**Section 2 – HTML**

**Les 12 – Structure of a webpage**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<p>Here is some text</p>

<p>Here is some more text</p>

<p>Here is some text <br> with a line break</p>

</body>

</html>

**Les 14 – Headers**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<h1>This is a big header</h1>

// Door keywords in een <h1> op te nemen wordt je site beter gevonden op Google

<h2>This is quite a big header</h2>

<h3>This is an h3 header</h3>

<h4>This is an h4 header</h4>

<h5>This is an h5 header</h5>

<h6>This is an h6 header</h6>

<h7>This is NOT an h7 header</h7>

// <h7> bestaat niet, wordt dus als gewone tekst afgebeeld

**Les 16 – Paragraph tags**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<p>Here is some text</p>

<p>Here is some more text</p>

<p>Here is some text <br> with a line break</p>

</body>

</html>

</body>

</html>

De browser herkent geen line-breaks in html tekst, daarvoor moet je de tekst in afzonderlijke paragrafen zetten of een <br> in de tekst opnemen.

NB De <br> is self closing, <br /> is alleen nodig in XHTML, de strikte variant van html.

**Les 18 – Formatting text**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<p>Here is some <strong>bold</strong> text</p>

// <b>bold</b> doet exact hetzelfde maar <strong> is gebruikelijk

<p>Here is some <em>italic</em> text</p>

// <i>italic</i> doet exact hetzelfde maar <em> is gebruikelijk (em = emphasis)

<p>Here is some <ins>underlined</ins> text </p>

// u>underline</u> doet exact hetzelfde maar <ins> is gebruikelijk (ins = inserted)

<hr>

// laat een horizontale regel zien, bijv. om delen van de pagina optisch te scheiden.

<p>Here is some <sup>superscript</sup> text</p>

<p>Here is some <sub>subscript</sub> text</p>

<p>Here is some <del>deleted</del> text</p>

</body>

</html>

**Les 20 – Unordered lists**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<ul>

<li><strong>Rob</strong></li>

<li><em>Kirsten</em></li>

<li><del>Tommy</del></li>

<li><ins>Ralphie</ins></li>

</ul>

</body>

</html>

**Les 22 – Ordered lists**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<ol type="a">

<li><strong>Rob</strong></li>

<li><em>Kirsten</em></li>

<li><del>Tommy</del></li>

<li><ins>Ralphie</ins></li>

</ol>

</body>

</html>

De ordered list kent nog meer opties:

* <ol start=10> start de lijst met index 10
* <ol reversed> begint met het hoogste nummer, eindigt met het laagste
* <ol type=”I”> gebruikt Romeinse cijfers

Zie voor meer mogelijkheden w3schools.com.

**Les 24 – Images**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<img src="https://upload.wikimedia.org/wikipedia/en/0/0b/Marge\_Simpson.png" width="100" height="100" align="left">

// Align=”left” zorgt dat het plaatje links van de tekst wordt getoond.

<p>Marge Simpson</p>

</body>

</html>

In dit voorbeeld wordt het plaatje ‘ge-hotlinked’, een directe verwijzing naar een andere site. Dit is niet verstandig omdat je geen controle hebt over die site; het plaatje zou van locatie kunnen veranderen of de toegang zou je ontzegd kunnen worden. Het wordt niet gewaardeerd omdat als jouw pagina veel bezoek heeft, dat ook voor de pagina met het plaatje geldt en die site dus veel nutteloos verkeer moet verwerken.

Dus alle plaatjes lokaal opslaan en met <src=”homer.png”> opvragen.

**Les 26 – Forms**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<form>

// Alle invoervelden binnen de form worden bij de submit verzonden

<p>Username: <input type="text" placeholder="your username"></p>

// Een standaard tekst input field

<p>Stay logged in: <input type="checkbox" checked></p>

// Checkbox: aan-afvinken

<p>Under 18: <input type="radio" name="age" value="u18"></p>

<p>Over 18: <input type="radio" name="age" value="o18"></p>

// Door dezelfde naam te gebruiken worden de radio buttons

// gegroepeerd. Dan kan binnen de groep slechts één button aangevinkt

// zijn, de vorige aangevinkte wordt automatisch uit gezet.

