

GRADUAAT IN HET PROGRAMMEREN

SEMESTER 1

ACADEMIEJAAR 2019-2020

---

# WEB FRONTEND BASICS



---

**howest.be**



# INHOUDSOPGAVE

<b>1</b>	<b>INLEIDING</b>	<b>5</b>
<b>1.1</b>	<b>World Wide Web</b>	<b>5</b>
1.1.1	Kort stukje geschiedenis	5
1.1.2	Enkele 'wist je datjes'	5
1.1.3	World Wide Web Consortium	5
1.1.4	Super strak design?	6
<b>1.2</b>	<b>Browser?</b>	<b>7</b>
1.2.1	Algemeen	7
1.2.2	Gekende browsers	8
1.2.3	Browser werking	9
<b>1.3</b>	<b>Terminologie</b>	<b>11</b>
1.3.1	Hypertext Markup Language - HTML	11
1.3.2	Van HTML naar HTML5	11
1.3.3	Cascading Style Sheets - CSS	12
<b>1.4</b>	<b>Elementen, Attributen &amp; Waarden</b>	<b>13</b>
1.4.1	HTML Elementen	13
1.4.2	Attributen en waarden	14
<b>1.5</b>	<b>Nesten van elementen</b>	<b>15</b>
<b>1.6</b>	<b>Editoren</b>	<b>16</b>
1.6.1	VS Code	16
<b>1.7</b>	<b>HTML Validatie</b>	<b>19</b>



# 1 INLEIDING

## 1.1 WORLD WIDE WEB

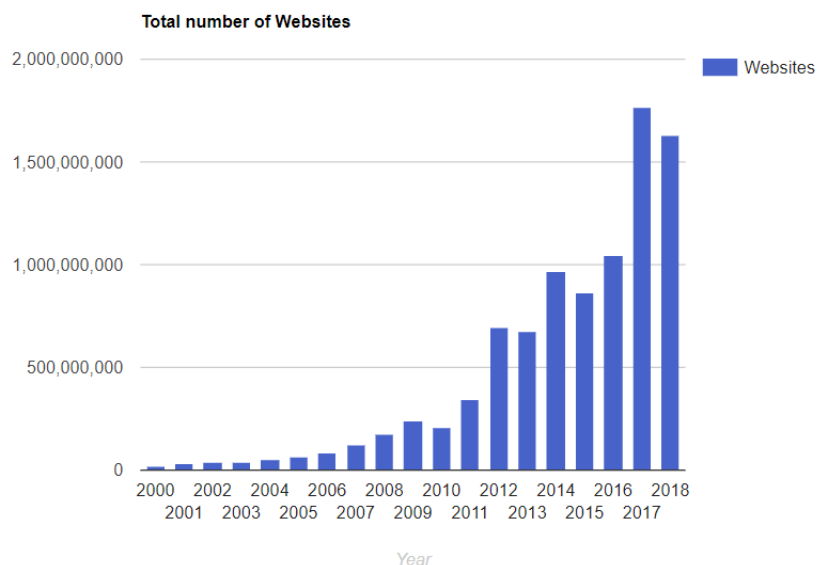
Het internet mag niet verward worden met de term “het wereld wijde web”. Het internet, een immense verzameling van aan elkaar gekoppelde computers, is slechts de basis voor vele diensten waaronder Usenet met zijn nieuwsgroepen, FTP voor bestandsoverdrachten, E-mail, Voice telefoon tussen computers, enzovoort. Het WWW is slechts een speler die gedragen wordt dankzij het bestaan van het internet.

### 1.1.1 KORT STUKJE GESCHIEDENIS

Sinds de lancering van het world wide web, hiervoor gaan we ongeveer terug naar 1991, is het aantal beschikbare websites enkel maar exponentieel toegenomen.

De statistieken (figuur 1) brengen deze evolutie mooi in kaart van 2000 tot en met 2015. Vandaag de dag kan je via [www.internetlivestats.com](http://www.internetlivestats.com) de teller volgen van het aantal gevonden unieke websites.

Deze heeft inmiddels de kaap van 1.700.000.000 overschrijden. Kleine kanttekening hierbij is dat ongeveer 75% van de websites geen actieve sites zijn maar geparkeerde domeinen of gelijkaardige.



1 bron: [internetlivestats.com](http://internetlivestats.com) ©

### 1.1.2 ENKELE ‘WIST JE DATJES’

- De eerste website (info.cern.ch) werd gepubliceerd op 6 augustus 1991 door de Britse natuurkundige Tim Berners-Lee, werkzaam in CERN, Zwitserland.
- CERN maakte op 30 april 1993 de World Wide Web technologie gratis beschikbaar voor het grote publiek, waardoor het internet kon bloeien.
- In 2013 groeide het internet met meer dan een derde op een jaar tijd. Van ongeveer 630 miljoen websites aan het begin van het jaar tot meer dan 850 miljoen in december 2013.
- Meer dan 50% van de websites van vandaag worden gehost op Apache of nginx, beide open source web servers. Sinds juni 2014 is Microsoft qua marktaandeel heel dicht bij Apache gekomen (*slechts een verschil van 0,15% scheidt de twee*). Als deze trend doorzet, zou Microsoft binnenkort voor het eerst in de geschiedenis de toonaangevende webserverontwikkelaar kunnen worden.

