

**HO
GENT**



Web Development I

09 Layout - Responsive

**HO
GENT**

Inhoud

- Inleiding
- Media queries
- Responsive images
 - resolution switching
 - art direction
 - file formats

Layout - Responsive

Inleiding

Responsive Web Design
is about creating web pages that look
good on all devices!

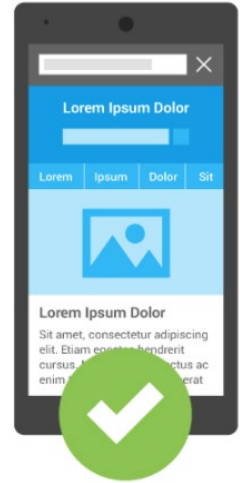


Waarom?

- betere user experience
- meer bezoekers
- hogere SE ranking
- betere laadtijden
- beheersbare ontwikkeling & onderhoud
 - geen aparte ontwikkeling voor \neq devices

Hoe?

- **HTML** en **CSS** gebruiken om pagina's geschikt te maken voor alle soorten schermen en devices
 - Lay-out van pagina kan wijzigen
 - inhoud kan verborgen/getoond worden
 - elementen kunnen anders getoond worden
 - menu's, buttons, tekst, ...



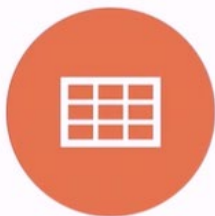
Mobile First



- start ontwikkeling met focus op mobile experience
 - minimaal, maar bevat alle belangrijke elementen
 - focus op inhoud
- geleidelijk opschalen voor grotere schermen
 - meer mogelijkheden voor extra inhoud
 - meer toeters & bellen

Technieken

Consider these guiding principles



**GRID-BASED
LAYOUT**

& flexbox reeds
behandeld



**IMAGES THAT
RESIZE**

dit hoofdstuk



MEDIA QUERIES

Layout - Responsive

De Viewport

Viewport

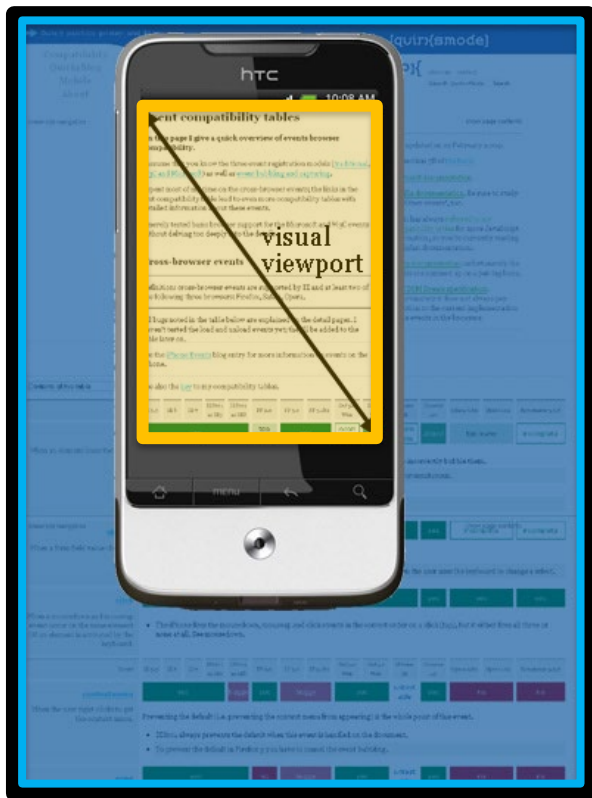
a viewport represents the part of the document you are viewing

- op grotere schermen komt dit overeen met de grootte van het browser venster
- op mobile devices komt de viewport overeen met het volledig scherm
 - de grootte van het browser venster kan op een mobiel niet aangepast worden

Viewport

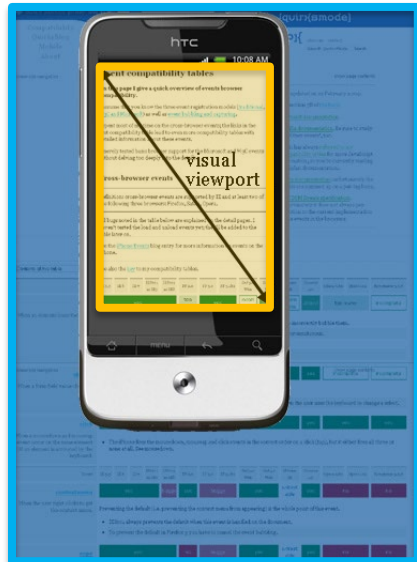
- mobiel: **visual & layout viewport**
 - visual viewport
 - bevat het gedeelte van de pagina dat momenteel op het scherm wordt getoond
 - layout viewport
 - is groter dan de visual viewport en bevat de layout van de volledige pagina
 - CSS werkt met de afmetingen van deze layout viewport