<p>Favourite food:

<select>

// Dit start een dropdown box

<option>Pizza</option>

<option selected>Ice Cream</option>

<option>Sandwiches</option>

// 3 opties, met ‘selected’ wordt de initiële keuze voorgeselecteerd

</select>

</p>

<p><input type="submit" value="Click me!"></p>

// Met de submit knop worden de input fields verstuurd

</form>

</body>

</html>

**Les 28 – Tables**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<table>

<thead>

// De <thead> is feitelijk overbodig maar staat wel zo netjes

<tr>

// tr = table row

<th>Name</th>

// th = table header

<th>Favourite<br>Colour</th>

</tr>

</thead>

<tr>

<td>Rob</td>

<td>Green</td>

// td = table data

</tr>

<tr>

<td>Kirsten</td>

<td>Orange</td>

</tr>

<tr>

<td>Tommy</td>

<td>Pink</td>

</tr>

<tr>

<td>Ralphie</td>

<td>Blue</td>

</tr>

</table>

</body>

</html>

Rob heeft de <tbody> uit de code weggelaten. Die is ook niet nodig maar staat wel zo netjes als je de header met een <thead> hebt aangegeven.

**Les 30 – Links**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

<p id="top"><a href="http://www.google.com">Click here to go to google.com</a></p>

<p><a href="http://www.wikipedia.org"><img src="homer.png"></a></p>

<p><a href="helloworld.html">Hello world!</a></p>

<p><a href="http://www.wikipedia.org"><img src="homer.png"></a></p>

<p id="thirdhomer"><a href="http://www.wikipedia.org"><img src="homer.png"></a></p>

<p><a href="http://www.wikipedia.org"><img src="homer.png"></a></p>

<p><a href="http://www.wikipedia.org"><img src="homer.png"></a></p>

<p><a href="#top">Back to top</a></p>

<p><a href="#thirdhomer">Go to third homer</a></p>

// Ga naar de paragraph op de huidige pagina met id ‘thirdhomer’

</body>

</html>

Het gebruik van relatieve links is handig voor als je je website wil verplaatsen, mits je uiteraard de structuur gelijk houdt dan blijft de site goed werken, zonder gebroken links.

**Les 32 – HTML Entities (weergave speciale karakters)**

<html>

<head>

<title>My Webpage</title>

</head>

<body>

&lt;a href="http://google.com"&gt;

</body>

</html>

Zie voor een komplete lijst van speciale karakters op <https://dev.w3.org/html5/html-author/charref>.

**Les 34 – iFrames**

<!doctype html>

<html>

<head>

<title>Hello World!</title>

<meta charset="utf-8" />

<meta http-equiv="Content-type" content="text/html; charset=utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1" />

</head>

<body>

<iframe width="560" height="315" src="https://www.youtube.com/embed/DrDm7uO4Fu0?rel=0" frameborder="0" allowfullscreen align="right"></iframe>

</body>

</html>

Let op dat niet alle sites toestaan dat je hun content in een iframe weergeeft. Google is een voorbeeld.

**Section 3 – CSS**

**Les 49 – Classes and ID’s**

Id’s mogen slechts één keer per pagina gebruikt worden, classes onbeperkt. Een class gebruik je als elementen eigenschappen met elkaar delen, een id als de eigenschappen uniek voor een bepaald element zijn. Een voorbeeld hiervan zijn vierkanten die dezelfde afmetingen hebben maar een afwijkende kleur. De afmetingen specificeer je in een class, de kleuren in id’s:

.square {

width: 100px;

height: 100px;

}

#red-square {

background-color: red;

}

#blue-square {

background-color: blue;

}

#green-square {

background-color: green;

}

De code uit de les:

<html>

<head>

<title>Webpage With Style</title>

<style type="text/css">

.red {

color:red;

}

// NB Dit betreft de tekstkleur! Opvulkleur geef je aan met

// background-color.

.large {

font-size: 200%;

}

#green {

color:green;

}

.underlined {

text-decoration: underline;

// tekst-decoration attribuut!

}

</style>

</head>

<body>

<p class="red large">The quick brown fox jumped over the lazy dog.</p>

<p class="large">Wow, I love internal CSS!</p>

<p id="green">This is some more text. <span class="underlined">And this text is underlined.</span></p>

// met <span> specifieke stukken tekst een andere style geven

<h1 class="red">CSS is cool!</h1>

</body>

</html>

**Les 51 – DIV’s**

Let op dat div’s altijd de volledige breedte van hun parent-element in beslag nemen, op het hoogste niveau dus de gehele pagina. Dit komt door het attribuut ‘**display’** dat default de waarde ‘**block’** heeft. Zet je ‘display’ op ‘**inline’** dan neemt de div alleen de benodigde ruimte in, dus bijv. de breedte van een erin opgenomen string. Een andere optie is ‘**inherit’** waar het de display waarde van de parent overneemt.