### 1.1.3 WORLD WIDE WEB CONSORTIUM

In 1994 werd in samenwerking met het CERN het World Wide Web Consortium het ([W3C](http://www.w3.org)) opgericht door Tim Berners-Lee. Deze organisatie heeft het doel om de ontwikkeling van webpagina's eenduidig te maken voor alle browsers door een gemeenschappelijke standaard vast te leggen. Deze cursus houdt zich voornamelijk aan de regels van het W3C omdat ze streven naar een model waar een webpagina in alle browsers op dezelfde manier wordt weergegeven.

### 1.1.4 SUPER STRAK DESIGN?

Sinds de lancering kende het internet niet alleen een grote groei maar evolueerde gaandeweg ook de factor 'design' mee met de tijd. Onderstaand een paar voorbeelden van hoe een aantal bekende websites er uitzagen toen ze voor het eerst gelanceerd werden.



Google ging live in 1998, Facebook in 2004. Zoals je zelf kan zien is er ondertussen heel wat veranderd aan het uitzicht van beide websites.



Als je zelf even wil rondneuzen hoe jouw favoriete website er vroeger uitzag, of je wil nog even voor nostalgische redenen terug in de tijd duiken?

Dan kan je dit altijd even doen via [web.archive.org](http://web.archive.org)

## 1.2 BROWSER?

---

### 1.2.1 ALGEMEEN

Om al deze websites en hun webpagina's te bekijken maken we gebruik van een browser. Nu wat doet zo een browser precies? Een browser zet webpagina's, die door een webserver zijn aangeleverd, om in een vorm die voor mensen leesbaar is.

Een aantal elementen die je vaak terugvindt op een webpagina zijn bijvoorbeeld:

- Verschillende soorten opmaak van tekst
- Afbeeldingen
- Links, of doorverwijzingen, naar andere webpagina's
- Animaties
- Video's
- ...

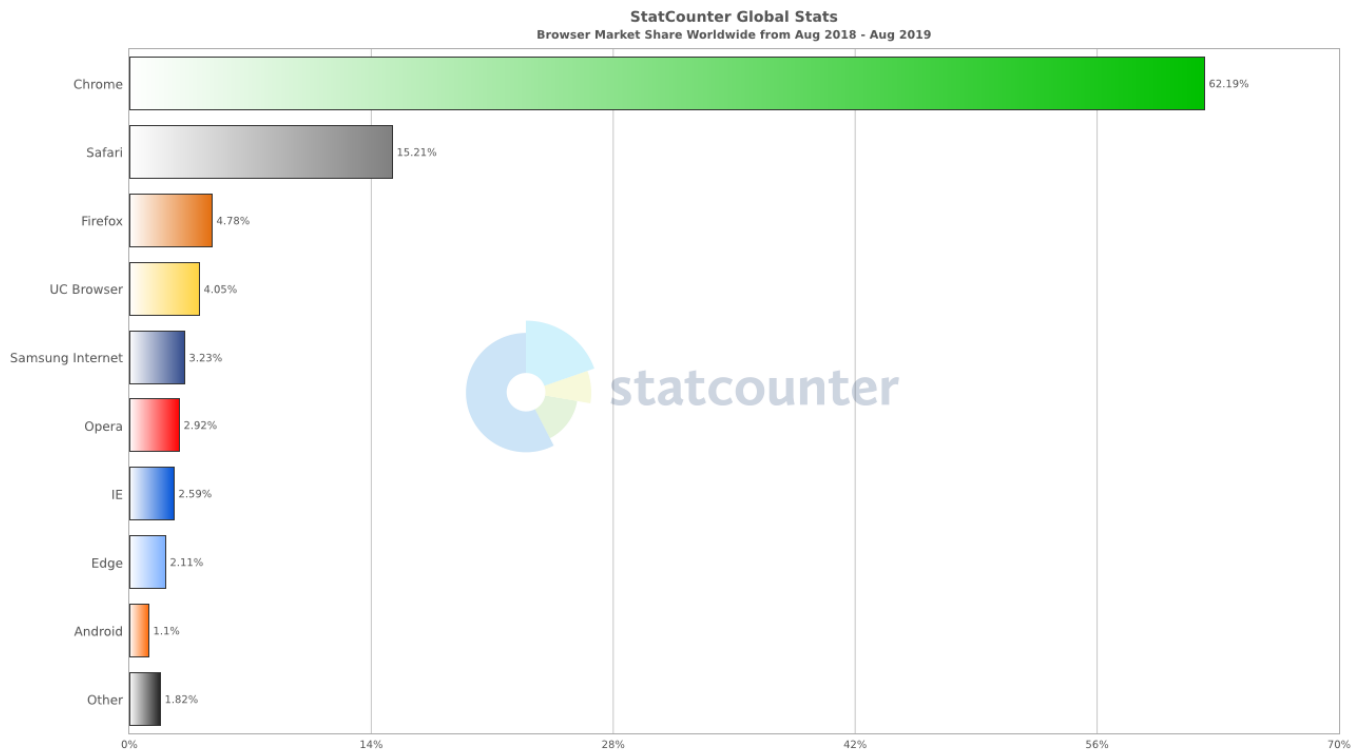
Er zijn webbrowsers die documenten voorlezen, andere dan weer zetten ze om in puntjes op een braillemachine. Maar het grotendeel van de browsers geven een webpagina weer op een computerscherm.

