hogeschool vives

# Web Development 1 : Labo 4 HTML - 3

studiegebied **HWB**

bachelor in het **toegepaste Informatica**

campus **Kortrijk**

academiejaar **2024-2025**

## Inhoudsopgave

# 1 Inleiding

Deze les behandelt enkele praktische zaken rond werken met afbeeldingen, HTML validering en online hosting.

## 1.1 Verslag

In dit deel van de cursus staan verschillende vragen die je moet beantwoorden en opdrachten om iets te maken of uit te proberen. Het is belangrijk dat je alle opdrachten zorgvuldig uitvoert!

Documenteer je werk in een verslagdocument '**verslag html deel 3.pdf**' waarin je:

- Voor elke uitprobeer opdracht een entry maakt met screenshots ter staving van wat je deed.
- Je antwoorden op de gestelde vragen neerschrijft.

Oplossingen van 'grotere' opdrachten (met veel code) bewaar je in een Webstorm project "**html deel 03"**. Per opdracht maak je in dit project een aparte folder waarin je de bestanden (en subfolders) plaatst.

# 2 Labo

## 2.1 Werken met afbeeldingen

We zagen reeds eerder dat een `<img>` een width en een height attribuut kunnen hebben om de afbeelding op de gewenste grootte te tonen. We verkiezen deze afmetingen in te stellen met CSS, het is immers een presentatie kwestie. Enkel in sommige randgevallen kan het interessant zijn om het toch via de HTML attributen te regelen.

Indien je met je camera een foto maakt is deze doorgaans zeer groot en neemt teveel ruimte in op een webpagina. Om ze kleiner af te beelden kun je de afmetingen op de pagina beperken (met CSS of width en height attributen) maar dan is het resultaat niet zo performant. De server stuurt immers de afbeelding in origineel formaat naar de browser, die dan het grootste deel van de informatie wegsmijt om ze kleiner te kunnen afbeelden. Dit verspilt bandbreedte en laadt de pagina trager in (denk bv. aan mobile bezoekers met een beperkte verbinding).

Een betere oplossing is de afbeelding op voorhand op de juiste grootte te brengen en het kleinere bestand online te zetten. Er rijst dan echter een probleem als we eenzelfde pagina voor mobile en desktop gebruikers willen maken :

- Een klein prentje dat ideaal is op een smartphone is voor een desktop gebruiker te klein (of, als de browser het vergroot, gepixelleerd)
- Een groot prentje dat ideaal is voor een desktop gebruiker verspilt bandbreedte voor een smartphone gebruiker.

Mogelijke oplossingen om voor elk device een gepaste afbeelding te bieden:

- CSS media queries (als het background images zijn)
- Dynamisch inladen met javascript naargelang de gedetecteerde schermgrootte
- Het nieuwe `<picture>` element
  - Zie bijvoorbeeld https://developer.mozilla.org/enUS/docs/Web/HTML/Element/picture#Examples.
    - Als je het browser venster kleiner maakt zul je zien dat de browser naar een andere versie overschakelt. Je kunt ook de bijkomende HTTP requests meevolgen op het Netwerk tabblad van de Developer Tools in je browser.

## 2.2 Afbeeldingen en bestandsformaten

Voor afbeeldingen kunnen er verschillende bestandsformaten gebruikt worden. In de context van webpagina's gebruikt men doorgaans jpeg, png en gif.

- Sommige formaten hebben een betere kwaliteit/bytes verhouding dan andere.

- Sommige formaten werken met lossy compression, andere met lossless compression of allebei.

## 2.3 Opdracht 1

Zoek op wat lossy en lossless compression is en geef een beschrijving van beiden.

Welke soort compression gebruiken de jpeg, png en gif formaten?

Lossless compression is wanner een bestand word gecompressed maar geen kwaliteitsverlies heeft

Lossy compression is wanneer een bestand word gecompressed en kwaliteitsverlies heeft.

Png en gif gebruiken lossless compression.