<html>

<head>

<title>Webpage With Style</title>

<style type="text/css">

.red {

color:red;

}

.large {

font-size: 200%;

}

#green {

color:green;

}

.underlined {

text-decoration: underline;

}

#first-section {

color:blue;

background-color: pink;

width:100px;

}

// background attribuut is een trucje om uit te vinden hoe groot de <div>

// werkelijk is

#second-section {

background-color: yellow;

}

</style>

</head>

<body>

<div id="first-section">

<p>The quick brown fox jumped over the lazy dog.</p>

<p>Wow, I love internal CSS!</p>

</div>

<div id="second-section">

<p id="green">This is some more text. <span class="underlined">And this text is underlined.</span></p>

<h1 class="red">CSS is cool!</h1>

</div>

</body>

</html>

**Les 53 – Colors**

Met HTML color codes kunnen kleuren nauwkeurig bepaald worden. Een code bestaat uit 6 hexadecimale cijfers, aangegeven met bijv. #1A2CF8. Gebruik een color picker voor alle mogelijkheden. Zie ook <html-color-codes.info>. Ik gebruik de color picker die in Chrome is ingebouwd of de **Digitale Kleurenmeter**, een extern programma.

**Les 55 – Floating**

<html>

<head>

<title>Webpage With Style</title>

<style type="text/css">

.red {

color:red;

}

.large {

font-size: 200%;

}

#green {

color:green;

}

.underlined {

text-decoration: underline;

}

#first-section {

color:#4285F4;

background-color: #CEF6CE;

width:200px;

float:left;

}

// left-floated content komt links van non-floated content terecht.

#second-section {

background-color: #8AB4F7;

width:200px;

float: right;

}

// right-floated content komt rechts van non-floated content terecht.

// het element wordt rechts op de pagina afgebeeld. Maak je de pagina

// kleiner dan beweegt het mee naar links (‘floating’), maar blijft rechts

// staan. Tenzij er geen plek is, dan verschuift het naar beneden, zonder

// dat er een witruimte zit tussen de div’s. Dit komt doordat de <p>

// standaard een margin heeft. Zet je die op 0, dan verdwijnt de

// witruimte.

.clear-float {

clear:both;

}

// Met clear:both worden vanaf het element die dit toepast de float

// instellingen ahw beëindigd en met een schone lei begonnen. Het

// effect is dat de elementen erna weer onder de eerdere floating

// elementen terecht komen, met een wit ruimte wel te verstaan.

</style>

</head>

<body>

<div id="first-section">

<p>The quick brown fox jumped over the lazy dog.</p>

<p>Wow, I love internal CSS!</p>

</div>

<div id="second-section">

<p id="green">This is some more text. <span class="underlined">And this text is underlined.</span></p>

<h1 class="red">CSS is cool!</h1>

</div>

<div class="clear-float"></div>

// De content na deze div komt direct onder de onderste floating div

// terecht

<p>This paragraph is not floated!</p>

// Non-floated content trekt zich niets aan van de floated content en

// zoekt als het ware een vrij plekje op.

</body>

</html>

**Les 57 – Positioning**

Met attribuut **position** op **relative** kun je een element een positie geven relatief tov de default positie. Dit doe je met de attributen

left: 50px; // beweeg 50 px naar **rechts** tov default positie

right: 50px; // beweeg 50 px naar **links** tov default positie

top: 50px; // beweeg 50 px naar **beneden** tov default positie

bottom: 50px; // beweeg 50 px naar **boven** tov default positie

Denk alsof je een margin toevoegt in de aangegeven richting. Merk op dat de content als het ware verschoven wordt en ook eventueel andere content kan overlappen. Dit kun je met een **z-index** ondervangen, het element met de hoogste z-index komt boven, mits het **position** attribuut **relative** is (maar met een paar andere werkt het ook).