Vrijwel alle browsers hebben de mogelijkheid om weblocaties op te slaan (bladwijzers), bestanden te downloaden en een geschiedenis bij te houden van waar de gebruiker geweest is. Sommige browsers voegen hier nog een aantal extra's aan toe zoals meerdere tabbladen, pop-upblockers, advertentiefiltering en automatisch zoeken op een zoekmachine naar eigen voorkeur.

## 1.2.2 GEKENDE BROWSERS

Iedereen werkt ermee maar staat veelal niet stil bij de keuze van een webbrowser. Vaak wordt gewoon genomen wat een vriend of kennis aanbeveelt of simpelweg wat er geïnstalleerd was op het toestel waarmee je aan de slag gaat.

Hieronder een overzicht van de top 6 gebruikte browsers wereldwijd (*augustus 2018 – augustus 2019*).



Wil je hier meer over weten, lezen, opzoeken?

We verwijzen je dan graag door naar de volgende website: [gs.statcounter.com](https://gs.statcounter.com)



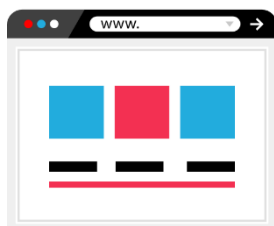
### 1.2.3 BROWSER WERKING

Om een correct beeld te hebben van hoe alles eigenlijk in zijn werk gaat staan we toch even stil om iets dieper in te gaan op de werking hiervan.

Geen paniek, we gaan niet te diep in detail en beperken ons tot een minimum van technische termen en technologie. Het draait erom even duidelijkheid te scheppen wat er allemaal achter de schermen gebeurt op het moment dat je aan het surfen gaat.

We beginnen bij het begin, laat ons even de volgende situatie schetsen. Je opent een browser (naar keuze) en wil even naar de facebook website gaan.

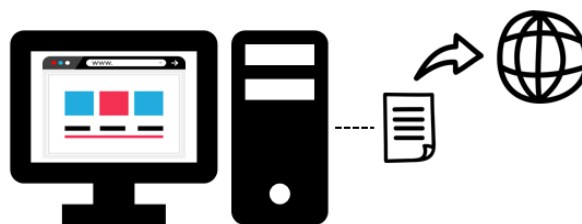
#### STAP EEN



Het invoeren van de site waar je heen wil surfen. In dit geval wordt dit facebook.com

#### STAP TWEE

Jouw computer stuurt vervolgens een 'bericht' uit die op zoek gaat naar deze informatie. De eerste stap waar jou bericht zal passeren is bij je provider. De gekendste providers bij ons zijn Telenet en Proximus. Deze ontvangt je bericht en kijkt of hij er mee aan de slag kan of niet.



#### STAP DRIE



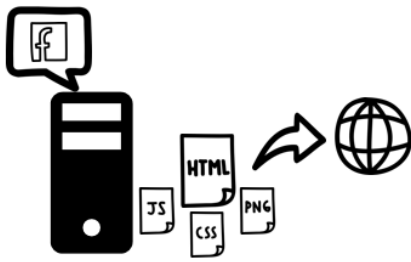
Je provider stuurt je bericht door naar de server van facebook en vraagt aan deze server als deze de gevraagde informatie beschikbaar heeft.

#### STAP VIER

De server van facebook gaat aan de slag om de gevraagde informatie op te halen. Dit kan je (weliswaar heel simplistisch) vergelijken met de mappen en folders op je eigen computer waar je iets in gaat zoeken.



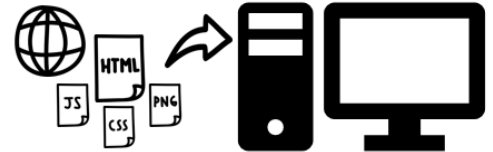
## STAP VIJF



De server van facebook stuurt de gevonden informatie terug naar je provider. Deze informatie bevat alles wat gerelateerd is aan je aanvraag.

