerdere objecten op de stage samenbundelen. Er wordt een soort onzichtbare schil om de objecten gezet.

Group icon in de property panel.

Verwar de group niet met een laag. In één laag kunnen meerdere groups voorkomen, maar andersom is natuurlijk dat niet mogelijk. Groepeer lagen met timeline folders.

Instance
Maak een instance door een library symbol naar de stage te slepen. De instance is een kopie (of clone) van de betreffende symbol.

Er blijft een link bestaan tussen symbol en instance. Wanneer de symbol verandert, veranderen ook alle instances op de stage.

Eigenschappen van een instance. De property panel toont tevens de band tussen instance en symbol.

Instances hebben onderling hun eigen eigenschappen, zoals x,y, width, height, scaleX, scaleY, rotation, alpha en filters. Zo kan elke instance een unieke set filters, blending modes en effecten krijgen.

Instance name: bal_mc MC = movie clip. BTN = button

Elk object heeft eigen timeline

Bewerk een symbol door op de stage te dubbelklikken op een instance.

Soms is het duidelijker de symbol in 'isolation mode' te bewerken. Dubbelklik op het icoon van de symbol in de library panel. Zo wordt je niet afgeleid door omliggende assets en artwork.

JavaScript

JavaScript maakt een webpagina interactief. Het is een client-side scripting taal. Dit betekent dat de browser (de client) JavaScript code ontvangt, interpreteert en daarna regel voor regel uitvoert.

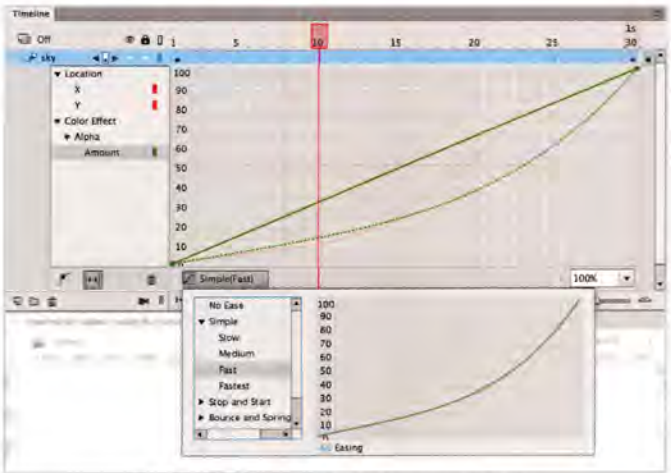
Keyframe

Een keyframe geeft een verandering van content op de stage weer. Het is te herkennen in de timeline panel als zwart bolletje (de startkeyframe) of diamandje (de keyframe van een motion tween).

Keyframe Interpolatie

Interpolatie is het invullen van een onbekende waarde tussen twee bekende waarden. Tussen twee keyframes wordt de waarde van een property geïnterpoleerd (zie tween frame).

Maak keyframe interpolaties door in de motion editor 'easing' in te stellen met de 'Apply ease for selected property' optie. Zo zijn complexe effecten mogelijk (zoals bounce effecten) *zonder* extra keyframes. De interpolatie wordt getoond als stippellijn.



De 'Apply ease for selected property' optie.
De stippellijn in de grafiek toont de keyframe interpolatie.

Lineaire animatie

De animate verloopt per frame gelijk en even snel. Animaties lijken hierdoor vaak minder natuurgetrouw dan animaties die over tijd versnellen of vertragen (zie easing en keyframe interpolatie).

Merge Shape

De merge shape (kortweg: shape) is het meest eenvoudige tekenobject van Animate. Het heeft een losse fill en stroke.

Het selectiepatroon van de shape.

De merge shape is te herkennen aan een karakteristiek stippelpatroon bij een actieve selectie.

Shape icon in de property panel.

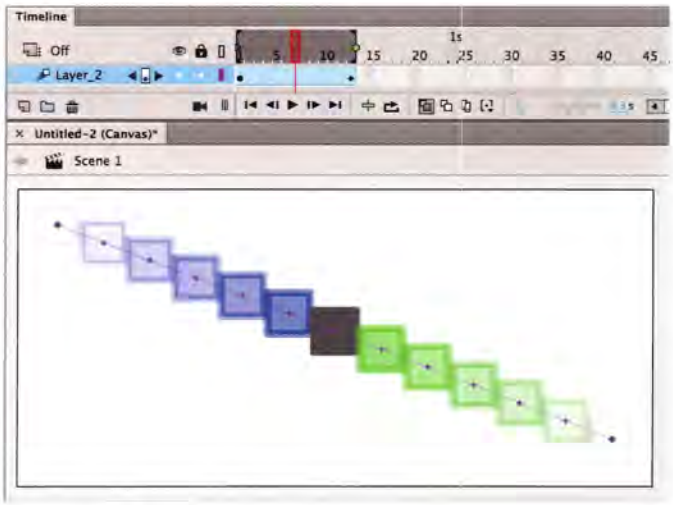
Twee merge shapes smelten samen tot één nieuwe shape als er geen actieve selectie is en ze op dezelfde laag over elkaar liggen.

Shapes zijn vrij te vervormen door te click & draggen op de zijkant van het object.

Onion Skinning

Animate toont normaal gesproken één frame tegelijk in de stage. Onion Skinning maakt het eenvoudiger een animatie te maken, omdat het meerdere frames tegelijk toont.

Op de stage wordt het frame onder de playhead normaal getoond. Frames in het verleden zijn blauw en toekomstige frames zijn groen.



Primitive (les)

De primitive is een soort drawing object. Het voordeel is dat bepaalde eigenschappen naderhand nog aan te passen zijn, zoals de hoeken van een rectangle primitive en cirkeleigenschappen van de oval primitive.

Primitve tools op de toolbar.

De primitive is in tegenstelling tot de merge shape en de drawing object niet vrij te vervormen.

MASKER (ONTOP)

ADD SHAPE hint van morph

Hit state: Extra figuur voor Hotspot

Script lagen altijd bovenaan (NAAM - Actions, Script, code)

HTML5 geen ovale gradients

Property

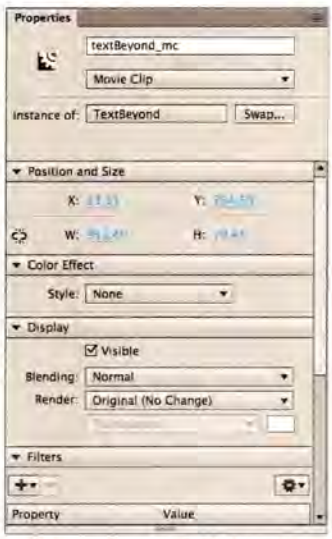
Een eigenschap van een instance of tekenobject op de stage.

Prototype

De JavaScript term voor een symbol is prototype of class.

Publishing

Het exportproces van Animate waarin HTML, CSS en JavaScript wordt weggeschreven naar de harde schijf.



Property panel van Animate.

Runtime Environment

De 'afspeelomgeving' (playback environment) van hetgene dat een authoring environment produceert.

Animate publiceert Canvasprojecten naar HTML5, CSS en JavaScript. De browser speelt het af en is dus de runtime environment.

Stage

Het zichtbare gebied van het project. In andere Adobe software noemen we dit gebied vaak de workarea of de canvas. De naam 'stage' (engels voor toneel) is geleend van het theater.

Objecten kunnen buiten de stage worden geplaatst. Ze zijn in dit geval niet zichtbaar in de runtime environment.

Symbol

Een symbol is een herbruikbaar object in de vorm van een graphic, button of movieclip. De symbol is herbruikbaar door er instances van te maken. Dit doe je door de symbol van library naar stage te slepen. De symbol is een soort moeder, kloon, blauwdruk of bouwtekening voor de instances.

Door symbols te gebruiken wordt de bestandsgrootte van het eindproject aanzienlijk verkleind. Het opslaan van meerdere instances neemt minder opslagruimte in beslag dan het maken van inhoudelijke kopieën van de symbol, omdat we refereren naar de eigenschappen van de symbol.

Symbol - Button

Gebruik buttons om interactieve knoppen te maken die reageren op muisklikken, rollovers of andere events.

Button icon

De button symbol heeft een speciale timeline met vier frames:

SVG

1. **Up** - de 'normal state' van de button.
2. **Over** - de 'mouseover state' van de button.
3. **Down** - de 'active of mousedown state' van de button.
4. **Hit** - de hitframe is niet zichtbaar op de stage. Het defineert het gedeelte van de knop dat reageert op de muis; handig als je button gaten heeft of qua vorm ongelijk is. Ook kun je onzichtbare knoppen maken: handig voor adventure games!

Labels van zero based index

Symbol - Graphic

Gebruik graphic symbols voor statische afbeeldingen en voor herbruikbare animaties. Graphic symbols hebben een eigen timeline, maar de playhead loopt gelijk met die van z'n parent's timeline.

Graphic icon

Je kunt graphic instances met de property panel geen eigen instance name geven. Dit betekent dat de instance niet via scripting is aan te spreken.

Vermijd 'Else' zoveel mogelijk

Symbol - MovieClip

MovieClips lijken op Graphic symbols, maar hebben een eigen onafhankelijke timeline. MovieClip instances kunnen met de property panel een eigen instance name krijgen. Dit betekent dat de instance via scripting is te benaderen.

MovieClip icon

MovieClips kunnen worden voorzien van diverse filtereffecten en blending modes, maar controleer of deze effecten wel compatible (en te animeren) zijn in canvasprojecten. Bij het publiceren van je project zal het output panel uitkomst bieden.

Tween frame (between frame)

Tussen twee keyframes zitten de 'between frames', ook wel tween frames genoemd. Zie keyframe interpolatie.

global script Z_setup

GSAP

Insert Frame (F5)

CDN

+SHIFT (select .....)

Guided paden niet kruisen.

post processing Lucas

HTML5: geen ovale gradients

JavaScript

Een webpagina die na het inladen niet meer verandert is 'statisch'. Een webpagina met JavaScript kan na het inladen wel veranderen en is 'dynamisch'.

JavaScript maakt webpagina's interactief: ze kunnen reageren op input van de gebruiker en op verschillende gebeurtenissen (events genoemd).

JavaScript maakt het mogelijk een webpagina te manipuleren door HTML-elementen toe te voegen, aan te passen en te verwijderen. Ook kan het de browser zelf manipuleren, zoals het aanmaken van een nieuw window en het veranderen van de breedte- en hoogte-eigenschappen van een browser window.

JavaScript

James Gosling is uitvinder van de programmeertaal Java, waarop JavaScript is gebaseerd. Hij kreeg het idee voor deze naam door de Java koffie in zijn hand!

Scripting

JavaScript is een client-side scriptingtaal. De browser (client) ontvangt het script, interpreteert het en voert de code direct regel voor regel uit. JavaScript is dus geen programmeertaal, omdat het niet wordt gecompileerd.

Voorbeelden van scriptingtalen zijn AppleScript (Apple), Coldfusion (Adobe), Perl, PHP, Python, Ruby en VBScript (Microsoft).

Compilen

De compiler is onderdeel van de authoring environment en verantwoordelijk voor het omzetten (compileren) van code. Tijdens het compileren wordt high-level source code omgezet naar binaire of low-level source-code. Het eindresultaat is af te spelen in de runtime environment.

- dblclick: de gebruiker dubbelklikt op een element
- keypress, keydown en keyup
- mouseover, mouseout, mousemove, mousedown en mouseup
- focus: een element wordt actief.
- blur: een element wordt inactief.
- change: een selectbox is aangepast of veranderd.
- load: de browser is klaar met het laden van een resource, het document of window.

Hoe computers 'denken'

Computers hebben geen idee van context en welke functies en mogelijkheden voorwerpen hebben.

Een trein, een boek of een tv: de computer heeft er geen idee van. Jij ziet een pen en begrijpt dat het is bedoeld om notities mee te maken en dat je moet opletten dat je geen vlekken maakt. Je weet dat als de pen niet werkt, de pen misschien wel leeg is. Een dier doet dit ook, maar dan op haar niveau. Een kat ziet de pen als speeltje of prooi.

Een computer kan dit soort complexe aannames niet maken. Het kan niet leren van zijn fouten, reageert niet spontaan op zijn omgeving, interpreteert zijn omgeving niet, is niet creatief en is niet in staat om te ervaren, te voelen en het heeft geen overlevingsdrang.

Zelfdenkende machines zijn science-fiction... Voor nu dan...

Voorkom dat je 'slimme' computersystemen menselijke eigenschappen toekent. De zelfrijdende auto lijkt misschien intelligent, maar voert alleen instructies uit in één bepaalde omgeving, omdat het hiervoor is geprogrammeerd.

Werken met modellen

Programmeurs maken zogenaamde modellen van gegevens en voorwerpen, zodat een computerprogramma er vervolgens iets mee kan doen. We vormen modellen met:

- Objects
- Instances & Prototypes
- Properties (variables)
- Methods (functions)
- Arguments
- Events

Objects

In een script wordt een fysiek voorwerp, zoals een webshopproduct, omschreven als object. Objecten kunnen ook abstracter zijn, zoals een bestelformulier.

Instances & Prototypes

Een webshop bevat natuurlijk niet één product, maar misschien wel honderd. Dit zijn de object instances.

Object instances zijn eigenlijk een soort kopie of kloon. Ze worden aangemaakt (geïnitialiseerd) door de object prototype (soms class genoemd): code dat dient als blauwdruk voor instances.

De prototype bepaalt welke eigenschappen (properties) instances kunnen aannemen, over welke acties en mogelijkheden (methods) ze kunnen beschikken en op wat voor gebeurtenissen ze kunnen reageren (events).

Zo is een bestelformulier een object instance van de 'form'-prototype. Het heeft HTML-attribute properties, reset en submit methods en events zoals onreset en onsubmit. Een 'form'-object bevat zelf ook objecten: een lijst met formelements. Tussen HTML-objecten is er dus een parent / child relatie.

Tijdens de Adobe Animate lessen heb je op een visuele manier leren werken met object prototypes (symbols) en object instances (de kopieën van de symbols op de stage).

Voorkom verwarring: een programmeur spreekt meestal gewoon over een object, en noemt het niet expliciet een object instance.

Variables

Variables zijn tijdelijke bewaarplaatsen voor gegevens (data). In JavaScript krijgt een variable eerst een naam (declaration), waarna gegevens (value) kunnen worden toegekend (assignment).

Properties

Properties zijn eigenschappen van een object, of te wel de variables van een object. Een property bestaat uit een name en value pair; vergelijkbaar met een CSS-declaration.

Functions

Een functie is een op zichzelf staand, herbruikbaar, ingekapseld stuk code voor het uitvoeren van een specifieke taak.

Functions kunnen gegevens ontvangen (in de vorm van argumenten) en gegevens teruggeven (in de vorm van de return value).

Methods

Methods zijn mogelijkheden van een object, of te wel de functions van een object.

Arguments

Arguments zijn gegevens die je meegeeft bij het aanroepen van een function of method.

Events

Events zijn gebeurtenissen waarop een object reageert, of te wel de actiemomenten waarop het object een method uitvoert of waardoor zijn properties wijzigt.

Animate
Bestandsnaam lib
caps

Instance naam: kleine letters
Eslint

2010:
Window.AddEventListener('load', fn
init

ES5 = goed. web
ES6+ (React, Angular, Vue)

Scripts ontwikkelen

Ervaren programmeurs kunnen een script intuïtief en uit het hoofd ontwikkelen.

Een beginnende programmeur ***moet*** eerst een plan maken door taken in stappen onder te verdelen, de stappen in detail uit te schrijven en daarna flowcharts en diagrammen op te stellen.

Scripts ontwikkelen heeft drie stappen:

1. Omschrijf het doel
2. Ontwikkel het script
3. Codeer elke stap van het script

1. Omschrijf het doel

Je wilt bijvoorbeeld met de auto boodschappen doen bij de Albert Heijn, zodat je vanavond een pastagerecht kunt koken.

2. Ontwikkel het script

Om het doel te bereiken moet vanaf een startpunt een aantal taken worden uitgevoerd. Één taak bestaat weer uit vele stappen, die uiteindelijk als JavaScript instructie worden opgeschreven.

Om ons boodschappendoel te bereiken moeten we ook een aantal taken in correcte volgorde uitvoeren:

Taakomschrijving:

1. Vaststellen welke producten we willen kopen.
2. Het huis verlaten en naar de auto gaan.
3. Naar de Albert Heijn rijden.
4. Bij de Albert Heijn boodschappen doen.
5. Naar huis rijden.
6. De auto verlaten en het huis binnen gaan.
7. Boodschappen opbergen.

Stappenplan van taak #1:

1. Naar de koelkast lopen
2. De koelkast openen
3. De inhoud inventariseren
4. Observeren welke producten we missen
5. Onthouden welke producten we moeten gaan kopen
6. De koelkast dicht doen
7. Naar de tafel lopen
8. Pen en papier oppakken van de tafel
9. Opschrijven welke producten we nodig hebben
10. Papier afscheuren
11. Papier in de broekzak stoppen

Het stappenplan is met opzet zo gedetaileerd mogelijk opgesteld, want hoe meer detail, hoe meer kans dat het programmeren een succes wordt.

> ***(!) Let op:***
> Voer je deze fase niet of slordig uit, dan loop je tijdens het scripten zeker vast. Leg jouw plan eerst voor aan de docent, *vóórdat* je gaat programmeren.

3. Codeer elke stap

Elke taak en elke stap wordt opgeschreven in de JavaScript syntax, zodat de browser het kan begrijpen.

programma structuur schetsen

Diagrammen

Ontwikkel scripts door het maken van diagrammen (het class diagram) en stroomdiagrammen (PSS en PSD).

programma structuur diagram

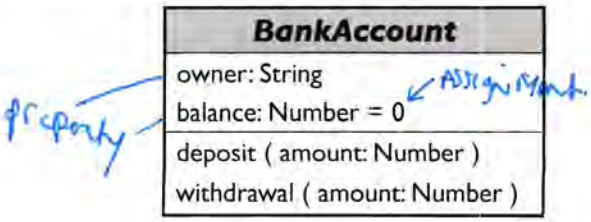
Unified Modeling Language (UML)

UML is in 1994 bedacht door Grady Booch, Ivar Jacobson en James Rumbaugh van Rational Software, om hun verschillende notatiesystemen te standaardiseren.

UML is ontwikkeld om het ontwerp van een systeem, zoals een computerprogramma, op één universele manier te visualiseren. Sinds 2005 is UML een ISO-standaard en wereldwijd uitgegroeid als de facto standaard voor programmeurs.

Class diagrams

Het class diagram is onderdeel van UML. Het omschrijft de structuur van een systeem (object) met zijn attributes (properties), activiteiten (methods) en onderlinge relaties tussen objecten.