De onderliggende content kun je enigszins laten zien als de bovenliggende content een **opacity** waarde heeft < 1. Waarde 1 is ondoorzichtig, waarde 0 is volkomen doorzichtig en daarmee onzichtbaar. Alles ertussen laat zowel de boven- als onderliggende content zien.

Merk ook op dat top: 50 px en bottom: -50 px op het zelfde neerkomen. Hetzelfde geldt ook voor **left** en **right**. In feite heb je right en bottom niet eens nodig, door left en top een positieve of negatieve waarde te geven kun je alle kanten op met het element.

Je kunt een element zo ook van het scherm laten verdwijnen. Dit wordt vaak toegepast als het element via een animatie op het scherm terecht komt.

Met het **position** attribuut op **absolute** haal je het element als het ware uit de flow van de pagina. De andere elementen krijgen dan een positie alsof dit absoluut gepositioneerde element niet bestaat, waardoor ze eronder of erboven kunnen komen.

Met left, right, bottom en top kun je het element een positie geven op dezelfde wijze als hierboven beschreven, **maar dan ten opzichte van zijn parent element**! Heeft het element geen parent, dan wordt de positie ten opzichte van de document body bepaald.

Het **position** attribuut **fixed** haalt het element ook uit de flow van de pagina en geeft het een vaste positie op het scherm. Bij scrollen blijft het element op dezelfde plek staan. Dit is bijv. handig als je een menu bovenaan het scherm hebt en er wil laten staan, hoever er ook gescrold wordt.

**Les 59 – Margins**

Deze gebruik je als je de elementen in de flow van de pagina wil houden en afstand tot andere elementen wil creëren, bijv:

margin: 100px 50px 30px 10px; // volgorde: top-right-bottom-left

In principe kun je met het position attribuut hetzelfde effect bereiken, maar het verschil is dat je het element – mits position niet relative – uit de flow van de pagina haalt.

**Les 61 – Padding**

Hiermee creëer je ruimte van een element binnen een ander element, bijv. tekst in een <div> geef je ruimte ten opzichte van de rand van de div door de padding op te geven. Padding werkt dus binnen een element, waar de margin erbuiten werkt.

Let op dat wanneer je de width of height van een element specificeert, de padding waarden daar nog boven op komen. Voorbeeld:

#second-section {

color: red;

background-color: blue;

width: 400px;

padding: 5px 10px 15px 20px;

}

De breedte van het element met id ‘second-section’ wordt daarmee 430 pixels (400+10+20).

Kijk je naar het resultaat van de code op het scherm dan zie je witruimten aan alle kanten van de elementen. De diverse browsers hanteren standaard (verschillende) witruimten voor diverse elementen. Wil je dat niet dan kun je bijv.

body, p {

margin: 0px;

padding: 0px;

}

in de pagina opnemen. Dit wordt ook wel een CSS Reset genoemd. Omdat de browsers verschillende regels er op nahouden wat betreft de default styling van standaard HTML elementen zoals <body> en <p>, kan de CSS Reset verschillen per browser. Google op CSS Reset voor meer info hierover.

**Les 63 – Borders**

Een border kent in principe 3 opmaakattributen: style, color en width. Deze kunnen per zijde – uitgaande van 4 zijden – ingesteld worden. Gebruik je voor alle 4 de zijden dezelfde attribuutwaarden dan heb je één regel code nodig:

#square-red {

background-color: red;

border: 10px yellow solid;

}

Hier specificeer je achtereenvolgens de width, color en style. Het kan ook per attribuut:

#square-red {

background-color: red;

border-style: ridge solid dotted dashed;

border-color: green beige pink palegoldenrod;

border-width: 10px 5px 2px 1px;

}

Hier specificeer je achtereenvolgens de top – right – bottom – left border. Je kunt elk attribuut expliciet per zijde specificeren, bijv:

border-left-style: solid;

border-right-color: black;

border-top-width: 3px;

De borders zijn tot nu toe rechthoekig. Mooier is om de hoeken af te ronden door de border-radius aan te geven:

border-radius: 20px;

Op deze manier kun je een **cirkel** maken, door voor de border-radius de helft van de width en height te nemen. Dit kun je in pixels uitdrukken, maar beter is het om een percentage te gebruiken, dat schaalt automatisch mee mochten width en height veranderen:

border-radius: 50%;

**Les 65 – Fonts**

Gebruik voor fonts **web safe fonts** die door de meeste browsers worden ondersteund. Google hier op, je vindt er vele op <w3schools.com>. Het font geef je aan met het font-family attribuut:

font-family: “Times New Roman”, times, serif

Hier zitten de fallback fonts in verwerkt, mocht de browser bijv. **Times New Roman** niet ondersteunen dan valt hij in eerste instantie terug op **times** en daarna op **serif**. Je kunt zo meer dan 3 fonts opgeven.