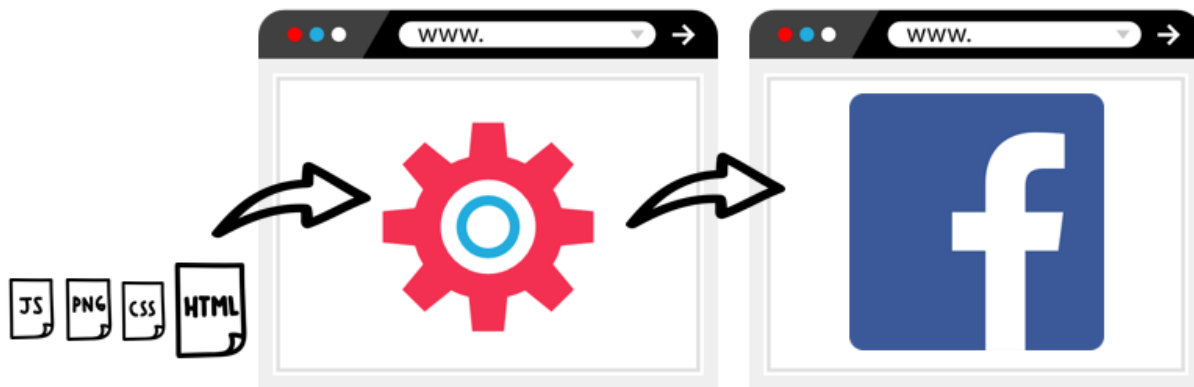
## STAP ZES

Je had het zeker wel al zien aankomen. Je provider stuurt vervolgens dit pakketje door richting jouw computer.



## STAP ZEVEN

Je browser gaat onmiddellijk aan de slag met deze informatie en tovert de ontvangen informatie om naar voor jou leesbare en verstaanbare visuele informatie.



Ziezo, we hebben de volledige schakel even zeer simpel in kaart gebracht. Onthoud echter dat dit alles gepaard gaat met de nodige complexiteit en dat er nog wel iets meer bij kijken komt achter de schermen. Maar voor deze cursus volstaat het om inzicht hebben in deze simpelere stappen.



Wil je hier meer over weten, lezen, opzoeken?

Check dan zeker even [HTML5 Rocks](#) voor wat meer informatie (*in detail*)

## 1.3 TERMINOLOGIE

### 1.3.1 HYPERTEXT MARKUP LANGUAGE - HTML

Dit is de taal en de structuur die we hanteren in het ontwikkelen van huidige webpagina's. Hoewel dit in een aantal (*verbeterde*) varianten voorkomt is HTML nog steeds de basis voor het ontwikkelen van alle webpagina's.

Vandaag kan bijna elke computer een HTML pagina weergeven, maar elke computer zal deze pagina op een eigen manier weergeven. Het is de taak van de browsers (*zoals Chrome, Firefox, Opera, Safari, ...*) om de html correct te interpreteren en af te beelden, maar er zijn nog andere factoren: de snelheid van de computer, de internetverbinding, de instellingen van het beeldscherm (resoluties, etc) en zoveel meer waar de ontwikkelaar mee rekening dient te houden. De webdesigner staat altijd voor de uitdaging om goede en "**browser safe**" pagina's te kunnen maken, zodat die overal op dezelfde manier worden weergegeven, ongeacht de browser, computer, en andere factoren die de ervaring kunnen beïnvloeden.

```
<p>De letter p staat voor "paragraph" wat "alinea" betekent in het engels, niet paragraaf!
Verwarrend niet? Een alinea is een verzameling zinnen terwijl een paragraaf meerdere alinea's
kan bevatten.</p>
<p>Wanneer we echter paragraaf vertalen naar het engels dan krijgen we "section".</p>
```

HTML Code voor de opbouw van twee alinea's.

HTML markeert tekst door middel van zogenaamde **tags**. Dit zijn de code objecten die tussen de bekjes **<** en **>** staan en vormen de basis voor de paginastructuur van een document omdat ze de weergave van de data bepalen. Een tag kan best omschreven worden als een etiket dat we rond een stuk tekst kleven om er een eigen betekenis aan te geven.

In zijn beginfase kon HTML ook de stijl van een pagina beïnvloeden: lettertypes, uitlijning, kleuren, enz. In HTML versie 4 heeft het W3C dergelijke stijlelementen afgekeurd (*deprecated*) en uit de officiële standaard gehaald en raadt ze het gebruik ervan af hoewel de meeste browsers ze nog steeds ondersteunen om alle sites te kunnen tonen. Deze tags worden hier niet meer behandeld.

Om de leegte van de afgekeurde stijlelementen te vullen creëerde men een nieuw systeem om stijlen te kunnen instellen voor een HTML pagina. Deze instructieset noemt men Cascading Style Sheets of kortweg CSS en vervangt de meeste vroegere HTML opmaak tags, maar kan nog veel meer. Meer hierover verder in deze cursus.