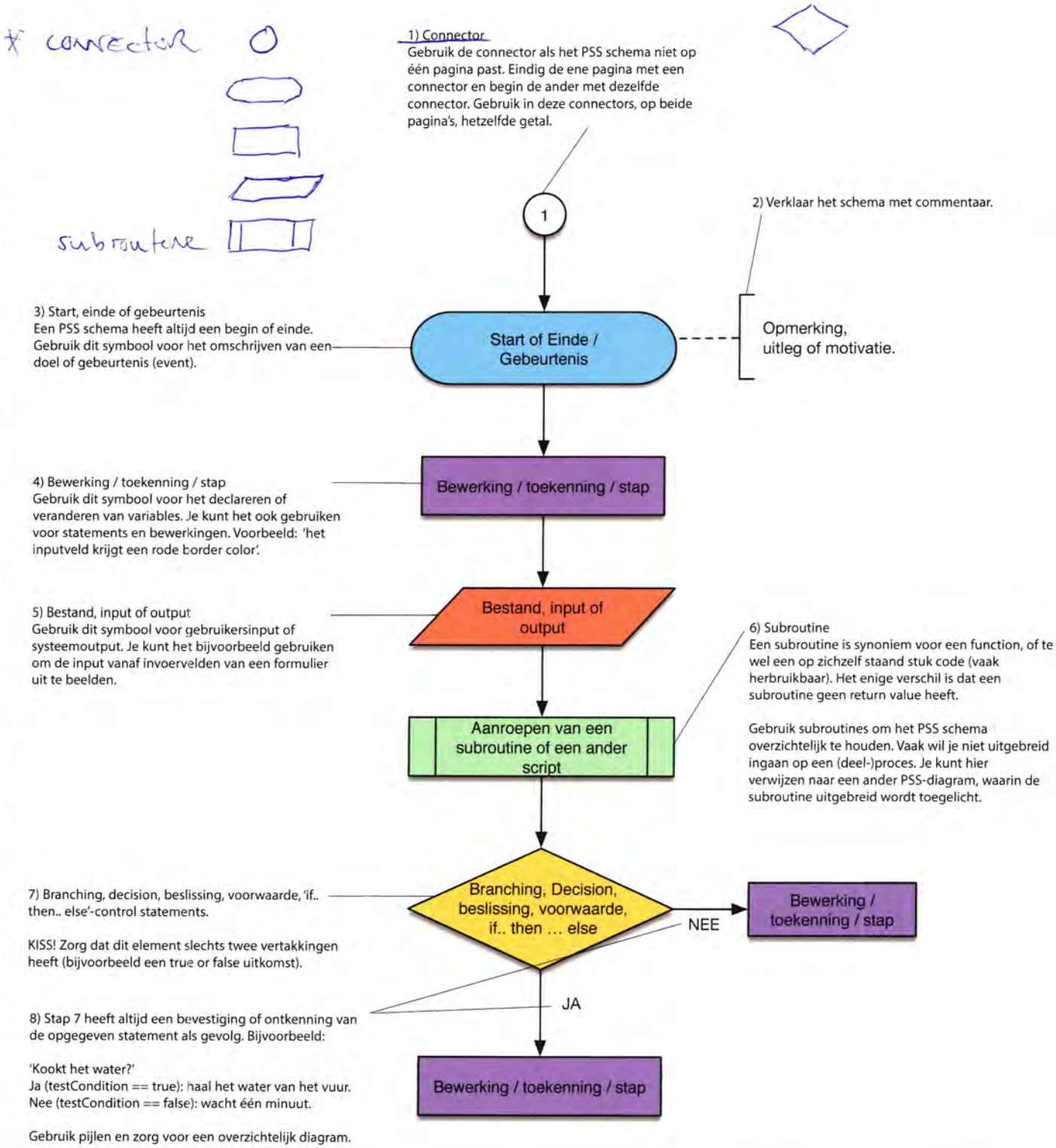
Dit is een eenvoudig class diagram voor een object genaamd 'BankAccount'.

Het object heeft twee properties: 'owner' van het type 'String' (zonder assignment) en 'balance' van het type 'Number', met de waarde '0'.

Het object heeft twee methods: 'withdrawal' en 'deposit' vereisen ieder één argument 'amount' van het type 'Number'.

Programma Stroom Schema (PSS)

PSS schema's worden soms flowchartschema's genoemd. Het is afgeleid van het 'Probabilistic Signature Scheme', ontwikkeld door Mihir Bellare en Phillip Rogaway.



(!) Let op:

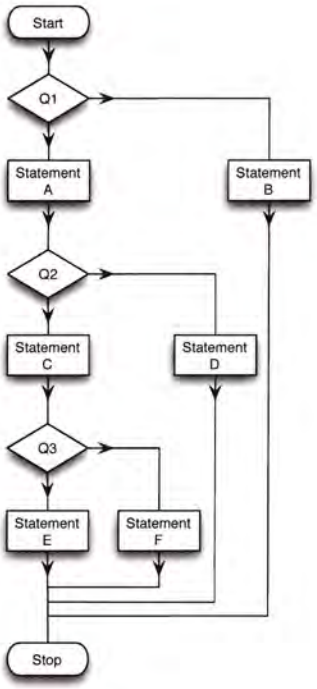
Je loopt tijdens het scripten zeker vast als je geen goede uitgebreide PSS schema's opstelt! Leg jouw plan eerst voor aan de docent, *vóórdat* je gaat programmeren.

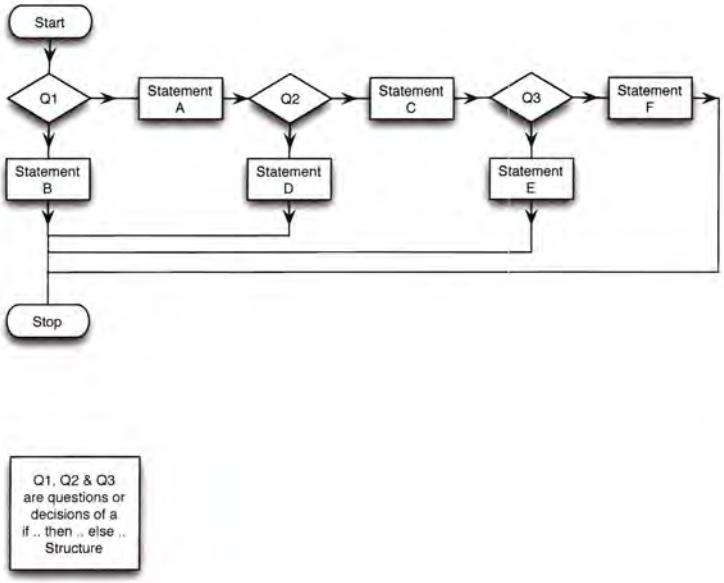
Nota Bene:

Het gebruik van kleur is niet noodzakelijk. De meeste PSS schema's worden zwart / wit getekend.

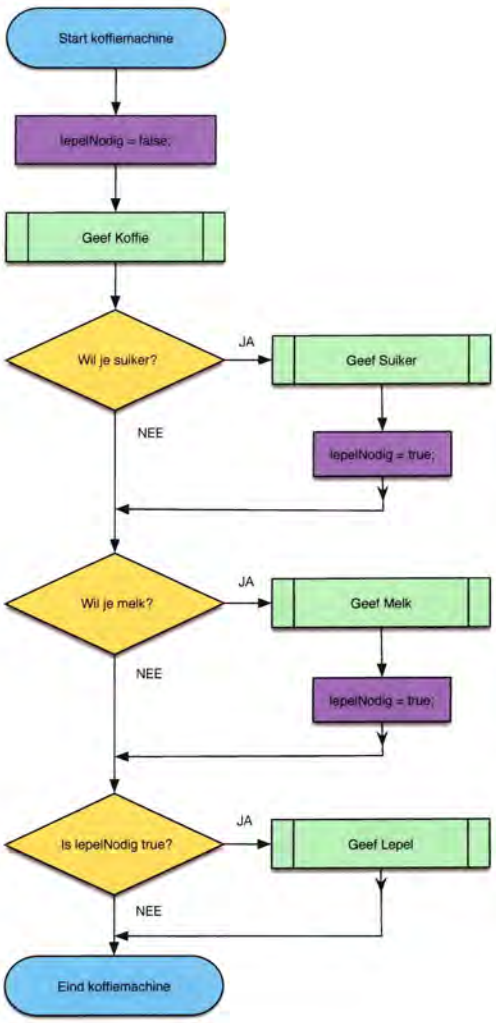
Do's and Don'ts

Een correct PSS-schema (linker voorbeeld) loopt van boven naar beneden. Het rechter voorbeeld laat zien hoe het *niet* moet.



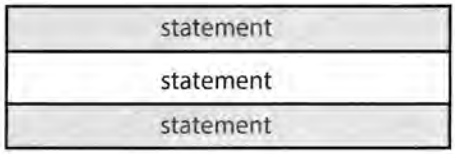


Voorbeeld

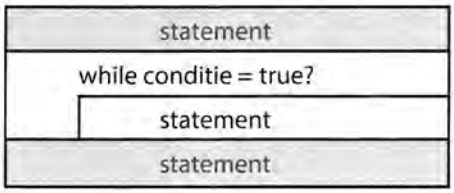


Programma Structuur Diagram (PSD)

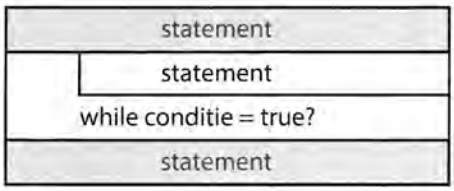
Het PSD-schema is bedacht door Isaac Nassi and Ben Shneiderman, die het in 1982 als studenten van SUNY-Stony Brook ontwikkelden. Door de eenvoudige notatie is de PSD sneller te maken en een goed alternatief voor PSS.



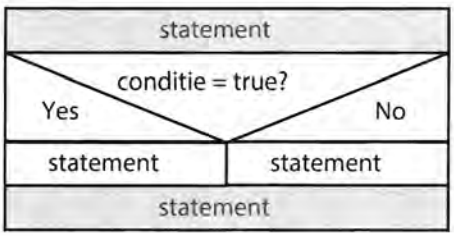
- **Statements** zijn opdrachten, commentaar of (sub-)routines. Ze staan op volgorde (sequentie). Voor de leesbaarheid worden de balken om en om gekleurd.



- **Iteraties** bevatten een conditie die bepaalt hoelang, of hoe vaak, er herhaald moet worden. De ingesprongen statement direct onder de conditie is de te herhalen opdracht. Dit voorbeeld is vergelijkbaar met 'while' of 'for' in JavaScript.



- **Iteratie**, maar nu is de controle van de conditie aan het einde van de loop. Dit voorbeeld is vergelijkbaar met 'do... while' in JavaScript.

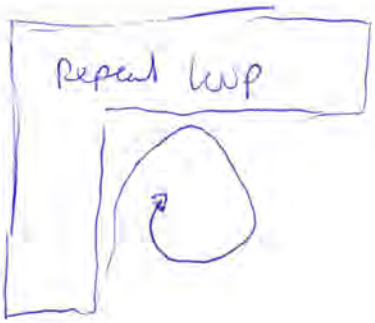


- **Selectie** is een keuzemoment. Het bevat een conditie die bepaalt of de statements onder ja óf onder nee worden afgehandeld. Dit voorbeeld is vergelijkbaar met 'if... else if' in JavaScript.

Afhankelijk van de situatie kunnen er verschillende aantallen statements onder 'ja' of 'nee' komen te staan. Zorg er dan voor dat het diagram overzichtelijk blijft!

Voorbeeld

```
function koffie() {
      geef(koffie);
      var lepelNodig = false;

      if (suiker){
            geef(suiker);
            lepelNodig = true;
      }

      if (melk){
            geef(melk);
            lepelNodig = true;
      }

      if (lepelNodig){
            geef(lepel);
      }
      drink(koffie);
}
```



Statements

Statements zijn instructies voor de webbrowser, geschreven in de JavaScript syntax. Een goede syntax is belangrijk, anders kan de browser de code niet goed interpreteren. Een statement is vergelijkbaar met een Nederlandse zin; het zijn de bouwblokken van je code.

Een statement kan op een eigen regel worden gezet:

```
eerste statement
tweede statement
```

Het is ook mogelijk meerdere statements op één regel te zetten. Gebruik dan de puntkomma (statement terminator) om een statement af te sluiten:

```
eerste statement; tweede statement;
```

In JavaScript is het gebruikelijk alle statements af te sluiten met de puntkomma en ze op een eigen regel te zetten. De code is zo optimaal te lezen en minder foutgevoelig.

Commentaar

Net als bij HTML en CSS moet code worden voorzien van commentaar. Commentaar is belangrijk in JavaScript, omdat het moeilijk te lezen is. Er zijn drie manieren om commentaar toe te voegen aan JavaScript:

```
// Dit is een single-line commentaar.

/* Dit commentaar
   bestaat uit meerdere regels. */

<!- HTML-achtig single line commentaar...
<!- ...in de praktijk gebruikt niemand deze methode!
```

Variables

In het leven staan sommige gegevens vast, zoals je naam en geboortedatum. Andere gegevens kunnen in de tussentijd veranderen, zoals je humeur of leeftijd. Variables zijn tijdelijke bewaarplaatsen voor dit soort gegevens (data).

Een variable 'humeur' heeft de waarde 'opgewerkt'. De variable 'leeftijd' heeft de waarde '24'. Aan de hand van gebeurtenissen of acties kunnen deze variables veranderen. Het winnen van de loterij verandert 'humeur' in 'blij' en de verjaardag verandert 'leeftijd' in '25',

Variables lijken qua syntax een beetje op CSS-declarations. Ze bestaan uit een name en een value (property en value in CSS).

Het aanmaken van een variable noemen we 'declaration'. Het toekennen van een waarde aan de nieuwe variable is de 'assignment'. Variables aanmaken doe je in een statement met een reserved keyword genaamd 'var':

```
var humeur = "blij";
var leeftijd = 24;
```

In veel programmeertalen moet je eerst een variable declareren, vóórdat je een waarde toekent. Met JavaScript maakt dat niet uit:

```
// Declareren, maar nog niet toekennen.
var humeur,leeftijd;

// Declareren én direct toekennen met een waarde.
var humeur = "blij";
var leeftijd = 24;

// Declareren en toekennen, in één statement.
var humeur = "blij", leeftijd = 24;

// Éérst declareren en daarna toekennen.
var humeur, leeftijd;
humeur = "blij";
leeftijd = 24;

/* Een variable aanmaken en daarna updaten met een
nieuwe waarde (re-assignment). */
var humeur = "blij";
humeur = "boos";
```

Variables do's en don'ts

- Geef een variable altijd een korte duidelijk naam.
- Toegestane karakters: a-z, A-Z, 0-9 en underscore '_'.
- Niet toegestaan: leestekens, accenten en spaties.
- Niet beginnen met een cijfer! ~~9naam~~; naam9
- In veel talen (bijv. CSS en JavaScript) zijn de namen van variables en id's 'case-sensitive' dus 'Naam', 'NAAM' en 'naam' zijn verschillend.
- Als de naam van een variable uit meerdere woorden bestaat gebruik dan de camelcase methode. Bijvoorbeeld 'customerName'.
- Overweeg voor de variable name het gebruik van een 'g' als het een globale variabele getreft: gWoonplaats.

Data types

Variables kunnen verschillende soorten waarden bevatten (data types), zoals:

- **Number:** positieve of negatieve hele getallen (integer numbers) en decimaal getallen (floating point numbers).
- **Boolean:** kan twee waarden hebben; óf true óf false.
- **String:** een serie karacters: 'a string of characters'.
- **Array:** een lijst met één of meerdere variables (feitelijk is de array een object).

"10" – "13" = –3

string

- **Object:** een object instance.
- **Null:** een niet-bestaande of ongeldige waarde.
- **Undefined:** de waarde die variables hebben na declaratie, maar vóór assignment.

Strictly typed en Loosely typed (les)

JavaScript is een 'loosely typed' of te wel een 'weakly typed' programmeertaal. Als het gaat om variables is een 'weak typing language' flexibel en minder strict dan een 'strong typing language'. Het betekent dat de variable tijdens zijn levensduur mag veranderen van data type:

```
var leeftijd = 19; // number data type.
leeftijd = "twintig"; // re-assignment naar string.
```

Tevens hoef je tijdens het aanmaken van een variable niet van te voren te bepalen wat het data type de variable is.

Expressions

Een expression is een javaScript statement, waarin een waarde wordt verwerkt of verrekent (results in). Dit noemen we evalueren (evaluate).

```
var color = "red"; // Een waarde toekennen.

var oppervlakte = 5 * 10; // Een som uitrekenen en
het resultaat opslaan in de var 'oppervlakte'.

if(1 == 2) // Twee waarden met elkaar vergelijken.
Deze expression geeft als antwoord 'false.
```

Expressions geven altijd een waarde terug. De waarde die een expression teruggeeft is afhankelijk van de 'operator'.

Operands

Je gebruikt operands om twee waarden of expressions met elkaar te vergelijken:

```
var groterOfKleiner = (5 + 10) < (11 + 12); // true
```

De operand is in dit voorbeeld het 'kleiner dan'-teken. De haakjes zijn er voor de leesbaarheid. Er staat:

1. is de uitkomst van 5 + 10 kleiner dan de uitkomst van 11+12?
2. Sla het antwoord van deze vraag op in de variable 'groterOfKleiner'.

Het antwoord op deze expression is 'true'. De nieuwe toegekende waarde van de variable 'groterOfKleiner' is dus de boolean 'true'.

```
var score1 = 90;
var score2 = 70;
var highscore1 = 100;
var highscore2 = 55;

if((score1 + score2) > (highscore1 + highscore2))
```

"10" + "13" = 101

Er staat: 'is het totaal van score1 opgeteld met score2 groter dan het totaal van highscore1 opgeteld met highscore2? De uitkomst van deze expression is de boolean 'true', want 160 is groter dan 155.

```
var winkelmandje = 12.4;
var gratisVerzendkostenVanaf = 25;

var rechtOpGratisVerzending = winkelmandje >=
gratisVerzendkostenVanaf;
```

De variabele 'rechtOpGratisVerzending' is na het evalueren van de expressie 'false'.

Operators

Expressions gebruiken operators om van meerdere waarden, één waarde te maken:

- Assignment operator
- Arithmic operators
- Comparison operators
- Logical operators
- String operators

Assignment operator (les)

- **=** Assignment (toekennen van een waarde):
  `var getal=5;`

Arithmic operators (les)

- **( )** Grouping. Het groeperen van expressions voor het sturen van 'operator precedance'.
- **+** Addition. Optellen.
- **-** Subtraction. Een getal aftrekken van een ander (5-2) of een getal negatief maken (-5).
- ***** Multiplication. Vermenigvuldigen.
- **/** Division. Delen.
- **++** Increment. Een getal ophogen met één.
- **--** Decrement. Een getal verlagen met één.
- **%** Modulo. Wat overblijft na het delen:
  13 % 2 = 1 want 12 / 2 = 6, rest 1.
  13 % 5 = 3 want 10 / 5 = 2, rest 3.

Let op 'operator precedence'. Delen en vermenigvuldigen gaan vòòr optellen en aftrekken. Gebruik haakjes ( ) om verwarring te voorkomen en om code beter leesbaar te maken.