Het aantal web safe fonts is niet heel groot. Wil je wat anders dan kun je **Google fonts** proberen. Hier staan er een enorm aantal. Je voegt ze in met een link naar een extern style sheet dat Google opgeeft:

<html>

<head>

<title>Webpage With Style</title>

<link href='https://fonts.googleapis.com/css?family=Shadows+Into+Light' rel='stylesheet' type='text/css'>

<style type="text/css">

p {

font-family: 'Shadows Into Light', cursive;

}

</style>

</head>

<body>

<p>This is some text!</p>

</body>

</html>

**Les 67 – Styling text**

Zie voor een uitgebreid overzicht aan tekst styling mogelijkheden <w3schools.com>.

<html>

<head>

<title>Webpage With Style</title>

<style type="text/css">

.bold {

font-weight: bold;

}

.italic {

font-style:italic;

}

.underlined {

text-decoration: underline;

}

a {

text-decoration:none;

}

a:hover {

text-decoration:underline;

color:green;

}

// Wanneer je over de link (<a>) hovered dan wordt de link

// onderstreept en groen

.big {

font-size: 200%;

}

</style>

</head>

<body>

<p class="bold">This is some bold text!</p>

<p class="italic">This is some italic text! <span class="bold">And this is bold and italic.</span></p>

// De tekst in de span is zowel italic als bold, italic was hij al vanwege de

// “italic” class en in de span komt daar nog eens “bold’ bij.

<p class="underlined">This text is underlined!</p>

<a href="">This is a non-underlined link</a>

<p class="big">This text is BIG!</p>

</body>

</html>

**Les 69 – Aligning text**

Naast de alignment geeft Rob een tip om een website van test-tekst te voorzien, voor het geval er nog geen tekst beschikbaar is voor de te bouwen site. Google hiervoor op **lorem ipsum**. Je krijgt dan Latijnse tekst die er beter uitziet dan het 1000 herhalen van dezelfde string.

<html>

<head>

<title>Webpage With Style</title>

<style type="text/css">

p {

text-align:right;

}

// NB Default is left aligned, hoef je dus niet op te geven.

div {

width:300px;

background-color:green;

}

.centered {

text-align: center;

}

.justified {

text-align: justify;

}

// Hiermee vul je elke regel op met tekst waardoor je een linker en

// rechter kantlijn krijgt.

</style>

</head>

<body>

<div>

<p>This is some right aligned text!</p>

<p class="centered">This text is centered.</p>

<p class="justified">Lorem ipsum dolor sit amet, scripta adversarium definitiones duo an, sed minim facilisi ei. Appareat dissentias honestatis ut duo. Recusabo deseruisse ex sea, ne veri saperet placerat mea. Rebum offendit id mel, affert nonumes has cu, iudico veritus vituperatoribus no mea. Ut has scaevola singulis, percipitur liberavisse sed ut. Idque copiosae sea ea. Elit tritani percipit ut mei, ne eros rationibus eam, et cum congue malorum partiendo. </p>

</div>

</body>

</html>

**Les 71 – Styling links (en pseudo classes)**

Open link in een nieuwe tab door target=”\_blank” op te geven:

<a href=”<http://www.google.com>” target=”\_blank”>

Google in new tab

</a>

Let op dat als je [**http://www**](http://www) weglaat dat de link relatief is en dus op de client moet staan.

De standaard styling van een link is dat deze blauw en onderstreept is. Onderstrepen is niet in de mode, dus laat je die weg:

a {

text-decoration: none;

}

Dan kun je met behulp van zg. pseudo classes de styling aanpassen. Een pseudo class geeft een toestand van het element weer. Zie <w3schools.com> voor de nodige voorbeelden. Een aantal voorbeelden:

a:hover {

text-decoration: underline;

}

// Hoveren over de link toont de link onderstreept

a:visited {

text-decoration: none;

color: green;

}

// Alle reeds eerder bezochte links worden in groen en niet onderstreept] // weergegeven.