### 1.3.2 VAN HTML NAAR HTML5

Sinds 2012 ondersteunen de meeste gebruikte browsers een nieuwe HTML specificatie, HTML5 genaamd. Deze versie introduceerde een hele reeks nieuwe elementen. Gelukkig is HTML5 geen letterlijke opvolger van HTML4, en kunnen we nog steeds **strikt volgens de HTML regels** coderen, en onze code zuiver houden.



Afgekeurde tags en attributen die niet langer deel uitmaken van HTML vind je hier:

<http://htmlhelp.com/reference/html40/deprecated.html>



HTML5 is een **aanbevolen specificatie** bij W3C sinds oktober 2014. Het bevat heel wat elementen om een moderne webpagina's van inhoud te voorzien.

### 1.3.3 CASCADING STYLE SHEETS - CSS

Omdat sommige alinea's en andere tags meestal een bepaalde stijl, kleur en "look" moeten hebben op een webpagina worden deze voorzien van zogenaamde stijl-definities. Omdat HTML voornamelijk de opbouw van een pagina voorlegt - de blokken, paragrafen en andere zaken – was er nood aan een andere meer uniforme taal die deze blokken hun eigen stijl en karakter geeft. We moeten niet enkel de blokken bepalen door middel van **<p>-tags**, maar we kunnen deze ook een eigen stijl geven. Dit gebeurt dankzij de "stijl" taal genaamd CSS. We gaan gedurende deze cursus zowel HTML en CSS combineren om tot uniforme en presentabele webpagina's te komen. De benaming "cascading" komt nog duidelijk aan bod.

De volgende CSS-code zal bijvoorbeeld alle alinea's <p> steeds in het vet afdrukken.

```
p {  
  font-weight: bold;  
}
```

*CSS Syntax voor de opmaak van alle alinea's.*

Merk op dat de CSS taal sterk lijkt op een C of Java programmeertaal door het gebruik van accolades **{ }** en puntkomma's, maar toch gaat het hier slechts om instructies voor de te gebruiken opmaak.

In dit geval heeft de eigenschap/property "font-weight" de instructie dat de tekst in elke "p" tag afgebeeld moet worden in het vet.



Cascading Stylesheets: [https://nl.wikipedia.org/wiki/Cascading\\_Style\\_Sheets](https://nl.wikipedia.org/wiki/Cascading_Style_Sheets)



De meest recente implementatie van CSS is "CSS level 3", dat een uitbreiding is bovenop CSS level 2.1.

## 1.4 ELEMENTEN, ATTRIBUTEN & WAARDEN

HTML heeft drie instructietypes waarmee we kunnen werken: elementen, attributen en waarden (*Elements, Attributes, Values*).

### 1.4.1 HTML ELEMENTEN

Elementen structureren de verschillende delen van een webpagina. Ze markeren bepaalde delen in de pagina zodat de inhoud een bepaalde functie krijgt. In het vorige hoofdstuk hebben we reeds kennisgemaakt met `<p>` elementen die aanduiden dat elk stuk tekst die ze omsluiten een alinea betreft. De elementen kunnen **tekst of andere elementen** bevatten, maar ze kunnen ook **leeg** zijn.

```
<h1>Deze tekst wordt beïnvloed en is nu een hoofding</h1>
```

*Een typisch HTML element dat tekst bevat*

Wanneer een element in HTML iets bevat, zoals tekst of andere elementen, dan moet het element **steeds afgesloten** worden met zijn sluitingstag. Het bovenstaande voorbeeld zorgt ervoor dat de ingesloten tekst als **hoofding (h1)** zal weergegeven worden. De openingstag is een `<h1>` en wordt ook opnieuw afgesloten met de sluitingstag `</h1>`.

Er bestaan ook elementen die géén inhoud kunnen bevatten. Deze elementen hebben dan ook **geen sluitingstag**, maar worden op een **alternatieve manier afgesloten**. Dergelijke elementen bestaan uit een zogenaamde **zelfsluitende (self-closing) tag** die zowel de openingstag als sluitingstag is. Een mooi voorbeeld is het `img` element, dat een afbeelding op de pagina plaatst.

```

```

*Voorbeeld van een zelfsluitend element*

Zoals je ziet is er niet echt een openingstag of een sluitingstag. **De tag sluit zichzelf** door middel van **een spatie, een slash en een bekje**.

Tot slot moet er nog vermeld worden dat in HTML **alle tags** in kleine letters geschreven moeten worden. Dit slaat uiteraard niet op de tekstuele inhoud ervan.