Jpeg gebruikt lossy compression

**Tip** Als je kiest voor een formaat met lossy compression, bewaar het dan pas op het allerlaatst in dit formaat en hou steeds een versie in een lossless formaat beschikbaar voor toekomstige bewerkingen.

Wat zou er gebeuren als je een afbeelding herhaaldelijk bewerkt en bewaart in een lossy formaat?

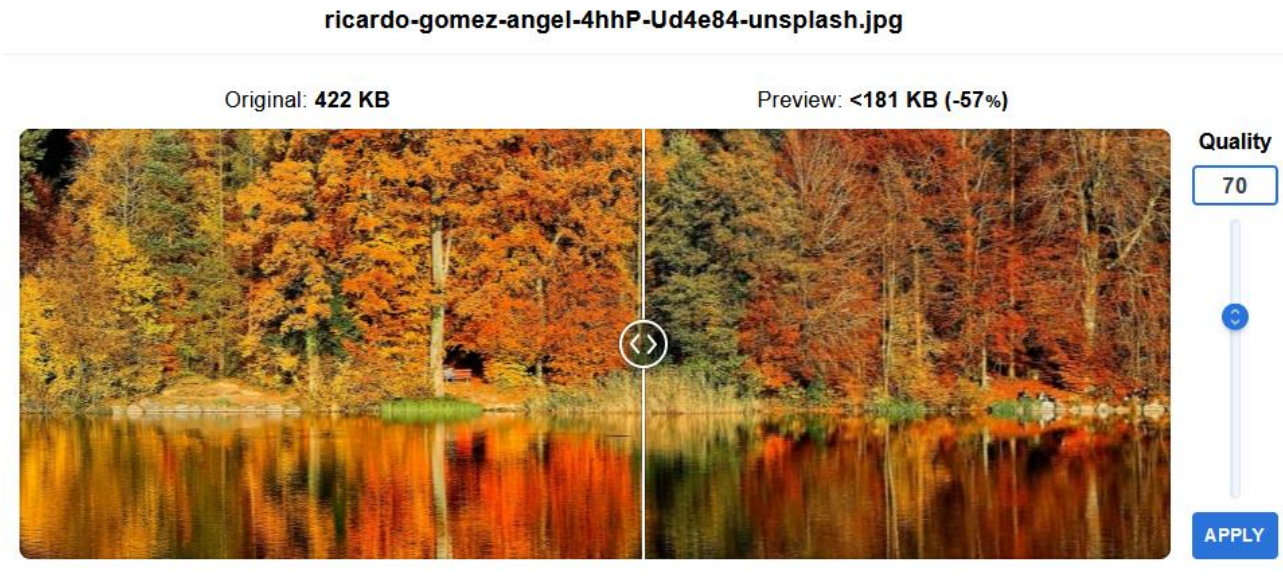
De kwaliteit zou blijven dalen.

## 2.4 Opdracht 2

Zoek een JPEG afbeelding op google images met medium grootte, die veel details bevat (bv. blaadjes aan een boom) en niet al te uniform is van kleur.

Ga naar https://imagecompressor.com/ en upload je afbeelding. De webpagina toont je nu het origineel en een preview versie met “Quality Factor” 90. Met de slider aan de rechterkant kun je de quality setting aanpassen. Zoom bij de preview afbeelding in op een detail en vergelijk quality settings 90, 70 en 40.


ricardo-gomez-angel-4hhP-Ud4e84-unsplash.jpg
Original: 422 KB
Preview: <422 KB (0%)
Quality
90
APPLY


ricardo-gomez-angel-4hhP-Ud4e84-unsplash.jpg
Original: 422 KB
Preview: <181 KB (-57%)
Quality
70
APPLY


ricardo-gomez-angel-4hhP-Ud4e84-unsplash.jpg
Original: 422 KB
Preview: <97 KB (-77%)
Quality
40
APPLY

Vergelijk hun kwaliteit en bestandsgrootte en noteer je bevindingen in je verslag. Je kunt door rechts te klikken op de preview en 'copy image' makkelijk de preview in je verslag overnemen.