NB. Volgens w3schools.com moet **hover** na **visited** komen en moet **active** na **hover** komen anders zijn de pseudo classes niet effectief.

**Les 73 – 81 – Challenge: BBC Website clone**

De beschrijving van hoe de website is gebouwd is in de video’s en de HTML file terug te vinden. Ik beschrijf hier alleen de bijzonderheden.

**Div centreren op pagina**

In het geval een width voor een div is opgegeven – anders beslaat hij normaliter de volledige breedte (display attribuut = block) – dan kun je dit bereiken door de margin voor left en right op auto te zetten:

#topbar {

margin: 10px auto;

}

**Font**

Om erachter te komen welk font is gebruikt op een site, kun je in Chrome op het betreffende element rechtsklikken en vervolgens Inspect selecteren. Er verschijnt dan een scherm met allerlei technische info over de pagina. Rechtsonder in deze pagina staat een gekleurd vierkant met info over de afmetingen. Daaronder is info te vinden over de gebruikte fonts.

**Search icon**

Deze kopieert Rob van de BBC site. Om het klikbaar te maken maakt Rob er geen <img> van maar een <input> met als source het image van het gedownloade vergrootglas.

**Les 78 – External CSS**

Het is handig om de styling code apart op te slaan. Dit maakt de pagina overzichtelijker voor developers. Je verwijst als volgt naar externe CSS:

<link rel=”stylesheet” type=”tekst/css” href=”bbc.css”>

**Section 12 – HTML5 and CSS3 Special Features**

**Les 244 – What is HTML5?**

HTML5 voegt integratie van audio en video toe aan HTML. Tevens biedt het meer mogelijkheden t.a.v. forms handling.

Op de site <caniuse.com> is te zien in hoeverre de HMTL5 features dor de diverse browsers worden ondersteund.

**Les 246 – HTML5 Form Features**

Een aantal nieuwe features tav form handling in HTML5.

<!DOCTYPE html>

<html>

<head>

<title>HTML 5 and CSS 3</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no, minimal-ui">

</head>

<body>

<form id="myForm" action="processform.php">

<p>Placeholder: <input type="text" placeholder="Insert Name Here"></p>

<p>Autofocus: <input type="text" autofocus></p>

// Zet de focus op dit invoerveld (= blauw omkaderd). Het invoeren start

// in dit invoerveld, ook als dat niet de bovenste is. Het eerste autofocus

// invoerveld in de code krijgt de focus.

<p>Autocomplete Off: <input type="text" autocomplete="off"></p>

// Voorkomt dat een eerder ingevoerde waarde wordt getoond, handig

// voor bijv. invullen van een creditcardnummer. NB. ‘on’ is de default.

<p>Required: <input type="text" required></p>

// Veld mag niet leeggelaten worden, anders volgt een standaard

// melding na indrukken submit knop. Dit kan natuurlijk ook met jQuery

// of php afgevangen worden met een specifieke foutmelding.

<p>Email: <input type="email"></p>

// Checkt of input aan email formaat voldoet, anders volgt een standaard

// foutmelding

<p>Pattern: <input type="text" pattern="[0-9]"</p>

// checkt of input aan opgegeven patroon voldoet, anders volgt een

// standaard foutmelding.

// Kijk voor info over reguliere expressies op <html5pattern.com>.

<datalist id="names">

<select>

<option value="Rob">Rob</option>

<option value="Kirsten">Kirsten</option>

<option value="Tommy">Tommy</option>

<option value="Ralphie">Ralphie</option>

</select>

</datalist>

<p>Autocomplete: <input type="text" list="names"></p>

// Een bijzondere vorm van autocomplete die gebruik maakt van de

// opgegeven lijst. Vul je bijv. een ‘R’ in, dan zal Rob verschijnen. Je kunt

// ook uit de dropdown lijst selecteren.

// De ‘select’ is hier niet nodig, voegt Rob om onduidelijke reden toe.

<p>Multiple file upload: <input type="file" multiple></p>

// Druk op dit inputfield en je krijgt een file-selector window. Met ‘multiple’

// kun je meer dan één file tegelijk selecteren.

<p><input type="submit" value="Submit to processform2.php" formaction="processform2.php" formmethod="post"></p>

// Dit inputfield kent een eigen verwerking, doet dus niet mee met de

// standaard action van deze form. Per inputfield kun je ook de mehode

// (get of post) opgeven.