Een overzicht van alle mogelijke HTML elementen kan je terugvinden via onderstaande site:  
<https://developer.mozilla.org/en-US/docs/Web/HTML/Element>

## 1.4.2 ATTRIBUTEN EN WAARDEN

Attributen maken het mogelijk om **informatie over de data** van het element te beschrijven, in tegenstelling tot de **data zelf** die we terugvinden tussen de openings- en sluitingstags. Wanneer een attribuut geplaatst wordt in een tag dan heeft deze **altijd** een waarde in HTML. Deze waarde wordt steeds ingesloten tussen dubbele aanhalingstekens, zoals het voorbeeld hieronder toont.

```
<a href="http://www.google.be">Ga naar Google</a>
```

*Een attribuut van het a-element. De aanhalingstekens zijn verplicht en omsluiten de **waarde** van het **attribuut href**.*

Een a-element creëert een hyperlink op de pagina. In dit geval zal de hyperlink de tekst "Ga naar Google" als data gebruiken en weergeven. We hebben echter ook extra informatie nodig bij deze data, namelijk het adres van de website waar we naartoe willen navigeren bij het klikken op de link. Dit gebeurt door het attribuut **href**. Let goed op de plaatsing van de aanhalingstekens en het gelijkheidsteken.

Hetzelfde geldt voor attributen van een zelfsluitende tag:

```

```

*Bij deze img-tag zijn er twee attributen gedefinieerd: waar de figuur gevonden kan worden (src) en de alternatieve tekst. Vergeet de spatie voor de eind-slash niet.*

Er bestaan twee soorten attributen: **specifieke** en **globale** attributen. Globale attributen (ook wel standaardattributen genoemd) zijn van toepassing op **alle** HTML-elementen. Specifieke attributen zijn steeds afhankelijk van het element.



Attributen kunnen niet zo maar willekeurige waarden bevatten. Voor een goede referentie neem je dus best even een kijkje online.



Een overzicht van alle global attributes kan je terugvinden via onderstaande site:  
[https://developer.mozilla.org/en-US/docs/Web/HTML/Global\\_attributes](https://developer.mozilla.org/en-US/docs/Web/HTML/Global_attributes)

## 1.5 NESTEN VAN ELEMENTEN

Alle elementen die niet zelfsluitend zijn, m.a.w. elementen die een openingstag en sluitingstag hebben, kunnen andere elementen bevatten. Elementen die zelfsluitend zijn kunnen uiteraard geen tekst of andere elementen meer bevatten. De elementen die ingesloten zijn door een ander element noemen we **Child** elementen, en het omsluitende element heet het **Parent** element. We kunnen dus gerust spreken van een hiërarchie in de structuur die sterk lijkt op een stamboom, de “document tree”.

In het onderstaande voorbeeld is het `div` element de **parent** van de `img` en de twee `p` elementen.

De `p` elementen zijn **children** van de `div`. De eerste `p` is **parent** van het `cite` element, wat op zijn beurt een **child** element is van dit `p` element.

```
<div>
  <p>
    De haas en de schildpad is een fabel door <cite>Aesopus</cite>
  </p>
  <p>Hij was een griekse verteller uit de oudheid.</p>
  
</div>
```

Deze relaties tussen elementen zijn belangrijk bij het stijlen van elementen. We gaan het hier nog uitgebreid over hebben wanneer we CSS onder de loep nemen.

Wanneer we nu een element **in** een element willen plaatsen dan is het belangrijk om deze hiërarchie te respecteren.

```
<p>De haas en de schildpad is een fabel door <cite>Aesopus <mark>(Griek)</mark></cite></p>
```

*Geneste elementen. Juist afgesloten.*

Het `p`-element bevat nu wat tekst, aangevuld met een verwijzing naar de auteur via het `cite`-element en wat verderop ook een `mark`-element. We zeggen dat het `mark`-element **genest** is in het `cite`-element.

Aangezien we nog steeds **in** het `cite`-element zaten toen we de openingstag `<mark>` schreven, moet ook de eindtag `</mark>` zich in dit `cite`-element bevinden. De volgende code zou dus volledig **FOUT** zijn:

```
<p>De haas en de schildpad is een fabel door <cite>Aesopus</p></cite>
```

*Nesten van elementen. **FOUT** afgesloten.*

Zodra je een sluitingstag plaatst moet deze dus overeenstemmen met de dichtst voorgaande, niet gesloten openingstag. Als we dus eerst een element X openen en daarna een element Y, dan moeten we eerst Y sluiten en daarna pas X.

## 1.6 EDITOREN

Er zijn tal van Editoren op de markt waar je mee aan de slag kan om met HTML en CSS je webpagina's tot leven te brengen. Je bent vrij om hier je eigen keuze in te maken.

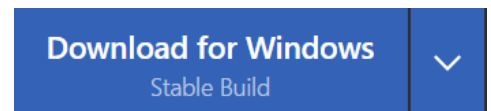
De voorbeelden en referenties in deze cursus zullen echter steeds verwijzen naar Visual Studio Code. We raden het gebruik van Visual Studio Code dan ook aan als je vlot wil meevolgen en niet te veel tijd wil verliezen om een andere editor onder de knie te krijgen.