* / %
+ –

les: String object

Comparison operators (les)

- **==** Equals to. Het vergelijken van twee waarden met elkaar (comparison).
- **===** Strictly equals to. Het vergelijken van twee waarden én het vergelijken van data type (equality). Number, string, etc
- **!=** Not equal. Controleren of een waarde niet gelijk is aan een andere waarden.
- **!==** Strictly not equal to. Controleren of een waarde niet gelijk is aan een andere waarden én vergelijken of ze hetzelfde data type zijn.
- **<** Less than. Kleiner dan.
- **<=** Less than or equal to. Kleiner of gelijk aan.
- **>** Greater than. Groter dan.
- **>=** Greater or equal to. Groter of gelijk aan.

Type coercion → definitie

JavaScript is 'loosely typed' of te wel 'weakly typed'. Dit kan problemen geven wanneer twee verschillende data types met elkaar worden vergeleken.

Bij het vergelijken of controleren van variables antwoordt JavaScript met 'true' of 'false', omdat JavaScript elke waarde ziet als 'true' of 'false'.

=== false
== true

Truthy value

Een 'truthy value' is een variable met een waarde die JavaScript behandelt *alsof* ze 'true' zijn: \ indirect true

```
if(true)
if(1) // het getal 'één'.
if(10/5) // een berekening.
if('string') // een string.
if('0') // een nul geschreven als string geschreven.
if('false') // false, als string geschreven.
if('true') // true, als string geschreven.
```

Falsy

Een 'falsy value' is een variable met een waarde die JavaScript behandelt *alsof* ze 'false' zijn: \ Ind. false

```
if(false)
if(0) // het getal 'nul'.
if('') // een lege string.
if(10/'score') // NaN (Not a Number)
if(highScore) // Een variable zonder assignment.
```

Equality

Type coercion kan verwarrende effecten hebben als twee waarden met elkaar worden vergeleken met de 'equals to' operator (==).

Gebruik daarom de 'strictly equals to' operator (===), zodat je ook de data types van de twee waarden vergelijkt:

Falsy

```
if(0 == false) // true!
/* JavaScript ziet beide waarden als falsy. */

if(0 === false) // false!
/* Het is 'true' dat beide falsy values zijn, maar
ze zijn verschillend qua data type! */
```

Logical Operators (les)

- **&&** Logical AND. Zijn beide waarden true? Return dan true. Als één van de twee false is, return dan false.
- **||** Logical OR. Is één van de twee waarden true? Return dan true. Anders, return false.
- **!** Not: negates an expression. Of te wel: true en false omdraaien.

```
// Voorbeelden van logical AND
if(true && true) // true!
if(true && false) // false!
if(false && true) // false!
if(false && false) // false!

// Voorbeelden van logical OR
if(true || true) // true!
if(true || false) // true!
if(false || true) // true!
if(false || false) // false!

var trueOrFalse = (2 < 5) && (3 >= 2); // true!

!(2<1) // Is 2 NIET kleiner dan 1? ... true!
!true // Not true is false!
!false // Not false is true!

if('3' !== 3) // true!
if('3' !== '3') // false!
```

Short circuit evaluation (les)

Logical expressions worden van links naar rechts gelezen. Als de eerste uitkomst / conditie voldoende informatie verschaft, wordt de tweede uitkomst / conditie niet uitgevoerd.

```
if(false && ietsAnders)
/* Er is direct false gevonden. Daarom kan de
uitkomst van deze 'logical AND expression' nooit
true zijn. De waarde van 'ietsAnders doet er dus
niet meer toe. De tweede waarde wordt daarom ook
niet gecontroleerd. */

if(true || ietsAnders)
/* Er is direct true gevonden. Daarom is de
uitkomst van deze 'logical OR expression' true. De
waarde van 'ietsAnders doet er dus niet meer toe.
De tweede waarde wordt daarom ook niet
gecontroleerd. */
```

String operators → concatenatie

- **+** Twee strings samenvoegen tot één nieuwe.
- **" of '** Markeert het begin of einde van een string.

```
var bedrijfsnaam = "College "+'of '+"Multimedia";
// College of Multimedia
```

goed oefenen !!

Arrays (les)

Een array in een variable die meerdere waarden kan opslaan. De 'array literal' is de aangeraden methode om een array aan te maken:

```
var boodschappen = [];
// Een lege array aanmaken.

var boodschappen = ["sla", "melk", "eieren"];
// Direct assignment. De array voorzien van drie strings.
```

Bron: Head First jQuery (O'Reilly).

- Een array voorzien van gegevens noemen we 'to populate the array'.
- Elementen in de array noemen we 'items'.
- Een item mag één van de data types zijn: strings, numbers, objects óf arrays!
- Je mag arrays populaten met items van verschillende data types. We noemen de array dan een 'mixed array'.
- Elke array item heeft een eigen positie in de array: de 'key' genoemd. Omdat deze positie met een getal wordt aangegeven, noemen we het ook wel de 'index key'. Als in object: key / name
- Computersystemen beginnen te tellen met het getal 0 (en niet met 1). Het eerste item in de array is daarom op te vragen met index key 0.

```
boodschappen[0];
// Het eerste item in de array opvragen.
```

Array properties en methods

Elke variable is in JavaScript in feite een object instance. De array is hierop geen uitzondering. Er zijn dus array properties en methods.

Zorg ervoor dat je bekent bent met de belangrijkste array properties en methods. Een volledige lijst vind je op w3schools.com en op de MDN web docs (developer.mozilla.org).

Numbers en strings hebben trouwens ook erg handige properties en methods. Een goede programmeur is tenminste op de hoogte van de verschillende mogelijkheden.

Array constructor / Literal

De array constructor is een alternatieve manier om een array aan te maken. De array is net als alle andere variables in feite een object instance. Je kunt daarom gebruik maken van de array object constructor om een array aan te maken:

cap

```
var mijnArray = new Array(2,3,4);
var nogEenArray = new Array("hallo", "iedereen");
```

Deze methode heeft een verwarrend effect. Als je bij het aanmaken van de array slechts één integer number opgeeft (bijvoorbeeld 10), dan krijgt de array 10 lege plaatsen, niet het getal tien op index 0! → Bug?

```
var mijnArray = new Array(10);
// Een lege array met 10 lege items!

var mijnArray = [10];
/* De array literal kent deze verwarring niet. We maken nu een array met 1 item (number met de value 10). */

var mijnArray = new Array(10, 11);
/* Een array aanmaken met twee items: number 10 op index key 0 en number 11 op index key 1. */

var mijnArray = [10, 11];
/* Hetzelfde, maar nu aangemaakt met de array literal. */
```

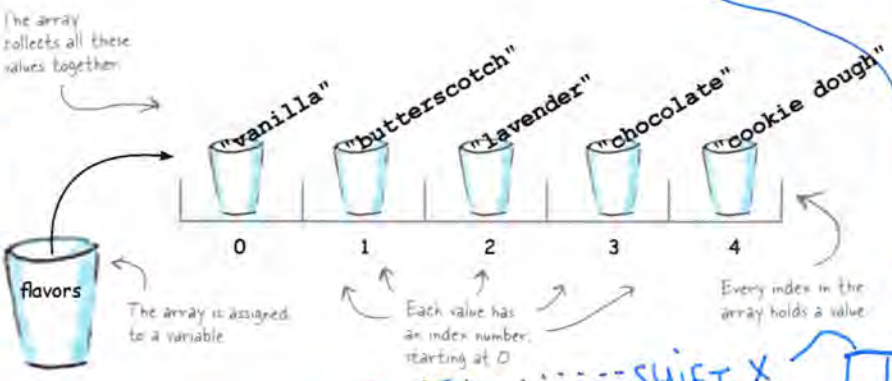
Array properties: .length;
Array methods: .pop();

Functie

eigenschap.

Array properties:
- .length
- prototype
- constructor

methods:



Bron: Head First jQuery (O'Reilly).

SHIFT() X

"SHIFT out"

UNSHIFT

X POP()

push()

Control Statements

In een script zijn er momenten dat er beslissingen worden gemaakt. De uitkomst bepaalt wat voor code wordt uitgevoerd.

Zie control statements als een splitsing in weg. In situatie A ga je linksaf en in situatie B ga je rechtsaf. Control statements worden ook conditional brancing, conditional logic of 'if.. else.. then'-statements genoemd. Een control statement bestaat uit twee onderdelen:

1. Een expression die moet worden geëvaleerd. Een waarde wordt vergeleken of gecontroleerd met een ander. De expression geeft een waarde terug (bijvoorbeeld true of false).

2. De control statement geeft aan welke code er aan de hand van de uitkomst van de expression moet worden uitgevoerd. Deze code staat tussen twee curly braces (code block genoemd).

Single if then zonder { }

```
if(condition == true){
   // Doe iets
} else if(anotherCondition == true){
   // Doe iets anders
} else{
   // Doe nog iets anders
}
```

Ternary logic (les)

Ternary logic is hetzelfde als een 'if.. else'-statement, maar dan korter geschreven:

```
var leeftijd = 39;
var toegang;

if(leeftijd >= 18){
   toegang = true;
} else {
   toegang = false;
}

// Nu korter genoteerd met ternary logic:
var toegang = leeftijd >= 18 ? true : false;
```

+ geneste TERNARY

Hierboven staat: Is de leeftijd groter of gelijk aan 18? Return dan true, anders return false. De uitkomst van deze expression wordt opgevangen in de variable 'toegang'.

Switch Statements (les)

Bij veel condities wordt de 'if... else' statement onoverzichtelijk. Gebruik dan een switch-statement.

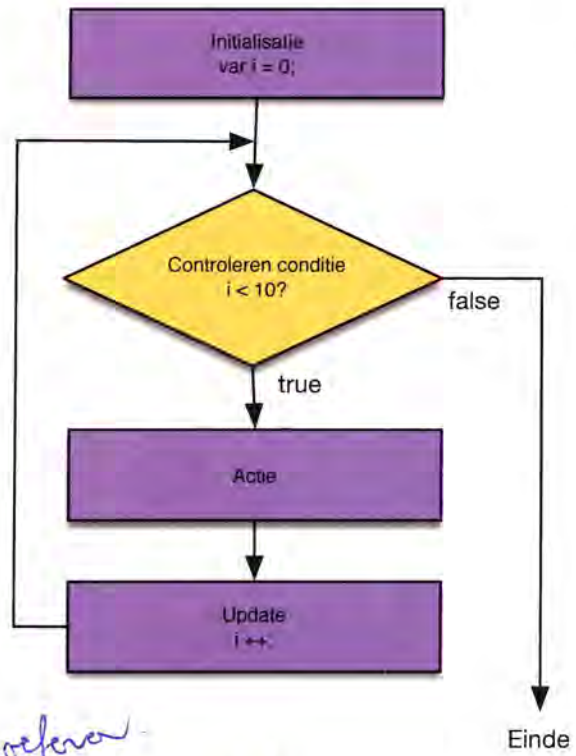
Het 'switch'-keyword geeft een waarde op die de 'case'-keywords moet testen (de condities). Gebruik het 'break'-keyword om uit de switch-statement te stappen. Het 'default'-keyword voorziet de switch van een 'fall-back'-situatie, indien geen van de cases true blijkt.

```
var melding;
var seizoen = "lente";

switch (seizoen){
   case "lente":
      melding = "Ga lammetjes kijken!";
      break;

   case "zomer":
      melding = "Het wordt een hete zomer!";
      break;

   case "herfst":
      melding = "Vergeet je regenjas niet!";
      break;

   case "winter":
      melding = "Dat wordt sneeuwballen gooien!";
      break;

   default:
      melding = "Geniet van een mooie dag!";
      break;
}
```

Loops & Iterations

Een 'loop'-statement controleert een conditie. Als de conditie true blijkt, wordt een code block uitgevoerd.

Daarna wordt de conditie nogmaals gecontroleerd en als het nog steeds true blijkt, wordt het uitvoeren van het code block herhaalt (itereren), totdat de conditie false blijkt.



referen
* Math & Date

JSDoc syntax /**

Onderdelen van een loop

Er zijn drie 'loop'-statements: 'while', 'do... while' en 'for'. Ze bestaan uit vier onderdelen:

1. **Initialiseren:** het begin van de loop. Je maakt een variable aan (vaak 'i' genoemd: kort voor iteration) die dient als teller. VAR i = 0;
2. **Conditie:** controleert of de loop door moet gaan of moet stoppen. Dit doe je door de 'i'-variable te controleren. i <= 10;
3. **Actie:** de code die steeds weer moet worden uitgevoerd als de conditie true blijkt. Console.log
4. **Update:** de 'i'-variable. Deze stap is erg belangrijk omdat een 'infinite-loop' moet worden voorkomen! Er moet altijd een conditie ontstaan zodat de loop kan eindigen. i++

While (les)

```
var i = 1; // initialisatie

while (i < 10){ // conditie
    console.log(i); // actie
    i++; // update
}
```

Do... While (les)

De do-while-loop lijkt op de while-loop. Het voert minimaal één keer de actie uit. De statements van het code block staat vòòr de condition.

```
var i=0; // initialisatie
do{
    console.log(i); // actie
    i++; // update
} while(i<=10); // conditie
```

For (les)

mag meerdere initialiseren/updaten

```
// initialisatie, conditie, update
for(var i=0; i<10; i++){
    console.log(i); // actie
}
```

i++, j++

De for-loop wordt veel gebruikt om door arrays heen te lo(o)pen. We gebruiken de array length-property als onderdeel van de conditie om te bepalen hoe vaak we moeten 'loopen'. In het actiegedeelte gebruiken we de i-variable als array index key.

```
var array = ["aap", "noot", "mies"];
for(var i=0; i<array.length; i++){
    // voor elk item in de array...
    console.log(array[i]); // console log het item.
}
```

Reserved Keywords

JavaScript heeft een aantal gereserveerde woorden die je niet mag gebruiken als variable- en function-naam. De dikgedrukte woorden moet je goed kennen.

abstract , arguments, await, boolean, **break**, byte, **case**, catch, char, class, const, **continue**, debugger, **default**, delete, **do**, double, **else**, enum, export, extends, **false**, final, finally, float, **for**, **function**, goto, **if**, implements, import, in, instanceof, int, interface, let, long, **of**, native, **new**, null, package, private, protected, public, **return**, short, static, super, **switch**, synchronized, **this**, throw, throws, transient, **true**, try, **typeof**, **var**, void, volitile, **while**, with, yield.

JavaScript Objects, Properties en Methods

Voorkom ook het gebruik van de build-in objects, properties en methods van JavaScript:

hasOwnProperty, Infinity, isFinite, isNaN, isPrototypeOf, length, Math, NaN, name, Number, Object, prototype, String, toString, undefined, valueOf.

Andere gereserveerde woorden

Je kunt ook beter de HTML namen en de window objects en properties niet gebruiken:

alert, all, anchor, anchors, area, assign, blur, button, checkbox, clearInterval, clearTimeout, clientInformation, close, closed, confirm, constructor, crypto, decodeURI, decodeURIComponent, defaultStatus, document, element, elements, embed, embeds, encodeURI, encodeURIComponent, escape, event, fileUpload, focus, form, forms, frame, innerHeight, innerWidth, layer, layers, link, location, mimeTypes, navigate, navigator, frames, frameRate, hidden, history, image, images, offscreenBuffering, open, opener, option, outerHeight, outerWidth, packages, pageX-Offset, pageYOffset, parent, parseFloat, parseInt, password, pkcs11, plugin, prompt, propertyIsEnum, radio, reset, screenX, screenY, scroll, secure, select, self, setInterval, setTimeout, status, submit, taint, text, textarea, top, unescape, untaint, window.

BABel, React (?)
Typescript, ANGULAR (?) } ES 6+

Functions

Een function is herbruikbare code met een eigen naam, dat apart staat (en functioneert) van de overige code. Goede functions zijn in meerdere situaties / projecten te gebruiken; ze maken code modulair en overzichtelijker.

Eerst maak een functie aan (function declaration) en codeer vervolgens wat de functie moet doen.

Nadat de functie is gedeclareerd kun je het overal in je script zo vaak als je wilt aanroepen (function call of te wel 'invoking a function' genoemd). Bij elke function call zal de functie zijn bijhorende code uitvoeren.

```
// function declaration
function naamVanJeFunctie(){
    /*
        Dit code block wordt uitgevoerd bij de elke
        function call.
    */
}

// invoking a function (function call)
naamVanJeFunctie();
```

Arguments

Bij het aanroepen van een function is het mogelijk gegevens van buitenaf mee te geven: de argumenten. Bij de function call 'injecteer' je de function als ware met die gegevens van buitenaf (passing arguments genoemd).

```
// invoking a function (and passing two arguments).
toonWelkomBericht("Henk", "De Wildt");
```

In bovenstaand voorbeeld wordt de function genaamd toonWelkomBericht() aangesproken. Er worden twee strings meegegeven die de function ontvangt als argumenten.

Parameters

Bij het aanroepen van functions kun je niet zomaar willekeurig argumenten meegeven. Bij het declareren van de function heb je bepaald hoeveel arguments de function gebruikt en in welke volgorde de arguments moeten worden meegegeven.

Als je bij het aanroepen van de function twee arguments wilt meegeven, dan moet je in de function declaration ook exact twee parameters hebben. Parameters zet je (komma-gescheiden) tussen de haakjes van je function declaration.

Parameters zijn een soort 'placeholders' voor de gegevens die je later als argument meegeeft bij het aanroepen van de functie.

```
function toonWelkomBericht(voornaam, achternaam){
    alert("Welkom, " + voornaam + " " + achternaam);
}
```

Bovenstaande code toont een window alert met daarin de tekst "Welkom, Henk De Wildt".

Jij hebt jouw function zelf ontworpen, dus je weet wat het data type van de arguments moet zijn. Toch moet je de ontvangen gegevens soms controleren. Misschien zijn de arguments afkomstig van user input? We kunnen (en mogen) dan op voorhand niet aannemen dat de argumenten van het juiste formaat zijn. Ga er altijd vanuit dat de user kwaad in de zin heeft.

Return values

Nadat een function is aangeroepen geeft het *altijd* een waarde terug: de 'return value'. Als je bij de function declaration zelf geen return value opgeeft, dan zal de function na het aanroepen 'undefined' teruggeven.

Je kunt de return value opslaan in een variable voor toekomstig gebruik. Voorbeeld:

```
function bereken(getal1, getal2){
    return getal1 + getal2;
}
var uitkomst = bereken(5,12);
```

Variable scope

De plaats waar je een variable declareert heeft gevolgen voor waar je de variable kunt gebruiken:

- **Local scope:** ook wel function level. Als je een variable declareert in een function, dan is die variable alleen in de betreffende function te gebruiken.
- **Global scope:** de variable wordt onthouden, zolang de webpagina is geladen. Global variables kosten meer geheugen, omdat ze langer bewaard moeten blijven. Gebruik globals alleen als het niet anders kan.

```
// global scope
var width = 600;
function showWidth(){
    // local scope
    var width = 300;
    console.log(width);
}
showWidth(); // 300! Niet 600!
```

Function expression

Als je een function gebruikt op een plek waar je normaal gesproken een expression zou verwachten, dan wordt de function als expression verwerkt. Dit heet een function expression.

```
var myFunc2 = function() {
    return 5;
}
```

Anonymous function

De anonymous function is een function zonder naam (ook wel self-excecuting function genoemd):

```
var myFunc = function() {
    alert("Hallo!");
}
document.getElementById("button").onclick = myFunc;
```

De function wordt pas uitgevoerd als de interpreter op desbetreffende regel is aangekomen.