Je kunt niet veel verschil zien bij de 90 en 70 je begint pas wat verschil te zien bij 40. Het verschil in grootte is veel. Bij kwaliteit 70 wordt de grootte meer dan gehalveerd. En van 70 naar 40 wordt het nog eens bijna gehalveerd.

**Uitbreiding**: maak een screenshot van Webstorm waarop wat tekst te zien is, je kunt dit makkelijk doen met de Snipping Tool in Windows. Bewaar dit in JPEG formaat en gebruik de eerdere webpagina om de “Quality Factor” te veranderen. Probeer bv. eens 70 en 40. Je kunt in de preview inzoomen en door de afbeelding scrollen. Kijk aandachtig naar de tekst in de afbeelding, wat valt je op? De tekst wordt moeilijker leesbaar.

## 2.5 HTML validering

Lees hoofdstuk 4 uit de HTML cursus.

HTML code moet aan bepaalde syntactische/structurele voorwaarden voldoen om geldig te zijn, denk bv. aan de nesting regels.

Programma's die HTML code moeten ontleden (bv. browsers en tools die de inhoud indexeren zoals de google bot) rekenen er in zekere mate op dat een pagina daadwerkelijk correcte HTML bevat.

De betere tools zijn relatief robuust zodat ze niet crashen als ze met slechte HTML-code geconfronteerd worden, maar dit is zeker niet altijd het geval. Browsers zijn daar best wel goed in en tonen ook foute HTML-code op een aanvaardbare manier aan de gebruiker. Elke browser doet dit echter op een net iets andere manier, dus wat de gebruiker te zien krijgt is soms onvoorspelbaar bij incorrecte HTML-code.

Vandaar dat het belangrijk is om steeds geldige HTML-code te schrijven. Merk op dat dit **beslist** geldt voor alles wat je indient in de rest vd cursus.

Webstorm checkt je HTML-code en toont fouten en waarschuwingen a.d.h.v. gekleurde streepjes in de marge. De controles in Webstorm zijn echter oppervlakkig. Als je zeker wil zijn dat je HTML correct is moet je een strengere tool gebruiken. Hiervoor bestaan er losstaande programma's, browser extensies en online diensten.

## 2.6 Opdracht 3

Valideer je oplossingen van onderstaande opdrachten (via file upload) bij de W3C Markup Validation

Service op https://validator.w3.org/

- Opdracht “Personal homepage” (deze met semantische elementen)
- Opdracht “Contact information”

- Opdracht “Opleidingsaanbod”

Is je HTML code foutloos? Bekijk beslist ook eens de warnings die je krijgt en probeer ze weg te krijgen door je rechtzettingen opnieuw te valideren!

Voor personal homepage werdt er aangeraden om een heading bij de tabel te plaaten. En werd er gezegd dat border attribute obsolete is en dat ik css moest gebruiken.

Bij Contact information waren er foutmeldingen voor de het border attribuut en voor de <br /> attribuut.

Bij opleidingsaanbod ging het ook over de <br /> attribuut.

## 2.7 Online hosting

Om je pagina's beschikbaar te maken vanop het internet, moeten deze op een bereikbare webserver worden geplaatst.

Je kan dit op een (al dan niet virtuele) server doen die enkel voor jouw site bestemd is (*dedicated hosting*) maar dan moet je doorgaans ook het beheer van die server voor je rekening nemen. Een eenvoudige manier om snel wat pagina's online te krijgen, is gebruik te maken van een sha*red hosting* oplossing waarbij je site samen met duizenden andere sites op eenzelfde webserver wordt geplaatst.

Indien je enige mate van garantie wil over *up-time*, performantie en dienstverlening moet je typisch betalen.

Er zijn ook gratis webhosts (typisch shared hosting) die je kunt proberen, al zullen die vaak automatisch reclame toevoegen op je pagina’s. Je kunt ook gebruik maken van GitHub pages.