### 1.6.1 VS CODE

In onderstaande stappen overlopen we kort de installatie van Visual Studio Code. Handige Extensions worden ook vermeld na de stappen van de installatie

#### INSTALLATIE

1. Download de laatste versie van Visual Studio Code via de **Download for Windows** knop op de website. Na enkele ogenblikken start je download automatisch.



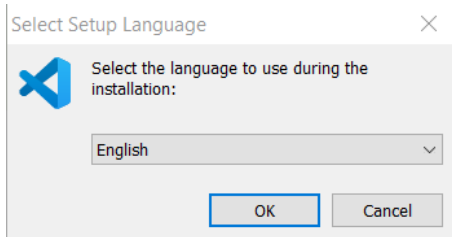
Indien je een ander platform draait, kies dan uiteraard voor de relevante versie.



Je kan de laatste versie van Visual Studio Code terugvinden via onderstaande link:

<https://code.visualstudio.com>

2. Kies voor **English** bij de installatie taal.

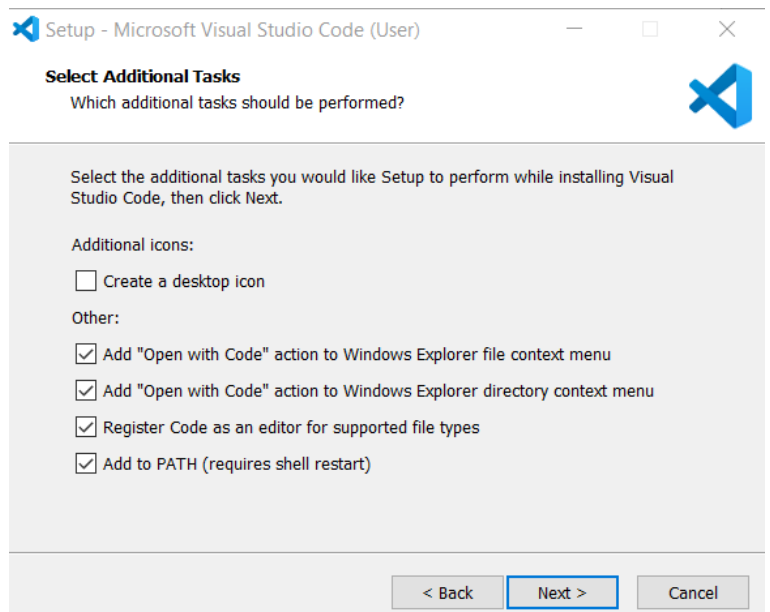


3. **Accepteer** vervolgens de **License Agreement** en klik op **Next**.

4. Zorg er vervolgens voor dat de **opties** zoals deze op de screenshot zichtbaar zijn **aangevinkt** staan. Deze zullen het je net iets makkelijker maken om met Visual Studio Code te werken.

Klik vervolgens op **Next** en **Install**.

Visual Studio Code wordt nu geïnstalleerd.






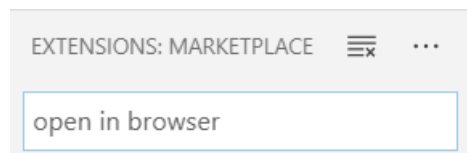
## NUTTIGE EXTENSIES

Er zijn tal van Extensies die beschikbaar zijn binnen Visual Studio Code om het leven van een developer makkelijker te maken. We kunnen ze uiteraard niet allemaal bespreken maar we pikken er wel even de twee meest interessante uit om meteen mee van start te gaan.

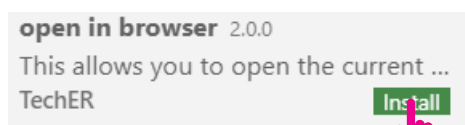
### 1. Open in browser – TechER

Deze extensie stelt je in staat om vanuit Visual Studio Code meteen je pagina door je default browser te laten openen door de toetsencombinatie **alt-b** te gebruiken. Je kan deze extensie rechtstreeks in je Visual Studio Code toevoegen door in de linker navigatiebalk op het icoon voor extensions te klikken 

Voeg vervolgens bovenaan in de zoekbalk **open in browser** als zoekterm in.



In de gefilterde lijst klik je vervolgens op de Install button die bij de extension **open in browser** van de publisher **TechER**




Meer is er niet nodig om deze extensie te installeren.