</form>

<p>Code outside form: <input form="myForm" type="text"></p>

// Hiermee kun je een input field **buiten** een form definieren, in pre-HTML5

// moesten alle velden binnen de <form> gedefinieerd zijn om ze na submit

// mee te nemen. Let wel dat “myForm” als id van de form gebruikt wordt!

</body>

</html>

**Les 248 – Audio & Video**

Vóór HTML5 moest je via een plugin audio en video afspelen.

Gebruik mp4 voor video en mp3 voor audio. Deze formaten worden door alle browsers ondersteund.

<!DOCTYPE html>

<html>

<head>

<title>HTML 5 and CSS 3</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no, minimal-ui">

</head>

<body>

<video width="320" height="240" controls id="video">

// Altijd width en height opgeven, anders kan beeld flikkeren

// optie ‘controls’ voegt controle knoppen play, pause etc toe, zonder

// ‘controls’ wordt de video automatisch afgespeeld

<source src="video.mp4" type="video/mp4">

Your browser does not support HTML5. Please upgrade your browser!

// Deze tekst wordt getoond als de <video> tag niet wordt ondertsteund

// door de browser

</video>

<button id="play">Play</button>

// Ook dmv javascript kan de video afgespeeld worden, mits deze een

// id krijgt (‘video’ in dit geval).

<audio controls>

// optie ‘controls’ voegt controle knoppen play, pause etc toe

<source src="audio.mp3" type="audio/mpeg">

Your browser does not support HTML5. Please upgrade your browser!

// Deze tekst wordt getoond als de <audio> tag niet wordt ondertsteund

// door de browser

</audio>

<script>

document.getElementById("play").onclick = function() {

if (document.getElementById("video").paused) {

// paused() functie checkt of video gepauseerd is

document.getElementById("video").play();

this.innerHTML = "Pause";

// Speel video af dmv play() functie en zet string in knop op ‘pause’

} else {

document.getElementById("video").pause();

this.innerHTML = "Play";

// Stop video dmv pause() functie en zet string in knop op ‘play’

}

}

</script>

</body>

</html>

**Les 250 – Canvas**

Met <canvas> kun je in basis op het scherm tekenen en meer geavanceerd 2D en 3D games spelen.

<!DOCTYPE html>

<html>

<head>

<title>HTML 5 and CSS 3</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no, minimal-ui">

<style>

#myCanvas {

border: 1px solid grey;

}

</style>

// Styling voor canvas, anders zie je nl alleen maar wit (niets dus)

</head>

<body>

<canvas id="myCanvas" width="500" height="400">

</canvas>

// Meer html niet nodig, de logica zit in de javascript

<script>

var myCanvas = document.getElementById("myCanvas");

var context = myCanvas.getContext("2d");

// 2d of 3d context definieren

var gradient = context.createLinearGradient(150, 100, 350, 300);

// definieer begin- en eindcoordinaat voor kleurenverloop, hier van

// (150,100) start tot 350 px breed en 300 px hoog als stop punt.

gradient.addColorStop(0, "white");

gradient.addColorStop(1, "#49C900")

// begin met wit op start (=0) en eindig met iets groens op stop (=1)

context.fillStyle = gradient;

context.fillRect(150, 100, 200, 200);

// eerst fillstyle instellen, het eigenlijke vullen doet de fillRect die hier een

// rechthoek van (150,100) met hoogte en breedte 200 tekent.

context.moveTo(150, 100);

context.lineTo(350, 300);

context.moveTo(150, 300);

context.lineTo(350, 100);

// Hier worden enkel ‘pen’ bewegingen gemaakt; moveTo() gaat met de

// pen omhoog, lineTo() met de pen omlaag. Er wordt echter nog niets

// getekend!

context.strokeStyle = "#F42924";

context.stroke();

// Definieer de kleur (rood) en teken de hierboven gedefinieerde lijnen

context.beginPath();

context.arc(100, 100, 50, Math.PI, 2\*Math.PI \* 3/4);

context.stroke();

// Teken een cirkelboog met middelpunt (100,100), straal 50px. Begin de

// boog op pi radialen (‘kwart voor negen’) en eindig hem op 1,5 pi rad.