Immediately Invoked Function Expressions

De IIFE (spreek uit als 'iffy') is een function zonder naam. Ze worden maar een keer uitgevoerd zodra de interpreter op desbetreffende regel is aangekomen.

De IFFE voorkomt conflicten tussen twee scripts die dezelfde variabele naam gebruiken. De variables die je in een IIFE gebruikt zijn beschermd voor de buitenwereld (zie variable scope).

zelf executerende functies

```
var oppervlakte = (function(){
    var width = 3;
    var height = 2;
    return width * height;
})();
```

De eerste set haakjes (in cyaan) noemen we de grouping operators. De tweede set haakjes (in magenta) noemen we de final parentheses, die ervoor zorgen dat de interpreter het geheel als expression behandeld.

Naming collisions

Het is gebruikelijk dat scripts door verschillende mensen zijn geschreven. Dit kan problemen opleveren omdat het ene script dezelfde namen gebruikt voor variables en functions als het andere script (naming collisions). Voorkom naming collisions door het gebruik van IFFE's.

Objecten

Je hebt eerder in de reader gelezen waarom we met objecten werken. Je kunt objecten aanmaken op dezelfde manier als een variabele (dus met het var keyword).

Object literal (les)

Onderstaande code maakt een object aan genaamd 'planeObject' met twee properties (engines en type) en één method (talkToPassengers):

var boodschappen = [..., ..., ...]
array

```
var planeObject = {
    engines   : 4,
    propellor : false,

    talkToPassengers : function() {
        alert("Hello party people, Welcome aboard.");
    }
};
```

```
planeObject.engines; // Object property opvragen
planeObject.talkToPassengers(); // Method uitvoeren
```

Object prototype (les)

Stel: je maakt een race game, dan wil je op de racebaan niet één, maar tientallen auto's laten rijden!

Je maakt deze auto's aan de hand van dezelfde blauwdruk of bouwtekening (object prototype), maar elke auto (object instance) op de racebaan is natuurlijk wel anders. Zo heeft iedere auto zijn eigen kleur, merknaam of snelheid.

Object prototypes maak je met het 'function'-keyword. Het is gebruikelijk de naam van de prototype te beginnen met een hoofdletter.

```
function Car(theBrand, theType, thePrice){
    this.sprite = new Image();
    this.sprite.src = "../images/car.svg";

    this.brand = theBrand;
    this.type = theType;
    this.price = "€" + parseFloat(thePrice);
    this.speed = 0;
    this.accelerate = function(byHowMuch){
        this.speed += byHowMuch;
    };
}
```

Object constructor (les)

Nadat je een prototype hebt aangemaakt, kun je instances maken. Dit doe je met het 'new'-keyword. Dit keyword is een zogenaamd constructor function, waarmee je een instance maakt.

brand type price

```
var car1 = new Car('Audi', 'a8', '40000');
var car2 = new Car('Corvette', 'C7', '95000');
```

Dot syntax notation

De properties en methods zijn toegankelijk middels de 'dot syntax notation':

```
alert(car1.speed);
car2.accelerate(140);
```

'this'-keyword (les)

Het 'this'-keyword kom je vaak tegen in combinatie met objecten. Met 'this' refereer je *in* het object, *naar* het object. Met andere woorden: de plek waar 'this' voorkomt bepaalt de betekenis ervan.

```
/*
    Een object genaamd 'shape' maken. Het object
    heeft een width-property en een showWidth-method.
*/
var shape = {
    width: 300,
    showWidth: function(){
        return this.width;
    }
};
console.log(shape.showWidth());
```

Object Models

Programmeurs maken modellen van gegevens en voorwerpen, zodat een computerprogramma er vervolgens iets mee kan doen. De browser kent twee modellen:

1. het model van de browsersoftware zelf: Browser Object Model (het window object).
2. het model dat de browser maakt van een webpagina: Document Object Model (het document object).

Browser Object Model

Het aanspreken en manipuleren van browsereigenschappen kan met de Browser Object Model.

Met het JavaScript 'window'-object kunt je eigenschappen van het browserscherm aanpassen, zoals het webadres en de hoogte en breedte van het scherm.

Het 'window'-object kun je ook zien als het 'root'-object. Het object is het startpunt van een complexe boomstructuur met parent- en childobjecten. Het belangrijkste childobject van window is de ingeladen webpagina (het document object).

Childobjecten zijn bereikbaar via de dot-notation:

```
window // De webbrowser.
window.document // De webpagina.
```

Het window-object heeft net als alle JavaScript objecten properties en methods.

Document Object Model

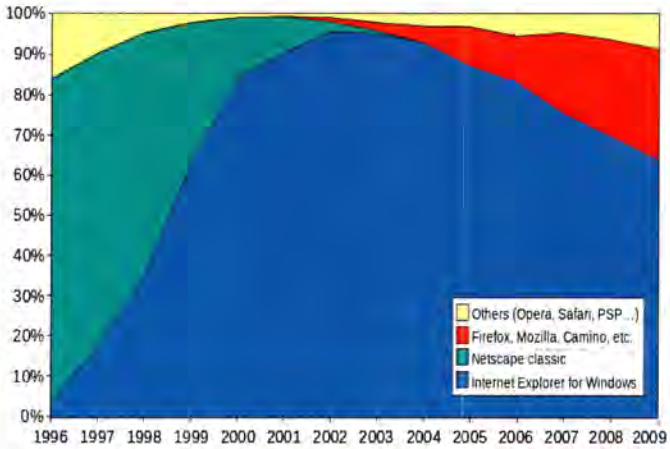
Voordat de browser een webpagina kan tonen moet het document eerst volledig zijn ontvangen. Daarna creëert de browser een model van de webpagina: de document object model. De DOM wordt opgeslagen in het geheugen en aan de hand hiervan wordt het webdocument getoond door de render-engine.

De DOM is een manier om de inhoud van een document te conceptualiseren in de vorm van een boomstructuur met een parent/child relation.

Alle HTML-elementen worden omschreven als objecten met eigenschappen (properties) en mogelijkheden (methods).

Browser wars

Tussen ±1995 en 2000 heerste er een strijd voor marktaandeel tussen twee browsergiganten: Netscape met Navigator en Microsoft met Internet Explorer.



De browsers hadden elk hun eigen manier om een ontvangen webdocument te conceptualiseren. Netscape gebruikte de volgende code om een HTML-element aan te spreken: `document.layers['mijnElement']`. Microsoft gebruikte: `document.all['mijnElement']`

Dit betekent dat JavaScript geschreven voor Netscape niet functioneerde in Internet Explorer en visa versa! Als webdeveloper moest je dus voor beide browsers andere JavaScript ontwikkelen.

W3C DOM

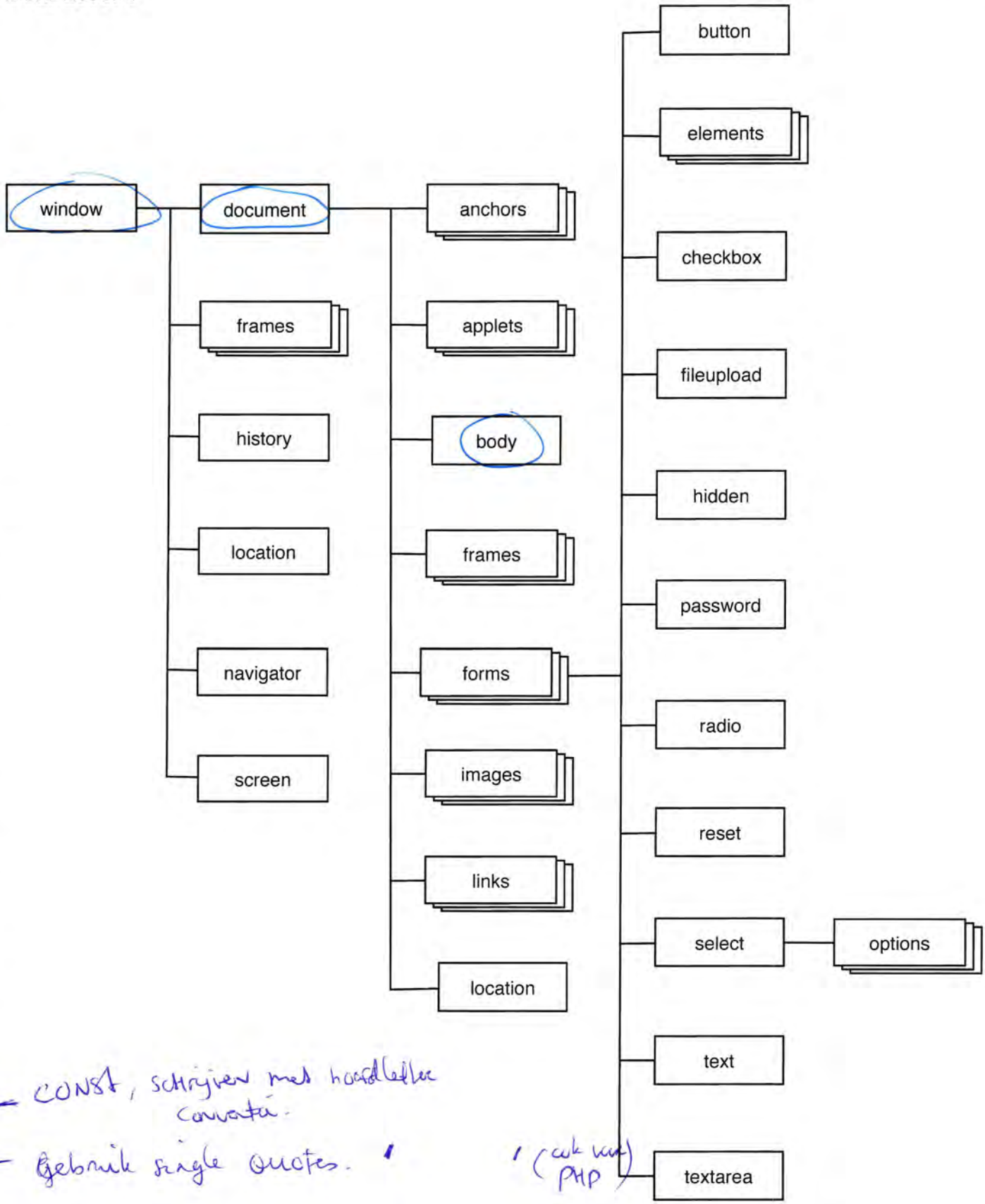
Gelukkig gingen Netscape en Microsoft uiteindelijk als leden van het W3C samenwerken om één standaard DOM te ontwikkelen. In oktober 1998 werd deze nieuwe 'DOM Level 1 standard' gelanceerd. De universele syntax om HTML-elementen aan te spreken werd:

```
document.getElementById('mijnElement');
```

Middels de DOM kun je alle objecten en HTML-elementen van een webpagina manipuleren:

```
// Een array met alle afbeeldingen.
document.images;

// De derde afbeelding van het document opvragen.
document.images[2];

// Een array met alle form-tags.
document.forms;

// Het formulier met het 'name'-attribute details.
document.forms['details'];

// Een array met alle weblinks.
document.links;
```

Dom Level I



- CONST, schrijven met hoofdletter
  Convention.
- Gebruik single quotes. ' (ook van PHP)

Nota Bene!

Deze pagina is niet compleet. Bekijk developer.mozilla.org of de 'JavaScript and Browser Objects Quick Reference' die je tijdens de lessen hebt ontvangen.

Window-object

Property	Omschrijving
.console L	Een verwijzing naar het 'console'-object van de webbrowser. Dit is een apart schermgedeelte dat vaak onderdeel is van de debugger of de developer tools.
.devicePixelRatio	De ratio tussen de fysieke pixels en de css pixels van het device.
.document	Een model van de ontvangen webpagina.
.fullScreen	Een boolean dat aangeeft of de browser in full screen modus staat.
.history	Een verwijzing naar een lijst met bezochte pagina's van het huidige scherm of tab.
.innerHeight	Hoogte van de browser viewport in pixels, inclusief een eventuele scrollbar.
.innerWidth	Breedte van de browser viewport in pixels, inclusief een eventuele scrollbar.
.location	De huidige URL of file path.
.name	Get / set de naam voor het window of de tab.
.navigator.userAgent	Informatie over de browservendor en -versie achterhalen.
.pageXOffset	Hoeveel pixels het window is gescrold in de lengte. Je kunt ook .scrollX gebruiken.
.pageYOffset	Hoeveel pixels het window is gescrold in de hoogte. Je kunt ook windows.scrollY gebruiken.
.screenX	X-coordinaat van het browserscherm, gerekend vanuit de linker bovenhoek van het beeldscherm.
.screenY	Y-coordinaat van het browserscherm, gerekend vanuit de linker bovenhoek van het beeldscherm.
.status	Get / set tekst op de statusbalk van de browser. Deze balk is in moderne browsers niet meer zichtbaar.

Method	Omschrijving
.alert(); L	Een dialoog voor de gebruiker.
.close(); L	Het window of de tab afsluiten. Werkt alleen als het eerder is aangemaakt met de .open() method.
.confirm();	Een dialoog voor de gebruiker met cancel (return value 'false') en ok (return value 'true') knoppen.
.console.log(string); L	Een tekst naar het consolescherm van de webbrowser schrijven.
.history.back();	Vergelijkbaar met de terugknop.
.history.go(-2);	Vergelijkbaar met twee keer achter elkaar de terugknop indrukken. Als dit niet mogelijk is, is de return value 'undefined'.
.open(url, name, features); L	Een nieuw browserwindow of -tab aanmaken. Deze functie wordt door browsers vaak onderdrukt als onderdeel van anti-spam maatregelen.
.print(); L	Opent het printer dialoog.
.prompt(string);	Een dialoog voor de gebruiker waarin tekst gezet kan worden. Het argument van de method is een string waarmee je de gebruiker een vraag kunt stellen. De cancel-knop van de prompt pop-up geeft de return value 'null'. De ok-knop geef als return value de inhoud van de string die de gebruiker zelf heeft ingetypt.

Escapes (JS)

\\
\'
\"
\n — newline
\R — return
\t — Tab

\b — Backspace (x)
\f — form feed (x)

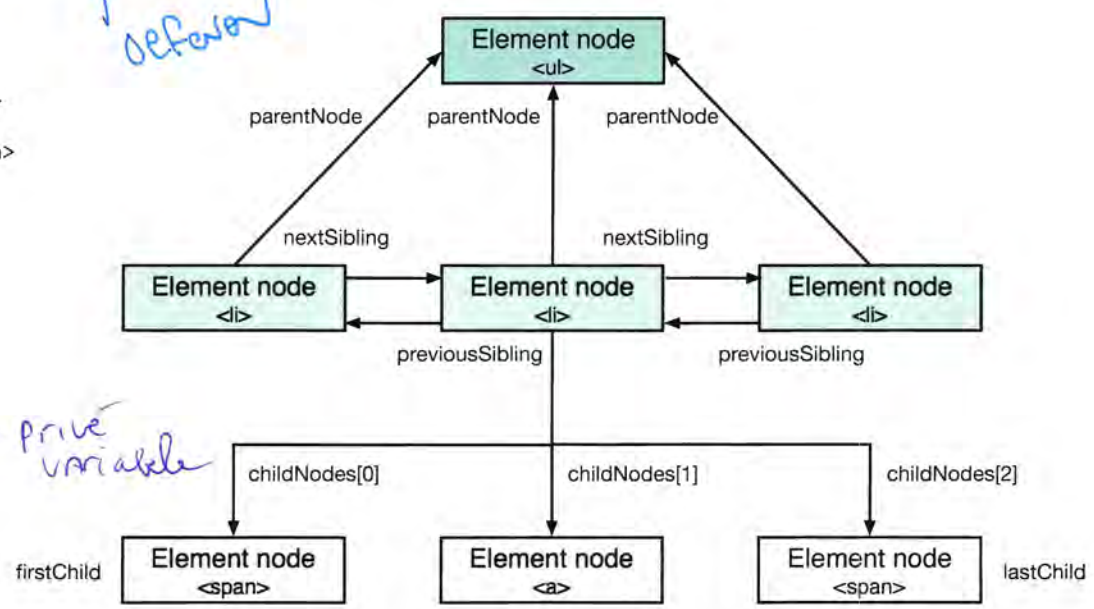
Nota Bene!

Deze pagina is niet compleet. Bekijk developer.mozilla.org voor een complete reference guide.

Property	Omschrijving
.characterSet	De characterset van het document.
.doctype	De HTML-doctype van het document.
.domain	Een string met de domeinnaam van de website. Bijvoorbeeld www.cmm.nl.
.lastModified	Datum en tijd wanneer het document voor het laatste is bewerkt.
.title	De tekstcontent van de title-tag van het document.
.URL	De Uniform Resource Locator van het document (webadres inclusief protocol).

DOM Traversing (travelling through)	
Property / Method	**Omschrijving**
.childElementCount	Het aantal child elementen van de parent element (geeft géén text- of attribute nodes terug).
.children	Een array met alle child elementen.
.firstChild	Selecteert de eerste node van de parent element
.firstElementChild	Het eerste child element node van de parent element.
.forms	Array met alle formulieren van het document.
.getElementById(id)	Selecteert de element node met de als parameter opgegeven ID-attribute.
.getElementsByClassName(class)	Array met elementen die de opgegeven class hebben.
.getElementsByTagName(tag)	Array met elementen die de opgegeven tag zijn.
.images	Array met alle afbeeldingen van het document.
.lastChild	Selecteert de laatste node van de parent
.lastElementChild	De laatste child element node van de parent element.
.links	Array met alle a-tags van het document.
.nextSibling	Selecteert de eerst volgende sibling node
.parentNode	Selecteert de eerste anchestor node.
.previousSibling	Selecteert de eerst voorgaande sibling node
.querySelector(cssSelector)	Een CSS-selector gebruiken (in de vorm van een string) om het eerst overeenkomende element node te verkrijgen.
.querySelectorAll(cssSelector);	Een CSS-selector gebruiken (in de vorm van een string) om een lijst te verkrijgen van alle overeenkomstige element nodes.