Je bestanden op de server krijgen kun je doen via een web interface (bv. via een admin webpagina uploaden) of een FTP programma, zoals FileZilla (https://filezilla-project.org).

## 2.8 Opdracht 4

Plaats je oplossing voor je persoonlijke webpagina op GitHub pages.

Lees eerst het document “Github Pages introductie.pdf”

**Let op**: je homepage wordt publiek gemaakt op het internet, je zou best je persoonlijke informatie vervangen door fictieve gegevens!

## 2.9 Mogelijke paden in een URL

Bij verschillende HTML-elementen kun je een URL opgeven naar een andere resource, denk bv. aan het `href` attribuut van een hyperlink of het `src` attribuut van een `img` element.

Lees de volgende uitleg over hyperlinks:

http://www.mediacollege.com/internet/html/hyperlinks.html

Let vooral op de secties over:

- Absolute path
- Document-relative path
- Site-relative path

Wat ook vaak gebruikt wordt is het gebruik van twee puntjes ".." om een niveau hoger te gaan. Bijvoorbeeld: `../images/test.jpg` zal één directory naar boven gaan om dan af te dalen in de directory "images" en daarin "test.jpg" te verwijzen.

Er zijn geen vaste regels, maar gebruik typisch:

- *Absolute paden* als je linkt naar documenten van een andere site (je kan trouwens niet anders in dit geval).
- *Document-relatieve paden* als je linkt naar images en andere resources die bij de pagina zelf horen.
    - Indien je website niet te complex is kun je ook op deze manier linken naar andere pagina's binnen dezelfde site ○ Wordt snel complex door de vele "`../../enz`" paden.
    - Zo wordt de pagina ook afhankelijk van haar eigen locatie t.o.v. andere pagina's op de site
- *Site-relatieve paden* zijn handig om naar andere pagina's binnen eenzelfde complexe site te linken.

## 2.10 Opdracht 5

Splits je persoonlijke pagina op in 3 documenten:

- Een hoofdpagina `index.html`, over jezelf (al dan niet fictief).
- Een pagina `hobbys.html`, over je hobby's.
- Een pagina `likes.html`, over je likes.

Voeg aan je hoofdpagina links toe naar de andere pagina's en zorg dat deze op hun beurt teruglinken naar je hoofdpagina.

Gebruik hiervoor eerst eens absolute paden van op je ontwikkelingsmachine (die beginnen dus met `http://localhost`). Ga na dat dit werkt op je eigen computer maar niet bij de online webhost.

Pas daarna je oplossing aan zodat document-relatieve paden gebruikt worden. Vergewis je ervan dat deze zowel op je eigen computer als op de online webhost werken

```
 <title>Toeter</title>
ead>
dy>
ader><h1>Dikke Toeter</h1></header>
ction>
 <header><h3>Persoonlijke informatie sectie</h3></header>
 <ul>
     <li>Volledige naam: Dikke Toeter</li>
     <li>Adres: blabla</li>
     <li>Telefoonnummer: +32 65 81 80 79</li>
     <li>dikke.toeter@myspace.be</li>
     <li><img src="images/picture._FMjpeg" alt="foto van mij"></li>
 </ul>
ection>
ction>
 <li><a href="http://localhost:63342/ClementDuchateau.github.io/labo4/paginaind
 <li><a href="http://localhost:63342/ClementDuchateau.github.io/labo4/paginaind
l>
ection>
ody>
tml>
```

Werkt op mijn pc maar niet op webserver.

```
        <li>dikke.toeter@myspace.be</li>
        <li><img src="images/picture._FMjpeg" alt="foto van mij"></li>
    </ul>
</section>
<section>
<ul>
    <li><a href="./hobbys.html">Hobbys</a></li>
    <li><a href="./likes.html">likes</a></li>
</ul>
</section>
</body>
</html>
```

Werkt op pc en webserver