// (’12 uur’). Nulpunt is op ‘kwart over drie’ en beweeg met de klok mee.

var textGradient = context.createLinearGradient(200, 50, 300, 50);

textGradient.addColorStop(0, "red");

textGradient.addColorStop(1, "yellow");

context.fillStyle = textGradient;

context.font= "20px Arial";

context.fillText("My Canvas", 200, 50);

// De tekst begint op (200,50), dat is linksonder de ‘M’. Er wordt een

// rechthoek van (200,50) met breedte 300 en hoogte 50 genomen voor

// de textGradient. Let op dat fillText() alleen de tekst met de fillStyle vult

// en dus niet de hele rechthoek.

</script>

</body>

</html>

**Les 252 – Clock challenge**

In deze les wordt een analoge klok gebouwd die de actuele tijd weergeeft en elke seconde wordt bijgewerkt.

<!DOCTYPE html>

<html>

<head>

<title>HTML 5 and CSS 3</title>

<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no, minimal-ui">

</head>

<body>

<canvas id="clock" width="500" height="500">

Please upgrade your browser.

</canvas>

<script>

function setTime() {

var canvas = document.getElementById("clock");

var context = canvas.getContext("2d");

var clockRadius = canvas.width/2;

// De straal van de klok is de helft van de canvas grootte

context.beginPath();

context.fillStyle = "black";

context.arc(clockRadius, clockRadius, clockRadius, 0, 2\*Math.PI);

context.fill();

// Teken een zwarte cirkel

context.fillStyle = "white";

context.beginPath();

context.arc(clockRadius, clockRadius, 10, 0, 2\*Math.PI);

context.fill();

// Teken een kleine witte cirkel in het midden, dan ziet het er met de

// wijzers er straks bij wat beter uit

context.font = clockRadius / 10 + "px arial";

context.textAlign = "center";

context.textBaseline = "middle";

// De cijfers van de klok staan precies in het midden van de nog te

// berekenen coördinaten.

for (var i = 1; i <= 12; i++) {

context.fillText(i, clockRadius + clockRadius \* 0.9 \* Math.sin(i \* 2\*Math.PI / 12), clockRadius - (clockRadius \* 0.9 \* Math.cos(i \* 2 \* Math.PI / 12)));

// Teken cijfer i. De wiskunde klopt, alleen vind ik het handiger om bij

// 1/3 pi te beginnen – waar het cijfer 1 komt te staan – en de hoek dan

// telkens met 1/6 pi af te laten nemen. Dit komt overeen met hoe ik

// het vroeger geleerd heb: op (1,0) is de hoek 1, op (0,1) is de hoek

// een ½ pi, op (0,-1) is de hoek -1/2 pi etc. De berekening wordt dan

// iets anders maar het resultaat is uiteraard hetzelfde.

}

var date = new Date();

var hours = date.getHours();

var minutes = date.getMinutes();

var seconds = date.getSeconds();

var fullHours = hours % 12 + minutes / 60 + seconds / 3600;

// Het exacte aantal uren

var hoursAngle = fullHours \* 2 \* Math.PI / 12;

var minutesAngle = minutes \* 2 \* Math.PI / 60;

var secondsAngle = seconds \* 2 \* Math.PI / 60;

// Ik zou de hoeken weer iets anders berekenen met hetzelfde effect

context.strokeStyle = "white";

context.moveTo(clockRadius, clockRadius);

context.lineTo(clockRadius + clockRadius \* 0.6 \* Math.sin(hoursAngle),

clockRadius - (clockRadius \* 0.6 \* Math.cos(hoursAngle)));

context.lineWidth = 5;

context.stroke();

// Hier is de uren wijzer getekend

context.moveTo(clockRadius, clockRadius);

context.lineTo(clockRadius + clockRadius \* 0.8 \* Math.sin(minutesAngle), clockRadius - (clockRadius \* 0.8 \* Math.cos(minutesAngle)));

context.lineWidth = 3;

context.stroke();

// Hier is de minuten wijzer getekend

context.moveTo(clockRadius, clockRadius);

context.lineTo(clockRadius + clockRadius \* 0.9 \* Math.sin(secondsAngle), clockRadius - (clockRadius \* 0.9 \* Math.cos(secondsAngle)));

context.lineWidth = 2;

context.stroke();

// Hier is de seconden wijzer getekend

}

setInterval(setTime, 1000);

// setInterval voert de functie setTime() elke 1000 ms uit.

</script>

</body>

</html>