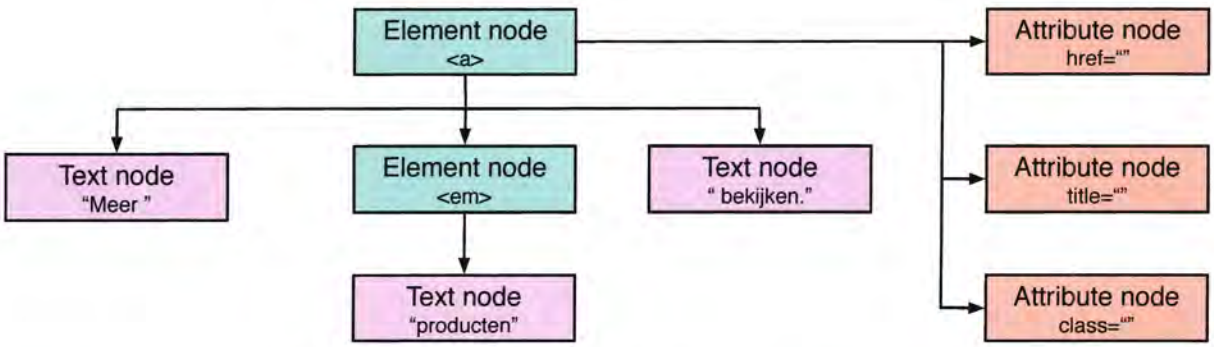
```
<ul>
    <li></li>
    <li>
        <span><span>
        <a></a>
        <span></span>
    </li>
    <li></li>
</ul>
```



DOM-elementen aanmaken	
Property / Method	**Omschrijving**
d.createElement(nieuweTag)	Een nieuw element node aanmaken (bijvoorbeeld een lege p-tag).
d.createTextNode(string)	Een nieuw text node aanmaken met de opgegeven stringtekst.
d.write(html)	Het gehele document herschrijven met de opgeven string van HTML-code (niet aangeraden).
e.appendChild(elementNode)	De opgegeven element node in een parent element node zetten. Het nieuwe element wordt als laatste sibling toegevoegd.
e.innerHTML	Get / set HTML-content van een element. Gebruik deze property niet voor gegevens die je ontvangen hebt van de gebruiker (vanwege mogelijke XSS attacks). Gebruik liever de veiligere method .textContent of de property .innerText. Eventuele event handlers voor de nieuwe content zullen niet meer werken. Als event handlers belangrijk zijn kun je de .createElement(), .createTextNode() en .appendChild() methods gebruiken.
e.innerText	De complete tekstinhoud van een element getten of setten, inclusief tekstinhoud van eventuele descendant element nodes.
e.nodeValue	Get / set de tekstinhoud van een text node.
e.textContent(string)	Get / set text inhoud van een element. Werkt in Internet Explorer sinds versie 9.
e.removeChild(elementNode)	De opgegeven element node van een parent deleten.

Nota Bene!
Note bene: In bovenstaand schema staat er een 'd' of 'e' voorafgaand aan de method of property. De 'd' staat voor 'window.document'. E staat voor een HTML-element.

```
<a href="" title="" class="">
   Meer <em>producten</em> bekijken
</a>
```



Attribute nodes bewerken	
Method	**Omschrijving**
.hasAttribute(attribute)	Controleren of het element de attribute heeft dat als parameter is meegegeven (return value true of false).
.getAttribute(attribute)	De value van het gevraagde attribute opvragen.
.setAttribute(attribute, value)	De value van een attribute wijzingen. Voorbeeld: el.setAttribute("class", "currentItem").
.removeAtrribute(attribute)	Het opgevraagde attribute weghalen.

Nota Bene!
Deze pagina is niet compleet. Bekijk developer.mozilla.org voor een complete reference guide.

Events

Events zijn gebeurtenissen die in de browser plaatsvinden. Events kunnen van toepassing zijn op het window-object, het document-object en natuurlijk de HTML-elementen van het document-object.

Event binding

Een event koppelen aan een object noemen we 'event binden'. Eventbinding kan op verschillende manieren gebeuren:

Methode #1: HTML event handlers

Deze achterhaalde methode wordt afgeraden, vanwege de principes graceful degredation en progressive enhancement. Je moet HTML en JavaScript strict van elkaar scheiden.

```
<div onclick="jouwFunctie('argument');">Tekst</div>
```

Methode #2: DOM event handlers

Deze methode hoort bij de oorspronkelijk DOM specificatie en daarom is er zeer goede browserondersteuning. In dit voorbeeld gebruiken we de 'onclick'-property. Deze property ontvangt de return value van een functie:

```
htmlObject.onclick = functieZonderArgumenten;
```

Het nadeel van deze methode is dat je niet zo maar argumenten kan meegeven aan de functie:

```
// Dit werkt NIET:
htmlObject.onclick = jouwFunctie('argument');
```

De anonieme function biedt uitkomst, omdat het als een soort 'wrapper' werkt. De anonieme functie is hier een expression waarvan de inhoud eerst wordt geëvalueerd. Het resultaat wordt toegekend aan de 'onclick'-property:

```
// aangeraden methode
htmlObject.onclick = function(){
    jouwFunctie('argument');
};
```

Methode #3: .addEventListener()

DOM level 2 event listeners zijn geïntroduceerd in 2000 (DOM level 2). Deze methode werkt niet op oude Internet Explorer browsers (versie 8 en eerder). De .addEventListener() method is de aangeraden manier om events te binden.

```
deDiv.addEventListener('click', function(){
            jouwFunctie('argument');
}, false);
```

Bubbling?

De method ontvangt 3 argumenten:

1. Het event dat moet worden opgevangen.
2. De function die moet worden uitgevoerd als het event plaats vindt.
3. Event flow: een boolean die bepaalt hoe events worden afgehandeld.

Event Flow

Het derde argument van de .addEventListener() method is de event flow.

Voorbeeld: een a-tag in een li-tag in een ul-tag. Alle drie zijn ze voorzien van een click event. Welk event moet JavaScript als eerste afhandelen?

Event bubbling

De default is event bubbling (setting: false). Bubbling gaat van meest specifieke tag naar minst specifieke tag. Bijvoorbeeld: a - li - ul - body - html - document.

Event capturing

Event capturing (setting: true) wordt niet ondersteund door IE8 en eerder. Het gaat van minst specifieke tag naar meest specifieke tag: document, - html - body - ul - li - a.

Event Object

Elke keer als er een event plaats vindt, maakt JavaScript automatisch een event object instance aan. Het event object bevat handige informatie zoals:

- Op welk HTML-element het event plaats vond.
- Welke toetsenbordknop de gebruiker indrukte bij een 'keypress'-event.
- Welke X- en Y-coördinaat van de viewport was aangeklikt.

In onderstaand voorbeeld binden we een 'blur'-event aan het HTML object met de variable naam 'el'.

Als het event plaats vindt vangen we het event object op en geven het de naam 'e'. We maken daarna een window alert waarin we de 'target'-property van het event object gebruiken.

```
el.addEventListener('blur', function(e){
   alert("De tag die het event afvuurde was " +
      e.target);
}, false);
```

Het eerste argument van de anonieme function is het event object (`function(e)`). Het event object is hier 'e' genoemd. We hadden hier ook `function(eventObject)` van kunnen maken als dat beter leesbaar is. Het gaat erom dat we met dit argument het event object 'opvangen' in een variable, zodat we er iets mee kunnen doen.

Het event object hoeft niet worden opgevangen als je het niet nodig hebt, maar het kan ook geen kwaad om het op te vangen om er vervolgens niets mee te doen.

Event Object Properties

De properties van het event object geeft meer informatie over het event dat plaats vond:

- **e.target:** geeft het meest specifieke element waar de event interactie mee had.
- **e.keycode:** de codepoint van een ingedrukte toets op het toetsenbord (van toepassing bij keypress events).
- **e.type:** het type event dat plaats vond.
- **e.cancelable:** de boolean geeft aan of het default behavior van een element met de e.preventDefault() method kan worden gecancelled.
- **e.screenX:** horizontale coördinaat van de muisklik t.o.v. het hele scherm (niet de window, maar het computerscherm).
- **e.screenY:** verticale coördinaat van de muisklik (niet de window, maar het computerscherm).
- **e.clientX:** horizontale coördinaat van de muisklik t.o.v. de browser window.
- **e.clientY:** verticale coördinaat van de muisklik t.o.v. de browser window.

Event Object Methods

Natuurlijk heeft het event object ook methods:

- **preventDefault():** cancelt het default behaviour van een element (als het cancelable is). Bijvoorbeeld de href-actie van een a-tag stoppen of het insturen van een formulier tegenhouden.
- **stopPropagation():** het event mag niet verder bubbelen of capturen naar een andere element.

KeyCodes

Een 'input'-element (maar ook het 'window'-object) kan luisteren naar key events van het toetsenbord. Elke toets heeft zijn eigen KeyCode, zo is 37 de linker cursortoets, 38 is omhoog, 39 is rechts en 40 is naar beneden:

```
window.addEventListener('keydown', function(e){
    console.log(e.keyCode);
    if(e.keyCode == 37){
        // cursor left
    } else if(e.keyCode == 38){
        // cursor up
    } else if(e.keyCode == 39){
        // cursor right
    } else if(e.keyCode == 40){
        // cursor down
    }
}, false);
```

window.onload

JavaScript moet pas worden uitgevoerd *nadat* de browser klaar is met het inladen van het webdocument. We kunnen namelijk geen elementen en objecten manipuleren als de browser nog geen model heeft gevormd!

De browser zal anders een error teruggeven dat het object dat we willen aanspreken (nog) niet bestaat!

De browser moet dus de kans krijgen de DOM op te stellen, *vóórdat* wij met onze JavaScript DOM-objecten aanspreken en manipuleren:

```
window.onload = function(){
    /*
        Het window object vuurt het event 'onload',
        zodra het document geheel is ingeladen.

        Nu kunnen we hier onze code uitvoeren.
    */
};
```

JavaScript Library

jQuery is een (gratis en open-source) verzameling scripts die geschreven zijn in JavaScript.

Het voordeel van jQuery is dat je veel minder code nodig hebt dan bij normale JavaScript en dankzij CSS-achtige selectors is het ook eenvoudiger en sneller te schrijven.

jQuery is volledig cross-browser en cross-platform getest en heeft het goede performance en snelheid.

Er is een grote community actief, dus er zijn online veel praktijkvoorbeelden te vinden en de documentatie is van wereldklasse.

Nadelen

jQuery is een grote, redelijk zware, library. De meeste websites gebruiken slechts een fractie van de jQuery library. Er worden dus veel onnodige kilobytes ingeladen. Gelukkig zijn er zeer goede 'light-weight' alternatieven, zoals zepto.js en Dojo.

jQuery Library & Object

- `$;` is de korte notatie voor de jQuery functie.
- `$();` is het complete jQuery object. `jQuery()` werkt ook.
- `$(document);` is het hele document (inclusief alle children)
- `$(document).ready(function(){ // code });` is de jQuery variant van: `window.onload = function(){ // code };`

JavaScript developers gebruiken vaak het dollarteken als shortcutsymbool voor hun functies of library, wat kan leiden tot naming collisions. Gebruik in dat geval `jQuery()` in plaats van `$()`. Je zou ook IFFE's kunnen toepassen.

jQuery variables

Gebruik het dollarteken voor een variable name, om zo onderscheid te maken tussen normale JavaScript variables en jQuery variables.

Zo zie je in één oog op slag of een variable gegevens bevat die uniek zijn aan jQuery:

```
var $jQueryObjecten = $('*');
console.log(typeof $jQueryObjecten);

var javaScriptObjecten = document.body;
console.log(typeof javaScriptObjecten);
```

Selectors

- `$('*');`
- `$('a');`
- `$('.card');`
- `$('#homeServices');`
- `$('main a');`
- `$('header, main');`

Tree Traversing

- `.children()`
- `.children().first()`
- `.children().last()`
- `.children().slice(2,4)`
- `.siblings()`
- `.parent()`
- `.parents()`
- `.closest()`
- `.next()`
- `.prev()`
- `.nextAll()`
- `.prevAll()`
- `.find()`

Filter / Test

- `:contains("text");`
- `:empty()`
- `:has en .has()`
- `:not() en .not()`
- `filter()`

Order in selection

- `:eq()`
- `.eq()`
- `:lt()`
- `:gt()`

DOM-manipulatie

- `.before()`
- `.after()`
- `.html()`
- `.text()`

Attributes

- `.addClass()`
- `.removeClass()`
- `.attr()`
- `.removeAttr()`

CSS

CSS poperties worden in camel case geschreven, maar de gewone schrijfwijze werkt ook. Shorthand CSS wordt niet ondersteund.

```
// Een enkele CSS-property:
$("h2").css("color", "rgba(0,255,0,0.2)");
$("h2").css("paddingLeft", "60px");
$("h2").css("paddingLeft", "+=150");
```

```
// Meerdere CSS-properties
$("div").css({
    color       : 'white',
    fontSize    : '2em'
});
```

Events

```
// Één event binden, met het event object.
$("h2").on('click', function(e){
    console.log(e.type, e.target, e.pageX, e.pageY);
});

// Meerdere events binden, zonder event object.
$("div").on('click mouseout', function(){
    $(this).css("color", "rgba(0,0,255,1)");
});
```

Het jQuery event object is volledig cross browser en platform compatible, in tegenstelling tot het event object van de klassieke .addEventListener() method.

Each

De .each() method is vergelijkbaar met de klassieke for-loop:

```
$("div").each(function(index, element){
    console.log("Div #:"+index+" Object: "+element);
});
```

Animate

De animate method heeft vier argumenten:
$(object).animate(css, duration, easing, oncomplete);

- **css:** een object (daarom ook de {}) met CSS-properties. Sommige properties zoals color en backgroundColor zijn niet te animeren. Je kunt hiervoor de jQuery color plug-in gebruiken.
- **duration:** duur van de animatie in milliseconde.
- **easing:** de gewenste keyframe interpolatie. Je kunt kiezen uit swing of linear (de jQuery easing plug-in ondersteunt veel meer methodes).
- **oncomplete:** de call-back functie die wordt uitgevoerd, zodra de animatie klaar is.

```
$("div").animate({
    left: "+=200px"
}, 2500, "swing", function(){
    doIetsAlsDeAnimatieKlaarIs();
});

$(this).animate({
    opacity : 1.0
}, 2500, "linear", function(){
    $(this).text("klaar!");
});
```

Animatie is ook mogelijk met de volgende jQuery methods:

- .fadeIn()
- .fadeOut()
- .fadeToggle()

- .slideDown()
- .slideUp()
- .slideToggle()

- .hide()
- .show()
- .toggle()

- .addClass()
- .removeClass()
- .toggleClass()

Chaining

Een verkorte schrijfwijze is meerdere methods, achter elkaar, in één statement te zetten. Dit principe heet 'chaining':

```
$("p.neat").addClass("ohmy").show("slow");
```

AJAX ondersteuning

jQuery ondersteunt AJAX (Asynchronous JavaScript And Xml). Met AJAX kun je met een webserver communiceren, zonder dat een webpagina herladen hoeft te worden.

De communicatie tussen client en host verliep vroeger middels een XML-bestand, maar tegenwoordig wordt JSON (JavaScript Object Notation) gebruikt.

Plug-ins

Er is een enorme hoeveelheid jQuery plug-ins beschikbaar:

- jQuery UI
- jQuery Color
- jQuery Easing
- Smooth Scroll
- AniView
- FullPage
- DrawSVG
- Wow.js
- Responsive Image Maps
- en tienduizenden andere!

Ze bestaan vaak uit een stuk HTML, CSS en JS. Wees voorzichtig met plug-ins: ze zijn vaak zwaar en maken de website trager.

De noodzaak voor CSS Preprocessors

In het begin van je project gaat het schrijven van CSS lekker snel, maar naar mate het project vordert wordt het beheer moeilijker, onoverzichtelijker en trager.

Hoe complexer websites worden, hoe meer je verlangt naar écht modulaire en programmeerbare CSS met variables, functions en loops. Dat kan met CSS preprocessors.

LESS vs Sass

LESS en Sass (Syntactically Awesome Style Sheets) zijn twee populaire CSS preprocessors. LESS is eenvoudiger te installeren en vereist minder tools om te draaien. Sass is robuster en heeft meer uitbreidingen (extensies) en daarom populair.

Sass vs SCSS

Sass wordt vaak in één adem genoemd met SCSS, maar de syntax verschilt. Sass is concise (minder 'verbose', of te wel korter te noteren) dan SCSS. Het gebruikt indentations (tabs) en nieuwe regels om selectors te nesten. SCSS gebruikt brackets en semicolons, vergelijkbaar met de CSS syntax:

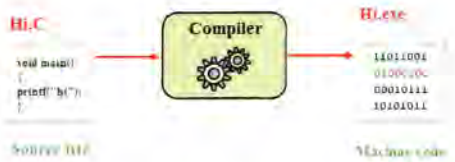
```
/*
   SASS: het inspringen en één regel per declaratie
   is verplicht!
*/

$blue: #3bbfce
$margin: 16px

.content-navigation
  border-color: $blue
  color: darken($blue, 9%)

.border
  padding: $margin / 2
  margin: $margin / 2
  border-color: $blue

/*
  SCSS: meerdere declaties op één regel mag dankzij
  de semi-column. Inspringen is puur voor de
  leesbaarheid.
*/

$blue: #3bbfce; $margin: 16px;

.content-navigation { border-color: $blue;
  color: darken($blue, 9%);
}

.border {
  padding: $margin / 2; margin: $margin / 2;
  border-color: $blue;
}
```

Sassy CSS

SCSS staat voor Sassy CSS. SCSS is snel te leren omdat de syntax erg lijkt op CSS. Je kunt bestaande CSS-bestanden renamen naar .scss en ze daarna uitbreiden met SCSS specifieke code.

Compiler

Compilen is het transformeren van source code van een formaat naar een ander. Bij het compilen wordt een 'high-level programming language', die voor mensen makkelijk te gebruiken is, omgezet naar een 'low-level programming language'.



Je schrijft iOS apps met de programmeertaal Swift in het programma Xcode. Op een gegeven moment is je app af en geef je Xcode de opdracht de app te bouwen. Dit is het compilen: de fase waarin jouw Swift code omgezet wordt naar een executable programma in de vorm van binaire code.

Ook in Animate compileer je elke keer als je het project publiceert met CMD+enter.

Het compileren van SCSS

SCSS moet gecompileerd worden naar CSS. Een SCSS-compiler is op MacOS makkelijk te installeren, omdat MacOS beschikt over Ruby software en bijhorende RubyGems package manager.

Eventueel kun je ook tekstprogramma's gebruiken met een ingebouwde compiler, zoals Adobe DreamWeaver.

Commentaar

SCSS-projecten zijn complex. Zorg daarom voor goed en veel commentaar. JavaScript single-line commentaar mag en verschijnt niet in de ouput (de compiler neemt single-line comments niet mee).

```
// Dit commentaar wordt niet gecompileerd.
```

CSS-commentaar (multi-line JS-commentaar) kan ook, en wordt WEL door de compiler meegenomen.

```
/*
   Dit commentaar is standaard CSS-commentaar en
   wordt wel gecompileerd.
*/
```

Schrijf altijd bovenaan SCSS-bestanden met commentaar een inhoudsopgave.

Source Maps

.map

De compiler kan naast de css-bestanden, in de outputmap, 'source map'-bestanden compileren. Source maps zijn JSON-bestanden met daarin de locatie van het SCSS-bronbestand.

Je kunt de browser configureren zodat het de source map uitleest. De SCSS-code is daarna direct in de browser te bewerken. Een ander voordeel is dat browsers de webpagina automatisch herladen nadat er opnieuw is gecompileerd.

Aan de rechterkant van de selector staat het scss-bronbestand met het regelnummer. Je kunt hierop klikken en in de browser direct veranderingen maken.