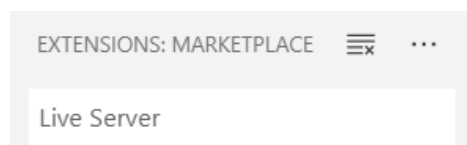
Je kan deze extensie ook terugvinden in de Visual Studio Marketplace via onderstaande link:

[Marketplace - open in browser - TechER](#)

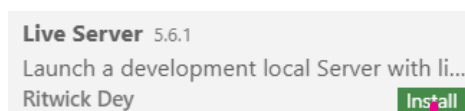
### 2. Live Server – Ritwick Dey

Deze extensie stelt je in staat om wijzigingen die je in Visual Studio Code maakt meteen door te voeren op de pagina in je browser. Je kan deze extensie rechtstreeks in je Visual Studio Code toevoegen door in de linker navigatiebalk op het icoon voor extensions te klikken 

Voer vervolgens bovenaan in de zoekbalk **Live Server** als zoekterm in.



In de gefilterde lijst klik je vervolgens op de Install button die bij de extension **Live Server** van de publisher **Ritwick Dey** staat.



Meer is er niet nodig om deze extensie te installeren.



Je kan deze extensie ook terugvinden in de Visual Studio Marketplace via onderstaande link:

[Marketplace - Live Server - Ritwick Dey](#)

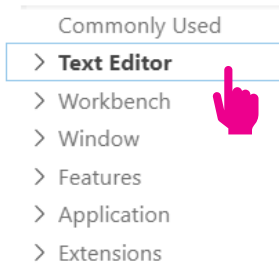
## AANBEVOLEN INSTELLINGEN

Tot slot hebben we nog een instelling die best handig is om in te schakelen in Visual Studio Code. Dat is de optie om je wijzigingen automatisch op te slaan. Op deze manier worden je laatste wijzigingen steeds meegenomen en verlies je alvast geen tijd met het zoeken waarom die laatste aanpassing niet correct weergegeven wordt in de browser.

In je Visual Studio Code ga je naar:

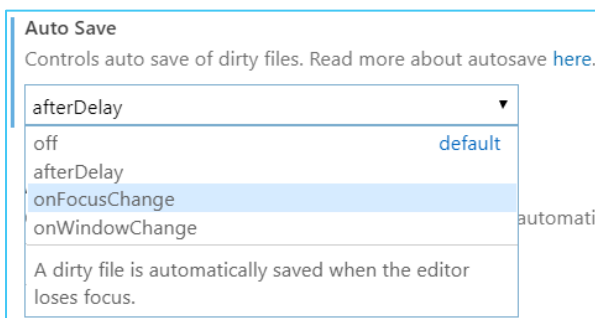
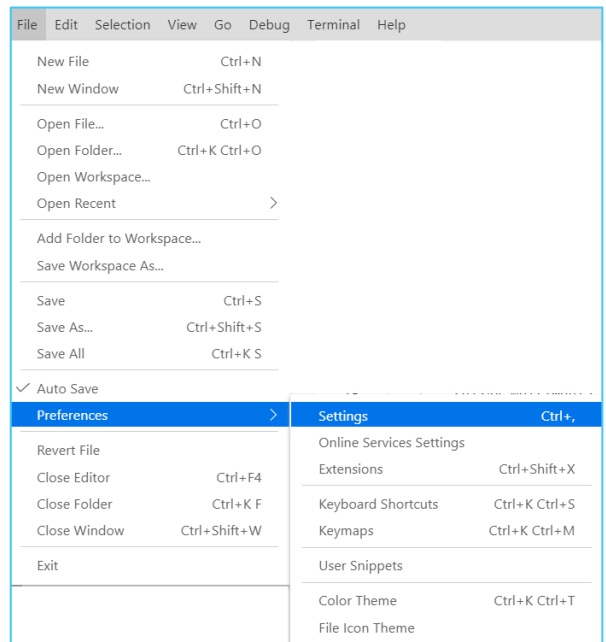
**File -> Preferences -> Settings**

Klik vervolgens de **Tekst Editor** node open



Na het openklappen selecteer je hier de menu optie **Files**

Daarin ga je opzoek naar de **Auto Save** setting en plaats deze op **onFocusChange**.



## 1.7 HTML VALIDATIE

Tijdens deze module werken we er naartoe om steeds geldige HTML code op te leveren. Om ervoor te zorgen dat je HTML code vrij van fouten is kan je hiervoor gebruik maken van een HTML Validator.



De officiële validatie is terug te vinden online via het W3C op volgende website:

[Markup Validation Service](https://validator.w3.org/)

Via deze website kan je HTML laten valideren op basis van een URL, een bestand dat je upload, of het copy pasten van je broncode. Deze verschillende opties zijn terug te vinden in de verschillende tabbladen op de website zelf.

Je bent vrij om gebruik te maken van andere validators die je online tegenkomt, of eventueel als plugin kan installeren in je browser zelf. Maar zorg ervoor dat je zelf dan ook even controleert als de door jou gekozen tool ook dezelfde errors toont als deze van de W3C. Daar deze manier van controleren voor opdrachten/examen gehanteerd zal worden door de lectoren.