Sass Partials

Partials zijn SCSS-bestanden die beginnen met een underscore (_). Je gebruikt partials om code op te delen in verschillende SCSS-bestanden.

```
@import "_jouwpartial" // De .scss is optioneel
```

Maak één hoofdbestand dat alle partials inlaadt (met de @import-directive). De compiler maakt hiervan één enkel CSS-bestand. Een voorbeeld hoe je met partials je project op een duurzame manier modulair kunt indelen:

```
klant-naam.scss (hoofdbestand)
    |- _setup.scss
    |- _reset.scss
    |- _layout.scss
    |- _typography.scss
    |- _icons.scss
    |- _shame.scss
```

Het is ook mogelijk partials te maken gebaseerd op functionaliteit, zoals _webshop.scss, _homepage.scss, _navigation.scss en _footer.scss.

Het is grotendeels een persoonlijke voorkeur hoe partials worden toegepast. Zorg daarom dat het hoofdbestand een inhoudsopgave bevat met duidelijk verklarend commentaar over de partials.

De _setup partial

In het hoofdbestand importeer je als eerste de _setup partial. Boven deze @import-directive mogen geen declarations staat (met uitzondering van een @charset-rule). De indeling van _setup:

1. Klassieke @import-rules
   (alleen gebruiken voor Google Fonts en dergelijke).
2. @import-directives naar externe mixin-libraries, zoals Bourbon, Neat, Bitters en Susy.
3. @font-face-declarations
   (meestal worden hiervoor mixin-libraries gebruikt).
4. Declarations van sassy maps en variables.
5. Jouw eigen @mixin functions. ?

De _shame partial

In complexe projecten is het gebruikelijk om als laatste een _shame partial te importeren. Je gebruikt _shame om na de lancering van de website even snel kleine updates en bugfixes te kunnen maken.

_shame gebruiken we dus als tijdelijke oplossingen, totdat er tijd is om de toegevoegde code in de juiste partials onder te verdelen.

Nesting

Groepeer selectors door ze te nesten. Het verbetert de leesbaarheid, omdat het parent element niet moet worden herhaald. Nest niet dieper dan vijf niveau's.

```
nav{
    padding: 1rem 0 0 0;

    ul{
        float: right;

        li{
            display: inline-block;

            a{
                display: block;
            } // einde a-rule
        } // einde li-rule
    } // einde ul-rule
} // einde nav-rule
```

Parent selector &

Met de ampersand (&) verwijs je in een geneste SCSS-rule naar de parent rule:

```
a{
    display: block;

    &:link, &:visited{
        color: navy;
    }

    &:hover{
        color: blue;
    }
}
```

Nested Properties (name spaces)

Veel CSS-properties hebben dezelfde name space, zoals font:

- font-family
- font-weight
- font-size
- font-style

Een name space moet normaal gesproken voor elke property in z'n geheel worden uitgeschreven en dat kost tijd. Met SCSS kun je een name space opgeven, zodat je het maar één keer hoeft te noteren:

```
p{
   color: black;
   font:{
      family: 'Lato', sans-serif;
      style: normal;
      weight: 300;
      size: 0.85rem;
   }
}
```

Variables

Gebruik SCSS-variables voor kleuren, fonts, breakpoints en andere gegevens die vaker terugkomen in verschillende rules:

```
$primaryFont: 'Lato', sans-serif;
h1{
   font-family: $primaryFont;
}
```

Variable scope

SCSS kent net als JavaScript variable scope. Een variable is alleen beschikbaar in de rule waarin het is gedeclareerd.

Onderstaande code werkt niet, omdat $primaryFont in de H1-rule is gedeclareerd, maar in de H2-rule wordt gebruikt:

```
h1{
   $primaryFont: 'Lato', sans-serif;
   font-family: $primaryFont;
}

h2{
   font-family: $primaryFont;
}
```

Declareer belangrijke variables in de global scope van een partial, bijvoorbeeld _setup.scss of _variables.scss en importeer de partial voor andere partials.

Omdat de variables als eerste worden geïmporteerd én omdat ze in de global scope zijn aangemaakt, kun je de variables aanroepen waar je maar wilt.

Data types

SCSS ondersteunt zeven data types:

- **Numbers:** zoals 1.2, 13 of 10px
- **Strings:** met en / of zonder quotes, zoals 'string 1', "string 2", string 3.
- **Kleuren:** HTML color names, hexadecimalen, rgb, rgba, hsl, hsla.
- **Lijsten** met spaties of komma's, zoals '1em 3em 0 2em' of 'Helvetica, Arial, sans-serif'.
- **Sassy Maps:** of te wel arrays.
- **Booleans:** true of false.
- **Nulls**: null.

Number Operations

Reken met operators die je kent vanuit JavaScript (-, +, /, *).

```
// Voorbeeld #1:
$lineHeight: $headingSize - ($headingSize * 0.1);

// Voorbeeld #2:
$breedte: 1000px;
width: $breedte / 2;

// Voorbeeld #3:
color: #010203 + #040506;
```

Sassy Maps (map-get)

Sassy maps lijken op JavaScript arrays. Het zijn variables die meerdere waardes kunnen opslaan:

```
// Declaration:
$map: (
  key: value,
  other-key: other-value
);

// Opvragen van een sassy map item:
map-get($map, key);
```

Sassy maps zijn erg handig voor kleuren en breakpoints:

```
$breakpoints: (
    small: 320px,
    medium: 540px,
    large: 768px,
    xlarge: 1200px
);

@media (min-width: map-get($breakpoints, xlarge)){
    h1{ color: green; }
}
```

@each directive

Met de @each directive kun je door een sassy map lo(o)pen:

```
$socialMediaIcons:(facebook, linkedin, twitter);

@each $icon in $socialMediaIcons {
    .icon-#{$icon}{
        background-image: url("icon-#{$icon}.png");
    }
}
```

Functions (native)

SCSS heeft, net als JavaScript, een aantal handige build-in functions (pre-defined objects en functions genoemd). Er zijn er te veel om op te noemen, dus raadpleeg de documentatie op sass-lang.com.

Om kleuren te manipuleren kun je onder andere mix(), darken(), lighten(), saturate() en desaturate() gebruiken. Bijvoorbeeld:

```
$primaryColor: #214290;
$colors: (
    greyMid:       mix(#FFFFFF, #000000, 50%),
    greyDark:      mix(#FFFFFF, #000000, 25%),
    primary:       $primaryColor,
    primaryLight:  lighten($primaryColor, 30%),
    primarySat:    saturate($primaryColor, 20%),
    primaryDesat:  desaturate($primaryColor, 20%)
);
```

@extend directive

Met @extend kun je properties van een selector doorgeven aan een andere selector. Gebruik @extend als je een groepje CSS-rules hebt die een relatie met elkaar delen.

Placeholder selector %

@extend wordt vaak gebruikt in combinatie met de placeholder selector (%). De placeholder selector lijkt op class (.) en id (#), maar heeft geen priority. Het is daarmee een neutrale selector zonder betekenis. Een voorbeeld van @extend in combinatie met de placeholder-selector:

```
%icons{
    display: inline-block;
    width: 1.5rem; height: 1.5rem;
    background-repeat: no-repeat;
    background-size: 100% 100%;
    background-position: center center;
}

.iconFacebook{
    @extend %icons;
    background-image: url("icon-facebook.png");
}

.iconTwitter{
    @extend %icons;
    background-image: url("icon-twitter.png");
}
```

De HTML code:

```
<a class="iconFacebook" href="#">Facebook</a>
```

Bovenstaande zorgt ervoor dat je in de HTML slechts één class nodig hebt, in plaats van twee of meer, zoals in dit klassieke voorbeeld:

```
<a class="icon facebook" href="#">Facebook</a>
```

@mixin directive

@mixin-directives zijn vergelijkbaar met JavaScript functions. Je kunt er herbruikbare code mee schrijven.

Dankzij @mixins kun je zelf een partial ontwikkelen met herbruikbare code die je in al je projecten toepast. Ook is het gebruik van niet-semantische class-namen (zoals .imgRight en .clearFloat) niet meer nodig.

```
@mixin transition{
    transition: all 200ms linear 0s;
}
```

@include directive

Met de @include-directive roep je een @mixin aan (vergelijkbaar met het aanroepen van een function in JavaScript):

```
a{
    display:block;

    &:link, &:visited{
        @include transition;
        color: #000000;
    }

    &:hover{
        color: #252525;
    }
}
```

@mixin parameters en argumenten

Je kunt een @mixin-directive voorzien van parameters:

```
@mixin font($family, $size, $lineHeight){
    font-family: $family;
    font-size: $size;
    line-height: $lineHeight;
}
```

En bij het aanroepen van de @mixin argumenten meegeven:

```
h1{
    @include font(Arial, 2rem, 1.8);
}
```

Conditional statements met @if

Met de @if-, @else- en @else if-directives is het mogelijk conditional logic toe te passen:

```
@mixin font($family){
    @if $family == primary {
        font-family: 'Lato', sans-serif;
    }

    @else if $family == secondary{
        font-family: 'Open Sans', sans-serif;
    }

    @else{
        font-family: sans-serif;
    }
}

h1{
    @include font(primary);
    color: green;
}
```

@extend vs @include

Op het eerste gezicht lijken @extend en @include op elkaar.

Gebruik @mixin voor CSS-rules die onderling géén relatie met elkaar hebben. Sommige developers zeggen dat de @mixin-directive niet goed wordt toegepast als het geen argumenten kan ontvangen.

Gebruik @extend in combinatie met een placeholder % selector voor CSS-rules die een relatie met elkaar hebben.

```
%brand-font {
    font-family: webfont, sans-serif;
    font-weight: 700;
}

...

h1 {
    @extend %brand-font;
    font-size: 2em;
}

...

.btn {
    @extend %brand-font;
    display: inline-block;
    padding: 1em;
}

...

.promo {
    @extend %brand-font;
    background-color: #BADA55;
    color: #fff;
}

...

.footer-message {
    @extend %brand-font;
    font-size: 0.75em;
}
```

```
.btn,
%btn {
    display: inline-block;
    padding: 1em;
}

.btn-positive {
    @extend %btn;
    background-color: green;
    color: white;
}

.btn-negative {
    @extend %btn,
    background-color: red;
    color: white;
}

.btn-neutral {
    @extend %btn;
    background-color: lightgray;
    color: black;
}
```

Links: @extend NIET correct toegepast (selectors hebben onderling geen relatie). Rechts: @extend WEL correct toegepast.

Correct opbouw van SCSS declarations

SCSS-declarations moeten als volgt worden opgesteld:

- Begin (indien van toepassing) met @extend directives.
- Dan komen @include directives.
- Daarna normale declaraties.
- Daarna pseudo-classes en pseudo-elementen zols &:hover{} en &::before {}.
- Daarna komen geneste selectors. Nest tot drie niveau's diep met een maximum van 50 regels per nesting.
- Als laatste komen de media queries.

```
nav{
    li{
        background-color: white;

        a{
            @extend %buttons;
            @include transitions;
            color: $primaryColor;
        }

        &:first-child a{
            border-top: 1px solid $primaryColor;
        }

        @include breakpoint(medium){
            margin: 1.2rem 0 0 0;
            display: inline-block;
        }
    }
}
```

SCSS best practises

- Werk modulair door je project op te delen in partials.
- Gebruik één hoofdbestand dat alleen partials importeert. Er wordt één CSS-bestand gecompileerd en je hebt maar één link-tag in de HTML staan.
- Schrijf géén vendor prefixes, maar gebruik een autoprefixer.
- De klassieke @import-rule (handig voor Google Fonts) moet als eerste worden uitgevoerd. Er mogen dus geen declarations boven de klassieke @import-rules komen te staan, anders zal de browser de @import-rule niet uitvoeren.
- Werk mobile first.
- Code moet correct zijn ingesprongen, anders wordt het onleesbaar.
- Compile altijd source maps mee, zodat je rechtstreeks in de browser kunt debuggen.
- Weet het verschil tussen @extend en @include en pas de technieken correct toe.
- Internet Explorer 9 en eerder ondersteunt maximaal 4095 regels per CSS-bestand. Internet Explorer 10 en later ondersteunt 65534 regels per CSS-bestand.

Domeinnaam

Registrars zijn bedrijven die domeinnamen uitgeven en reserveren. Een registrar is geaccrediteerd door een 'top-level domain'- (gTLD) of een 'country code top-level domain'- (ccTLD) register. In Nederland is dat de Stichting Internet Domeinregistratie Nederland (SIDN).

Niet elk bedrijf dat webhosting levert, levert ook (direct) de domeinnaam. Controleer vóór het kopen van een domeinnaam de betrouwbaarheid van de leverancier door bijvoorbeeld de voorwaarden te lezen.

Shared hosting

Shared hosting is een voordelig hostingproduct, waarbij men de webserver deelt met anderen. De servercapaciteit wordt over meerdere accounts verspreid. Het is geschikt voor kleine websites met beperkte bandbreedte.

Dedictated hosting

Met dedicated hosting (reseller accounts of managed hosting server genoemd) heb je zelf de beschikking over een complete webserver. Performance en beschikbaarheid wordt niet beïnvloed door andere gebruikers en de webserver is naar wens te configureren.

Serverplatform

De meeste webservers (±45%) draaien Apache software op een Linux of Unix besturingssysteem.

Apache wordt in één adem genoemd met de PHP server-side scriptingtaal en MySQL-databases.

Het Windowsplatform is met een marktaandeel van ±40% ook populair en ondersteunt PHP of .NET (ASP) server-side scriptingtalen.

Schijfruimte

Shared hosting accounts hebben beperkte schijfruimte. Voor een kleine website is minimaal één gigabyte aan te raden.

Dedicated servers hebben meerdere harde schijven in RAID-opstelling. Een halve terabyte schijfruimte is geen uitzondering.

Bandbreedte

Bandbreedte (dataverkeer) is te vergelijken met de benzine van een auto. Shared hosting accounts hebben (afhankelijk van het pakket) zo'n ±10 tot 250GB dataverkeer per maand. De website is voor de rest van de maand niet meer beschikbaar als de datalimiet wordt overschreden.

Staar je niet blind op de hoeveelheid bandbreedte; snelheid is veel belangrijker.

Wantrouw hostingbedrijven die 'onbeperkte bandbreedte' aanbieden. Jouw auto heeft tenslotte ook niet onbeperkt benzine.

Deze website heeft voor de rest van de maand geen benzine meer!

Back-ups

Het maken van back-ups is kostbaar en daarom niet vanzelfsprekend.

Dedicated webservers maken elke maand, elke week, elke dag en soms zelfs elke uur een back-up. De back-up wordt op een andere locatie opgeslagen ('off-site'). Dit betekent dat de back-up altijd beschikbaar is, zelfs in het geval van een ramp, zoals diefstal, brand, een aardbeving of overstroming.

Zelf een webserver installeren

Tot nu toe testten we websites door ze direct vanaf de harde schijf, via het file-protocol, in de webbrowser te bekijken. Dit is geen realistische testomgeving en ongeschikt voor het ontwikkelen van websites die WordPress-, PHP/MySQL- en AJAX-technieken gebruiken.

Een goede testomgeving:

- werkt met echte serversoftware, zoals Apache.
- is beschikbaar via het HyperText Transfer Protocol. Het file-protocol is ongeschikt voor het testen van complexe websites.
- is beschikbaar voor alle devices binnen een LAN, zodat de website kan worden getest op telefoons, tablets en diverse besturingssystemen en webbrowsers.
- beschikt over een server-side scriptingtaal (zoals PHP).
- beschikt over een databasesysteem (zoals MySQL) en een database beheeromgeving (zoals PHPMyAdmin).

MAMP PRO

MacOS is een Unix systeem en heeft de mogelijkheden voor Apache, PHP en MySQL ingebouwd. Het is relatief eenvoudig om van een Mac een webserver te maken.

Het configuren vereist echter veel technische kennis. Daarom is er software beschikbaar zoals MAMP en XAMP.

MAMP PRO ondersteunt meerdere accounts (virtuele hosts), die met een gebruiksvriendelijke GUI-interface worden geconfigureerd middels zeven tabs.

De interface van MAMP PRO is overzichtelijk ingedeeld in 7 tabs.

General

Een TCP poort is een digitale brievenbus. Tussen twee computers verloopt de communicatie via een poort.

Clients en hosts communiceren via TCP poort 80 (http), 443 (ssl) en 3306 (MySQL).

Om ervoor te zorgen dat het LAN-verkeer niet het WAN-verkeer in de weg zit, staat MAMP standaard ingesteld op de poorten 8888, 8890 en 8889.

Deze instelling kan handig zijn, maar je moet niet vergeten om deze afwijkende poortnummers toe te voegen aan elke url. Bijvoorbeeld: jouwWebsite.beta/nieuws.html:8888.

De afwijkende poortnummers zorgen vaak voor verwarring. Kies daarom voor `Set ports to 80,442 & 3306`.

Voor een lokale webserverinstallatie hoef je de setting '`Run apache/mysl server as user`' niet te veranderen.

Kies bij `Apache error log` om Apachelogbestanden op een toegankelijke locatie op te slaan. Apache logt onder andere alle errors en transacties tussen client en server: onmisbaar voor het opsporen van configuratieproblemen.

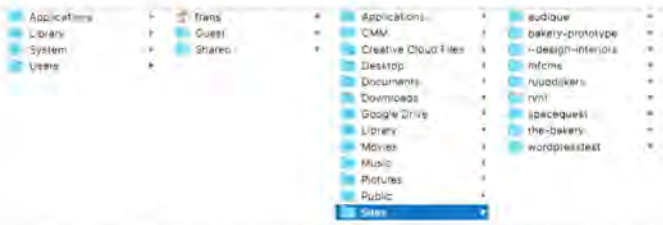
Hosts

Maak in deze tab een virtual host aan door links onderin op + te klikken. In de lijst is Localhost niet te verwijderen, omdat het de default host is.

Stel eerst de `Server name` in. Dit is het adres van de virtuele host, bijvoorbeeld www.jouwwebsite.beta.

Voorkom verwarring tussen LAN en WAN door de domeinnaamextensie '.beta' te gebruiken en gebruik *nooit* de .dev-extensie!

Elke host heeft natuurlijk zijn eigen `Document Root`. Wijs hier de www-map aan van je project, die je bij voorkeur in een submap van de Sites-map van jouw User account zet (*nóóit* op de Desktop).



Maak een projectmap aan in de Sitesmap van je user account. In de projectmap staat de www-map: de Document root.

Laat `IP address` staan op "*". Dit betekent dat de server het lokale LAN-adres van de computer gebruikt.

In de meeste gevallen kun je `Port number` laten staat op de default setting.

`PHP version` bepaalt de versie van de server-side scriptingtaal die je wilt gebruiken. Dit is te veranderen in de 'PHP'-tab. Gebruik dezelfde versie als de remote webserver.

`Dynamic DNS` hoef je niet in te stellen voor een normale LAN testomgeving.

`Naming resolution` bepaalt hoe de naam van de virtuele host wordt gemapt. `Via Xip.io` is een handige instelling als de website op verschillende apparaten binnen het LAN moet worden getest. Het configureren ervan vereist enige kennis.

Download via de Apple App Store of via de Google Play Store de gratis MAMP Viewer om zonder extra configuratie je virtual host te bekijken op mobile devices.

De 'Host'-tab is onderverdeeld in vier tabs: General (reeds besproken), Extended, SSL en Extras:

- **Extended** `(Options for <Directory> directive)`
  - Indexes: aanvinken om directory listings toe te staan (folderbeveiliging uit).
  - Includes: server side includes (SSI) gebruiken (bestanden met een shtml extensie).
  - FollowSymLinks: gebruik symbolic links als je wilt verwijzen naar resources buiten de document root.
  - SymlinksIfOwnerMatch: een strengere versie van FollowSymLinks.
- **SSL**
  - Maak een self-signed certificate aan om lokaal een SSL-verbinding te testen.

    Om het verkeer van een remote webserver te beschermen heb je een certificate bestand en de certificate key nodig van een certified authority (CA); een instantie dat SSL-certificaten uitgeeft.
- **Extras**
  - CMS-software installeren met een druk op de knop. Sla dit over en installeer WordPress handmatig.

Modules

Bepaalde Apache modules uitzetten of aanzetten. Gebruik dezelfde modules als de remote webserver.

PHP

Kies dezelfde PHP-versie als de remote webserver.

Controleer de PHP-versie en de beschikbare Apache modules van de remote server, door tijdelijk een 'phpinfo.php'- bestand met de inhoud `<?php phpinfo(); ?>` te uploaden. Vraag daarna met een webbrower de url van dit bestand op.

Delete phpinfo.php altijd na gebruik in verband met security issues.

Wees terughoudend met het selecteren van de nieuwste PHP-versie, omdat webserverbeheerders meestal even wachten met het installeren van de nieuwste software.

MySQL

Gebruik na het opstarten van de virtual hosts de phpMyAdmin software om MySQL databases te beheren.

Via phpMyAdmin is het mogelijk back-ups te maken van de volledige WordPress database.

phpMyAdmin maakt het beheren van een MySQL-database makkelijk.

Dynamic DNS

Deze tab is niet van toepassing voor lokale virtual hosts.

Postfix

Stel hier de routing in voor e-mailverkeer die de virtual host genereerd.

WordPress

WordPress is een gratis open-source content management systeem (CMS) voor blogs, websites en webshops. Het is uitgevonden door Matt Mullenweg, die zijn geld verdient met het maken van WordPress plug-ins.

Matt Mullenweg.

WordPress is oorsponkelijk bedoeld als CMS voor blogs, maar is uitgegroeid als defacto standaard voor het maken van websites en webshops.

Het succes van WordPress is te danken aan het feit dat het gratis, open source, SEO-vriendelijk en makkelijk in gebruik is. WordPress kent een grote developer community, die bezig zijn met ontwikkelen van het systeem, het schrijven en het bijwerken van de Codex-documentatie, het designen van thema's en het ontwikkelen plug-ins.

Statische website

Tot nu toe hebben we statische websites gemaakt. Er is een client (de webbrowser met een verzoek) en een host (de server met een antwoord).

Iedere URL van de website komt overeen met één document op de webserver. Statische websites met tien nieuwsartikelen hebben dan vaak elf nieuwspagina's: één overzichtspagina en tien artikelpagina's.

Het proces van een statische website. De client verzoekt een document, en de host levert het document.

Dit principe werkt goed voor kleine websites, maar wat als een webshop duizend producten heeft?

Dynamische websites

WordPress is een dynamisch systeem en werkt met een database en server-side scripting. Anders dan bij een statische website worden webpagina's 'on-demand' aangemaakt door de webserver.

Stel: de gebruiker bekijkt het nieuwsoverzicht van een WordPressblog en klikt op het derde nieuwsartikel. De http-request van de client is bijvoorbeeld:

```
www.mijnblog.nl/nieuws/3/het-derde-nieuws-artikel/
```

De webserver verwerkt het verzoek door in de database op zoek te gaan naar het opgevraagde nieuwsbericht (#3).

In de database bestaat een nieuwsbericht bijvoorbeeld uit een id, titel, het artikel en een afbeelding. Deze gegevens worden uit de database gehaald en met behulp van een server-side scriptingtaal (zoals PHP) 'on demand' in de artikelpagina geplaatst. De dynamisch gegenereerde webpagina wordt daarna opgestuurd naar de gebruiker.

Het proces van een dynamische website.

Thema's

Een thema is de front-end van een WordPress website. Het is het gedeelte waar de gebruiker interactie mee heeft en wat het uiterlijk van de website bepaalt.

Een thema voegt geen nieuwe mogelijkheden toe aan een WordPress installatie; dat doen de plug-ins.

Er zijn duizenden gratis en betaalde thema's te vinden, maar ze zijn lang niet allemaal even goed. Populaire marktplaatsen zijn Elegant Themes, ThemeForest en Theme Isle.

Met voldoende kennis op het gebied van PHP en MySQL is het ook mogelijk thema's helemaal op maat te ontwikkelen.

Themachecklist

- Bepaal of het thema visueel en functioneel past bij het project.
- Kies een thema dat met minimale aanpassingen bruikbaar is voor het project.

- Probeer het thema eerst uit, voordat je het koopt. Gebruik hiervoor een WordPress testinstallatie. De themamarktplaatsen hebben meestal online demo's.

- Bekijk hoeveel widgetarea's het thema heeft, waar ze staan in het webdesign en wat voor navigatiemenu's mogelijk zijn.

Andere zaken die bepalen of een thema geschikt is:

- Vormgeving (bronbestanden bijgeleverd?)
- Snelheid
- Geldige HTML5 en CSS3
- SASS-bestanden bijgeleverd?
- Bootstrap (welke versie?)
- Geoptimaliseerd voor zoekmachines
- RWD
- Goede ondersteuning voor de Nederlandse taal
- Compatibiliteit met de WooCommerce webshop plug-in.

Widgets

Widgets zijn websitetoevoegingen, die je gemakkelijk op een grafische manier kunt plaatsen. Voorbeelden zijn tekstblokken, zoekvelden, een calender en een categorieoverzicht.

Het is afhankelijk van het thema waar widget komen te staan. Sommige thema's hebben slechts één widgetlocatie, andere wel vier of meer.

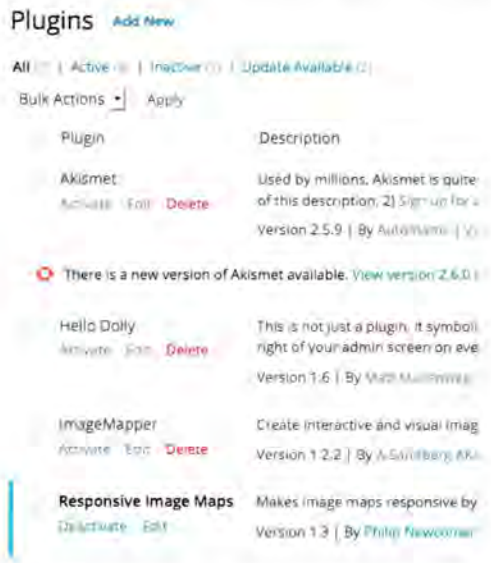
Via het adminpanel 'Appearance > Widgets' kun je je widgets beheren.

Plug-ins

Plug-ins voegen nieuwe mogelijkheden toe aan de WordPress installatie. Ken je de term 'There's an app for that'? Dat geldt ook zeker voor de tienduizenden WordPress plug-ins! Je kunt het zo gek niet bedenken of er is een plug-in voor.

Zo zijn er plug-ins voor slideshows, contactformulieren en calenders. Er zijn ook plug-ins voor volledige webshops, zoals de WooCommerce plug-in.

Via het adminpanel 'Plugins' kun je je plugins beheren, aan/uit zetten en toevoegen of verwijderen. Controleer voor het installeren of de plug-in is getest voor jou WordPress versie en lees altijd het commentaar van de plug-in developer.



Een aantal 'must-have' plug-ins die niet mogen ontbreken:

- Contact Form 7, Visual Form Builder (PRO) of Gravity Forms
- Remove Category URL
- Custom Post Type UI
- Advanced Custom Fields (ACF)
- Coming Soon Maintenance Mode
- Yoast SEO
- Resize before upload / Insanity
- iThemes Security
- Theme My Login
- Simple Page Ordering
- WP Example Content
- Easy FancyBox
- Easy Twitter Feed Widget
- WYSIWYG Widgets / Widget Blocks - WPML (Betaald)

Risico's van Plug-ins

Plug-ins brengen beveiligingsrisico's met zich mee. Doe altijd onderzoek naar de plug-in voordat je het installeert. Het aantal reviews, of de plug-in recent nog is geüpdatet en de compatibiliteit met jouw WordPress versie geven slechts een indicatie en bieden geen zekerheid of de plug-in qua beveiliging en techniek op orde is. Plug-ins maken je websites langzamer. Installeer ze alleen als het nodig is.

Hackerproof installeren

WordPress is geïnstalleerd op ±25% van alle websites (bron W3Techs) en daarom een populair hackerdoelwit. Geef voldoende aandacht aan onderstaande beveiligingstips om jouw installatie veilig te houden!

1. Zorg dat de remote webserver altijd SSL-encryptie gebruikt.

2. Maak op de remote webserver een nieuwe MySQL-database aan. Of maak met MAMP PRO een lokale virtual host aan en maak daarna met de bijgeleverde phpMyAdmin software een nieuwe database aan.

3. Verzin een willekeurige of cryptische naam voor de database name & user en zorg voor een ingewikkeld database password. Vergeet niet de database user te koppelen aan de database, maar geef de user niet alle database rechten. Alter, Create, Delete, Index, Select, Drop, Insert en Update zijn in de meeste gevallen voldoende.

4. Ga naar WordPress.org en download de nieuwste WordPress versie.

5. Unzip de download en stop de bestanden in de document root van de remote webserver (via FTP) of de virtual host.

6. Ga met een webbrowser, middels HTTP, naar de document root van de remote webserver of virtual host.

7. Doorloop het installatieproces met behulp van de aangemaakte databasegegevens:

De befaamde 'fool-proof' WordPress installatie.

- `Table prefix`: Standaard gebruikt WordPress `wp_` als prefix voor je databasetabellen, maar hackers weten dit ook! Gebruik daarom een willekeurige of cryptische prefix, zoals een klantnummer of een afkorting. Vergeet niet een underscore toe te voegen aan het einde van de prefix!

- Het e-mailadres is verplicht voor password recovery en e-mails die je ontvangt van de WordPress installatie (bijvoorbeeld over updates, webshopbestellingen of nieuw commentaar).

- Denk goed na over een mooie username, want je kunt de username van het admin account achteraf moeilijk aanpassen.

8. Nadat de installatiestappen zijn doorlopen, ga naar phpMyAdmin en klik op de WordPress database. Merk op dat tables de door jouw gekozen prefix hebben gekregen.

9. Check de `wp_users` table. Merk op dat de admin user ID #1 heeft gekregen. Hackers weten dit ook, dus is het verstandig om het ID aan te passen naar een willekeurig getal. Voer in phpMyAdmin de volgende SQL-commando's uit:

```
UPDATE wp_users SET ID = 342432 WHERE ID = 1;
UPDATE wp_usermeta SET user_id = 342432 WHERE
user_id = 1;
```

Nota Bene: ID '342432' is willekeurig gekozen en vergeet niet om de door jouw gehanteerde prefix te gebruiken in plaats van '`wp_`':

10. In de document root van je project open je het bestand 'wp-config.php', dat tijdens het installatieproces is aangemaakt. Halverwege het bestand vind je diverse salt-keys. Je kunt ervoor kiezen deze keys te vernieuwen door de bijhorende instructie te volgen.

Je kunt WordPress ook handmatig installeren door wp-config-sample.php te bewerken, voordat je stap 6 uitvoert. Vergeet het bestand na het bewerken niet te renamen naar wp-config.php.

11. Bestudeer het Codexartikel 'Hardening WordPress' en pas de tips toe.

12. Ga met een webbrowser, middels HTTP, naar jouwWebsiteUrl/wp-admin en log-in.

Back-ups maken

WordPress websites zijn een populair hackerdoelwit. Daarnaast kan jouw klant iets verkeerd doen of gaat de WordPress installatie stuk door verkeerd gebruik van plug-ins.

Het is daarom belangrijk regelmatig een back-up te maken.

Stap 1: database back-up

Maak met phpMyAdmin een backup van de WordPress database als SQL-bestandsformaat:

- Log-in in de phpMyAdmin omgeving van de webserver.
- Selecteer de WordPress database en te kies voor "Export".
- Selecteer het SQL bestandsformaat en sla het op je computer op.

Stap 2: WP-bestanden back-up

- Gebruik FTP-software om in te loggen op de webserver.
- Download de gehele document root. Je hebt álle WordPress files nodig!
- Let op: als er een .htaccess bestand aanwezig is, dan moet je deze ook downloaden.

  .htaccess is een verborgen bestand, dus het kan zijn dat je het standaard niet ziet staan. Stel de FTP-software in zodat je verborgen dot-bestanden kunt zien.

De WordPress installatie verhuizen

Als je je WordPress installatie wilt verhuizen naar een andere webserver of domeinnaam kun je onderstaande stappen doorlopen.

- Bestudeer eerst het Codexartikel 'Moving WordPress'.
- Open het gedownloade sql-bestand van 'stap 1' in een teksteditor.
- Met de 'find and replace' mogelijkheid van je teksteditor zoek je naar de gehele oude URL van je website en verander je het naar de nieuwe URL. Bijvoorbeeld: http://www.cmm.nl naar: http://www.nieuwesite.nl.
- Log-in in de phpMyAdmin omgeving van de nieuwe webserver.
- Maak een nieuwe SQL database aan volgens de security tips van het vorige hoofdstuk van deze reader.
- Selecteer de nieuw aangemaakte database en kies daarna voor "Import".
- Selecteer het SQL-bestand van je computer en upload het naar de WordPress database.
- Bewerk het wp-config.php bestand dat je tijdens 'stap 2' naar je computer hebt gedownload.
- Bewerk de volgende regels:

```
define('DB_NAME', 'database_name_here');
define('DB_USER', 'username_here');
define('DB_PASSWORD', 'password_here');
define('DB_HOST', 'localhost');
```

- Wijzig indien nodig de karakterset:

```
define('DB_CHARSET', 'utf8');
```

- Wijzig indien nodig de prefix voor de database tabellen:

```
$table_prefix = 'wp_';
```

- Bewerk de salt-keys volgens de instructies in het commentaar om de WordPress installatie extra veilig te maken:

```
define('AUTH_KEY', '');
define('SECURE_AUTH_KEY', '');
define('LOGGED_IN_KEY', '');
define('NONCE_KEY', '');
```

- Upload de WordPress bestanden van je computer naar de document root van de nieuwe webserver.

WordPress talen

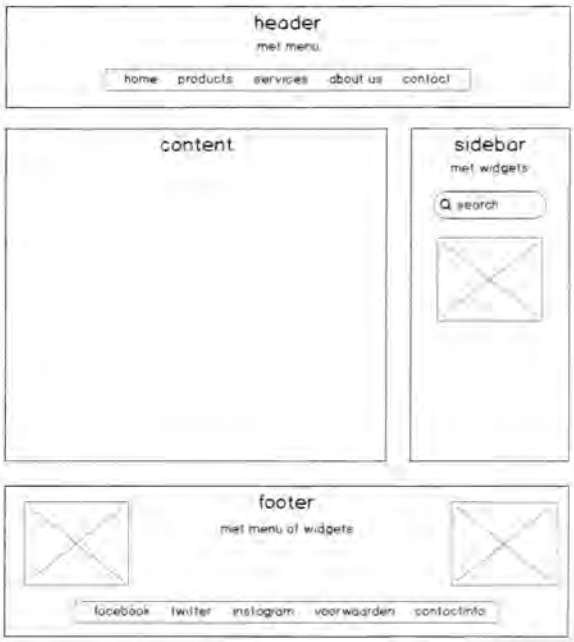
- Download de gewenste taal van WordPress.org.
- Plaats nu de nieuwe taalbestanden via FTP in de directory: `wp-content > themes > theme > languages`
- Open nu wp-config.php en pas de volgende regel aan: `define('WPLANG', 'nl_NL');`

Sinds WordPress 4.1 zijn er diverse talen opgenomen, die je via de instellingen van de WordPress back-end eenvoudig kunt instellen.

Themastructuur

WordPress thema's zijn allemaal opgebouwd volgens een vaste structuur. Ze bevatten meestal een header met navigatiemenu, een sidebar met ruimte voor widgets, een footer (al dan niet met ruimte voor een extra menu en / of widgets) en een contentgedeelte.

Onderstaande schets laat een veelgebruikte thema-structuur zien. De positie van de blokken is niet relevant, omdat dit per thema anders kan zijn.



WordPress Template Hierarchy

WordPress heeft een unieke manier om gegevens uit de database te 'mappen' naar het contentgedeelte van het thema.

Thema's worden opgesteld aan de hand van de zogenaamde 'template hierarchy'. Het bepaalt welke themabestanden worden gebruikt in een bepaalde situatie.

Dit principe wordt ook wel de 'WordPress Taxonomy' genoemd; de manier waarop WordPress gegevens groepeert. Je bent reeds bekent met twee manieren: de categorieën en de tags.

Themabestanden

Thema's worden geïnstalleerd in een eigen map in de directory `wp-content > themes`. Elk thema bestaat minimaal uit vier bestanden:

- **screenshot.png:** een schermafbeelding van het themadesign voor gebruik in de WordPress back-end `Appearance > Themes`.
- **index.php:** het hoofdbestand van het thema, waarin databasegegevens (zoals de posts) worden getoond.
- **style.css:** bevat alle CSS-code voor het thema.
- **functions.php:** een bestand waarin het standaard WordPressgedrag overschreven kan worden.

Meestal heeft het thema ook header.php, sidebar.php en footer.php bestanden. Deze bestanden worden ingeladen door index.php.

In header.php wordt met een WordPress function het navigatiemenu ingeladen. Op vergelijkbare manier worden Widgets ingeladen in sidebar.php of footer.php.

De meeste thema's bestaan uit veel meer bestanden, zoals bijvoorbeeld:

- 404.php
- archive.php
- category.php
- comments.php
- page.php
- search.php
- single.php

Een thema ontleden

Voordat je een thema kunt aanpassen is het cruciaal dat je de werking van de WordPress Hierarchy begrijpt. Het bepaalt welk themabestand wordt gebruikt in een bepaalde situatie.

Stel; je hebt twee categorieën aangemaakt en per categorie een aantal posts geschreven. Je hebt vervolgens de categorieën in het navigatiemenu gezet, zodat dat de gebruiker via het hoofdmenu naar de categorieën kan navigeren en de betreffende posts kan bekijken.

Vraag: Welk themabestand gebruikt WordPress om in het contentgedeelte van dat bestand het categorieoverzicht te tonen?

Om een antwoord te vinden op deze vraag gebruik je het WordPress Template Hierarchy schema.

Antwoord: Bevat het thema alleen index.php? Dan gebruikt WordPress index.php. Is archive.php ook aanwezig? Dan wordt dat gebruikt. Is category.php ook aanwezig? Dan gebruikt WordPress juist dat bestand.

CMS

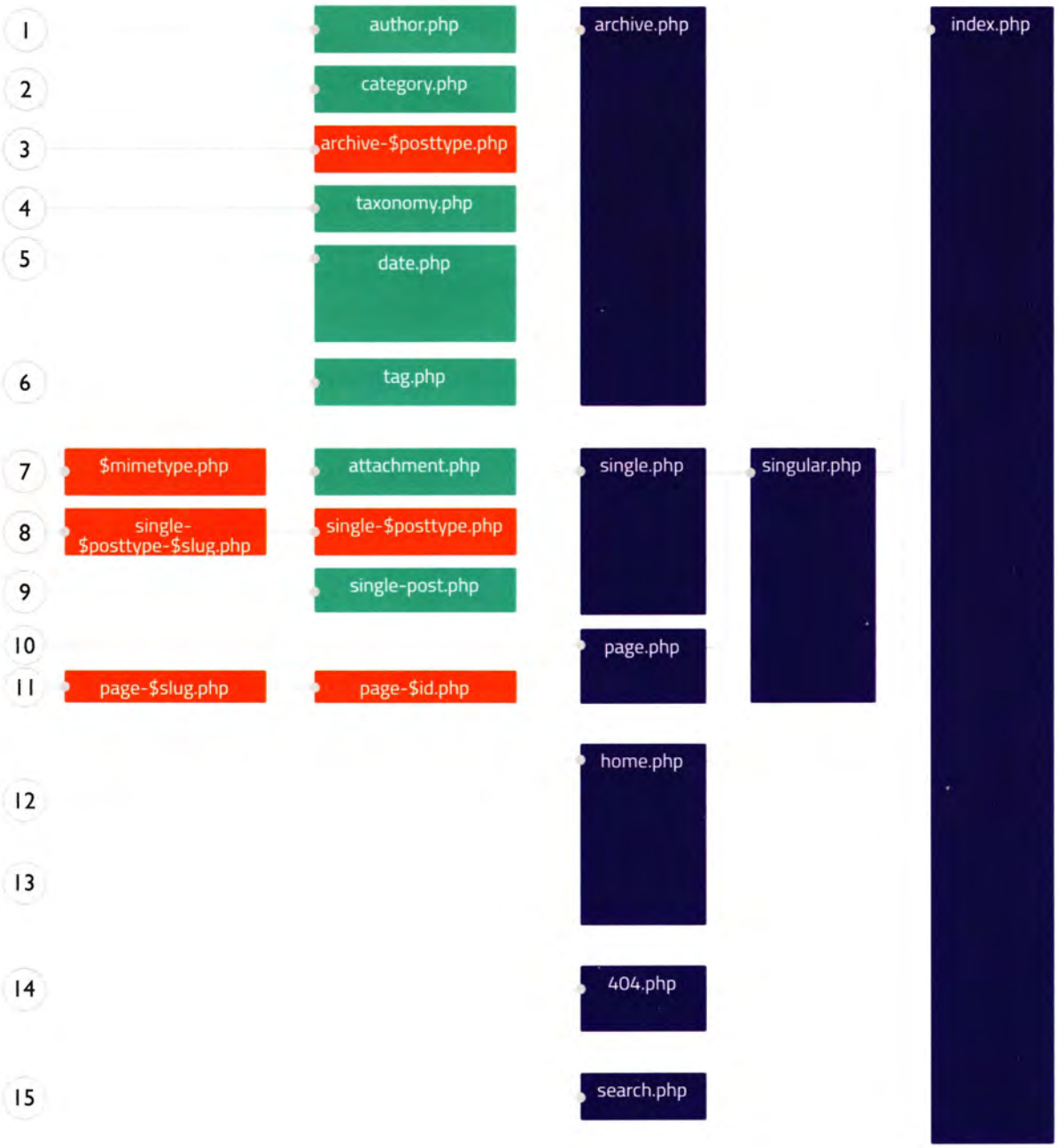
WordPress Template Hierarchy

Archive Page		Author Archive	author-$nicename.php	author-$id.php	1
		Category Archive	category-$slug.php	$category-$id.php	2
		Custom Post Type Archive			3
		Custom Taxonomy Archive	taxonomy-$taxonomy-$term.php	taxonomy-$taxonomy.php	4
		Date Archive	Year Archive		5
			Month Archive		
			Day Archive		
		Tag Archive	tag-$slug.php	tag-$id.php	6
Singular Page	Single Post Page	Attachment Post	$mimetype-$subtype.php	$subtype.php	7
		Custom Post		If selected: $custom.php	8
		Blog Post		If selected: $custom.php	9
	Static Page	Page Template	Custom Template	$custom.php	10
			Default Template		11
Site Front Page	front-page.php	Page Shown On Front			12
		Posts Shown On Front			
Blog Posts Index page					13
Error 404 Page					14
Search Result Page					15

Donker blauw: Primary Template
Groen: Secondary Template
Rood: Variable Template
Grijs: Page Type

WordPress Template Hierarchy

CMS



De Command Line Interface

Je bent gewend instructies te geven aan de computer via een muis en de GUI van het besturingssysteem.

Front-End Developers communiceren vaak op een rechtstreekse manier met hun systeem via de Command Line Interface (CLI). Zij vinden het prettiger tekstcommando's te geven, omdat de GUI hen in de weg zit.

Git BASH?

Windows heeft hiervoor de Command Prompt of de modernere PowerShell. MacOS heeft de Terminal.app.

De CLI van Windows. Hier wordt een verbinding gemaakt met een Linux computer (RaspberryPi) en software geïnstalleerd.

CLI op de Mac

MacOS is een UNIX systeem met daar omheen een mooie grafische schil (Quartz genoemd). Je gebruikt Terminal.app om het systeem tekstcommando's te geven (te vinden in Applications - Utilities).

Hieronder staan een aantal veelgebruikte UNIX-commando's.

`$` - het dollarteken achter jouw username geeft aan dat commando's worden uitgevoerd als normale user (met alle bijhorende rechten, beperkingen en mogelijkheden).

`#` - het hekje betekent dat commando's worden uitgevoerd als super user (root user met volledige systeemtoegang).

`sudo -s` - switch naar de super user. Sudo staat voor Super User DO.

`sudo commando` - voert 'commando' uit als super user. Wees altijd voorzichtig met sudo commando's!

`reboot` - de computer herstarten. Dit commando werkt alleen als je super user bent.

`shutdown -h now` - de computer uitzetten. Dit commando werkt alleen als je super user bent.

`exit` - De super user uitloggen en verder gaan als normale gebruiker.

`whoami` - laat zien welke user op dit moment met de CLI werkt.

`pwd` - staat voor 'print working directory'. Of te wel: het commando laat zien in welke directory je momenteel werkt.

`clear` - clear screen: het CLI window opschonen.

`↑ | ↓` - de cursor toetsen voor omhoog en omlaag laten recent gebruikte commando's zien.

`open 'p.app'` - in de huidige directory het programma 'p.app' openen.

`open .` - open de huidige directory in de Finder.

`touch 't.txt'` - in de huidige directory het bestand 't.txt' aanmaken.

`nano 't.txt'` - in de huidige directory het bestand 't.txt' aanmaken (of bewerken) met de nano-teksteditor.

Change directory

`cd 'mapnaam'` - staat voor 'change directory'. Met dit commando ga je vanuit je huidige directory (pwd) naar de subdirectory 'mapnaam'.

`cd ..` - ga vanuit je huidige directory naar de parent directory.

`cd /` - ga naar de document root van de harde schijf.

`cd ~/` - ga naar de home-directory van de huidige user.

List files

`ls` - laat een lijst zien met bestanden en directories die in de huidige directory staan.

`ls -a` - laat nu ook verborgen bestanden zien. Deze bestanden beginnen met een '.'

`ls -A` - laat bijna alle bestanden zien, behalve '.' en '..'

`ls -la` - gedetaileerde bestandsinformatie bekijken.

Make directory

`mkdir test` - maak een nieuwe directory 'test' in de huidige directory.

Remove

`rm t.gif` - delete het bestand genaamd 't.gif'.

`rmdir test` - haal de 'test'-directory weg. (alleen als de directory helemaal leeg is).

`rm -r test` - delete de 'test'-directory, inclusief alle files en directories die er in staan.

Move

`mv a b` - Het bestand 'a' hernoemen naar 'b'.

`mv a map/b` - Het bestand 'a' hernoemen naar 'b' én verplaatsen naar de directory 'map'.

NodeJS

NodeJS is een JavaScript runtime platform, gebaseerd op de V8 JavaScript engine van Google Chrome. Met NodeJS kun je JavaScript gebruiken zónder hulp van een browser! Je kunt met NodeJS:

- complete (web-)applicaties bouwen
- een lokale webserver opzetten (een testomgeving)
- repetieve taken automatiseren, zoals de browser automatisch herladen als je code wijzigt, sass compileren en js/css minifyen.
- de website uploaden naar een remote server (deployment genoemd).

Gebruik de installer van nodejs.org om NodeJs te installeren op jouw computer.

Package Managers

Een package manager is een tool voor automatisch softwarebeheer, zoals het installeren, updaten, configureren en verwijderen van software.

NodeJS gebruikt de NodeJS Package Manager (NPM). NPM wordt geïnstalleerd met NodeJS.

Packages

Softwareprogramma's worden packages of modules genoemd. Packages worden door de package manager gedownload van een centrale server (de repository).

De software wordt vervolgens automatisch geïnstalleerd en geconfigureerd.

Dependencies

Het komt vaak voor dat software afhankelijk is van andere software (dependencies genoemd). Packages bevatten informatie welke packages ze op hun beurt nodig hebben om te kunnen functioneren.

Installeer je via de package manager een video player, dan had je misschien nog niet de alle noodzakelijke CODECS geïnstalleerd, die de player wel nodig heeft. De package manager zorgt ervoor dat al deze zogenaamde 'dependencies' ook worden gedownload en geïnstalleerd.

Repositories

De repository is een webserver met een database van packages. Als je weet welke package manager je gebruikt, weet je ook welke repository je gebruikt. Voor NodeJs is dat npmjs.com.

Op de website van de repository kun je zoeken naar packages voor jouw project. Zoek op npmjs.com maar eens op naar:

- jQuery, jQueryUi en jQuery.easing
- Animate.css
- Aniview (jquery-aniview)
- Font Awesome (font-awesome)
- SCSS (node-sass)
- Bourbon
- Bourbon Neat (sag-bourbon-neat)

Voordelen van package managers

- Software is op één centrale plek te vinden. De NPM-community beschikt over duizenden packages! Je hoeft niet meer naar een website te gaan om handmatig software te downloaden en te installeren.
- Je kunt packages van andere developers gebruiken als dependency van jouw project. Ook kun je jouw project als package publiceren naar de repository, zodat het beschikbaar is voor anderen.
- Het wordt makkelijk om verschillende versies van software te beheren. De repository beschikt namelijk niet alleen over de laatste versie van een package.
- Software developers uploaden hun packages en updates zelf naar de repository. Je hoeft dus nooit zelf te gaan zoeken naar updates. De package manager kan al je software in één keer updaten.
- De repositorybeheerders checken packages, *vóórdat* jij ze kunt downloaden. Je weet dus zeker dat ze goed werken en niet schadelijk zijn voor je systeem.
- Omdat iedereen ter wereld bij de packages kan, kan ook iedereen de werking van de package valideren, controleren, uitbreiden en testen. De kans is vrijwel nihil dat er malware of spyware in gedownloade software zit.
- Dependencies hoef je niet zelf te beheren. Dat is juist de taak van de package manager.
- Als je software wilt verwijderen, kun je ervoor kiezen om ook de dependencies te verwijderen, zodat er geen onnodige files achterblijven op je computer. De package manager zorgt er dus voor dat je altijd een optimaal draaiend systeem hebt, zonder onnodige software!

Voorbeelden

Enkele repositories met hun eigen package manager:

- **RubyGems:** is de package manager van het Ruby framework. Packages worden 'gems' genoemd en de repository is beschikbaar via rubygems.org.

  Ruby is standaard geïnstalleerd op MacOS. Type in de Terminal `ruby -v` en `gem -v` om te controleren welke versies van Ruby en RubyGems jouw computer heeft.

  Type in de Terminal vervolgens `gem search sass` om de repository te doorzoeken. In de lijst zie je de Sass package, die je kunt gebruiken om SCSS-bestanden te compileren. Type `gem install sass` om de package te installeren. Je kunt daarna in de directory van jouw project het `sass --watch` commando geven. Bijvoorbeeld: `sass --watch input.scss:output.css`

- **Apt:** de Advanced Package Tool van het Linux Debian besturingssysteem.

- **NPM:** is de package manager van NodeJS. Packages zijn te vinden op de repository npmjs.com. Type in de Terminal `npm -v` controleren welke NPM-versie is geïnstalleerd.

Package.json

Nadat NodeJs is geïnstalleerd kun je een eigen project opzetten.

In de directory van jouw project geef je het commando `npm init` om een 'package.json'-bestand aan te maken. In het bestand wordt een JSON-object aangemaakt met diverse properties die NPM begrijpt.

De CLI vraagt om gegevens die je zo compleet mogelijk moet invullen, zoals onder andere:

- **name:** een korte, url-vriendelijke naam van project.

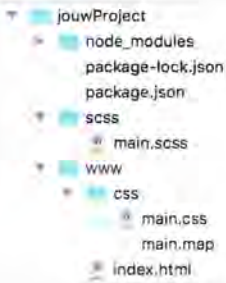
- **version:** de versie van je website of applicatie.

- **description:** een omschrijving van je project. Deze omschrijving is met name relevant als je je project uiteindelijk wilt publiceren naar een repository.

- **main:** is het entry point van je project. Met andere woorden: de root-directory van jouw project (de www-folder).

- **keywords:** een array met strings. Deze keywords helpen anderen jouw package te vinden in de repository.

Nadat deze stappen zijn doorlopen is jouw websiteproject een package geworden. Dit betekent dat NPM dependencies kan installeren die jij voor jouw project nodig hebt, zoals jQuery, animate.css en vele anderen!



Een eenvoudige directorystructuur.

Packages installeren

Stel; je wilt animate.css gebruiken. Zoek de package op npmjs.com of type: `npm find animate.css`

Nu je de package hebt gevonden kun je de laatste versie installeren met: `npm install --save-prod animate.css`

Bekijk na het installeren het 'package.json'-bestand. Zoals je ziet is animate.css toegevoegd als dependency. Het dakje (^) voor het versienummer geeft aan dat dit de minimale versie is dat jouw project nodig heeft om te functioneren. NPM houdt hiermee rekening als je het update commando geeft (`npm update`).

In de directory van je project is een 'node_modules'-map verschenen. NPM heeft Animate.css van de repository gedownload en in deze directory gezet.

Production en Developer Dependencies

De `--save-prod` parameter van het `install`-commando installeert de package als een 'production dependency'. Dit betekent dat de package tijdens de deployment fase mee wordt gepubliceerd, omdat de package als dependency natuurlijk een onmisbaar onderdeel is voor de werking van jouw software.

`--save-dev` installeert de package als 'devDependency.' Dit betekent dat de package bij deployment juist NIET mee wordt gepubliceerd. Je gebruikt `--save-dev` voor packages die jij lokaal nodig hebt om je project te kunnen maken, maar die niet noodzakelijk zijn voor de werking van je software. Een SCSS-compiler is hiervan een voorbeeld.

Geïnstalleerde packages bekijken

Na het installeren van een package vind je een automatisch gegenereerd bestand genaamd 'package-lock.json'. Het bestand beschrijft de aangemaakte directorystructuur en de geïnstalleerde packages. NPM houdt op deze manier overzicht en het kan commando's uitvoeren in de 'node_modules'-directory en in 'package.json'.

`npm list` is het commando om aan NPM te vragen welke packages zijn geïnstalleerd. Als er packages zijn geïnstalleerd die op hun beurt weer veel dependencies gebruiken, dan is `npm list --depth=0` vaak overzichtelijker.

Met `npm view animate.css version` controleer je welke versie van de animate.css package is geïnstalleerd.

Packages updaten

Met `npm outdated` kun je controleren of er verouderde packages zijn die geüpdated kunnen worden.

Update packages naar de laatste versie met `npm update`.

Packages publiceren

Als jouw project (je package) af is kun je het publiceren naar de repository van jouw bedrijf. Zo kunnen collega's jouw werk bekijken en nieuwe versies schrijven.

Heb jij die revolutionaire nieuwe versie van jQuery geschreven? Dan kun je de package natuurlijk ook ter review indienen bij NPM!

Package verwijderen

`npm uninstall animate.css` verwijdert de package.

Je hoeft de package manager niet te controleren, maar kijk even of de package is verwijderd uit package.json en uit de 'node_modules'-directory.

SCSS gebruiken met nodeJS

Installeer de volgende packages als devDependencies om te kunnen werken met SCSS:

```
npm install --save-dev node-sass
npm install --save-dev nodemon
```

Nodemon houdt een directory in de gaten en kan vervolgens automatisch een commando uitvoeren als bestanden in deze directory veranderen.

In ons geval moet nodemon de bestanden van de scss-directory in de gaten houden. Zodra een bestand verandert geeft nodemon de opdracht aan node-sass om te compileren. Bewerk hiervoor de scripts-property in de package.json:

```
"scripts": {
    "build-css": "node-sass --include-path scss
    scss/main.scss www/css/main.css --source-map
    www/css/main.map",

    "watch-css": "nodemon -e scss -x \"npm run
    build-css\""
}
```

Nota Bene!
Dit codevoorbeeld gaat uit van de directorystructuur afgebeeld in de vorige pagina.

Je kunt de scss nu zelf éénmaal compileren met `npm run build-css`. Je kunt het nodemon laten doen met `npm run watch-css`. Stop nodemon met de toetscombinatie CTRL-C.

Browser Sync

Browser Sync is een nodeJS package dat alle browserschermen automatisch ververst zodra jij een html, css of js bestand wijzigt. Dit bespaart een hoop tijd en zorgt ervoor dat je makkelijker websites kan testen! Installeer eerst de browser-sync package:

```
npm install --save-dev browser-sync
```

Bewerk vervolgens de scripts-property in de package.json en voeg het volgende commando toe:

```
"start": "browser-sync start --server --files '**/*.
css, **/*.html, **/*.js, !node_modules/**/*'
--directory --port 7777 --browser 'Google Chrome'"
```

Om Browser Sync te starten geef je het commando:

```
npm run start
```

Het commando start een lokale webserver, bereikbaar via poort 7777. Browser Sync houdt alle .css, .html en .js bestanden in de gaten, maar NIET de bestanden in de 'node_modules'-directory. Google Chrome moet worden geladen met een directorylisting van je project.

Om meerdere browsers tegelijk te gebruiken kopieër je de url van de Chrome adresbalk en plak je het in een andere browser. Scroll je nu één van de browsers dan scrollen alle andere browsers automatisch mee, ook als je dat op een telefoon op tablet doet!

College of MultiMedia

De inhoud van deze reader mag niet zonder uitdrukkelijke toestemming van College of MultiMedia worden gereproduceerd, gekopieërd of worden gebruikt voor andere doeleinden dan waarvoor College of MultiMedia deze heeft bedoeld.

Deze reader is samengesteld met eigen lesmaterialen van College of MultiMedia.
Bij gebruik van ander materiaal is de bron bij het desbetreffende onderwerp vermeld.

© 2019 - College of Multimedia