Viewport



- de **visual viewport** toont een deel van wat beschikbaar is in de **layout viewport**
- door te zoomen (in/uit), te ‘pannen’, portrait/landscape mode te gebruiken breng je een ander deel van de **layout viewport** in de **visual viewport**

Viewport



- de grootte van de layout viewport wordt bepaald door de fabrikant van de browser
- bij volledig uitzoomen komt de volledige inhoud van de layout viewport in de visual viewport



Viewport

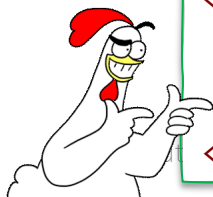
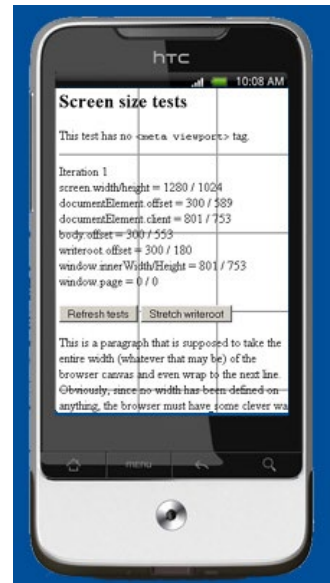
- probleem
 - pagina wordt gebouwd t.o.v. **layout viewport breedte**
 - uitgedrukt in **DIP - Device Independent Pixels**
 - bij openen van de pagina gaat de browser **uitzoomen** om je de volledige pagina te tonen in de **visual viewport**...
 - ...en dat ziet er meestal niet uit...



Viewport meta tag



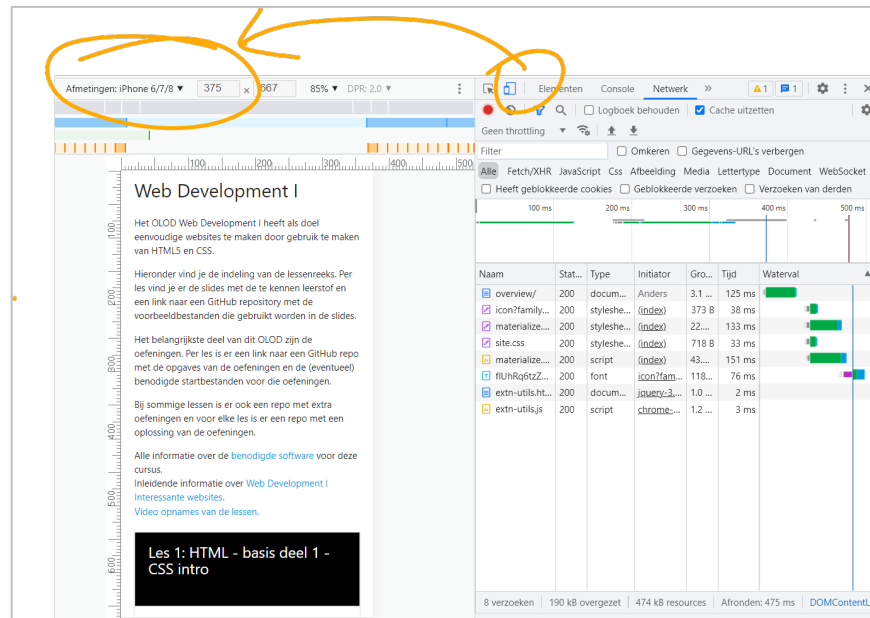
- via de **viewport meta tag** wordt het mogelijk de viewport width gelijk te stellen aan de device width
- **! altijd gebruiken bij RWD !**
- **width = device-width**
 - breedte waarop we werken is de breedte van het device in DIP
- **initial-scale = 1.0**
 - 1 DIP = 1 CSS Pixel



```
<head>
...
<meta name="viewport" content="width=device-width, initial-scale=1.0">
...
</head>
```


Voor we starten...

- maak gebruik van de **responsive web developer tools** van je browser
 - bekijk de pagina op verschillende devices via een emulator
 - bekijk pagina op eigen configuratie



Layout - Responsive

Media Queries

Media query

a **media query** is a method of testing certain aspects of the user agent or device that the document is being displayed in.

- **@media**
- aspecten: **media type & media features**
 - logische expressie die *true* of *false* is
 - afhankelijk van het resultaat van de test kan bepaalde opmaak al dan niet toegepast worden

Media query - voorbeeld

01_SimpleMediaQuery

```
p {  
    font-size: 1.5em;  
    background-color: burlywood;  
}  
  
@media screen and (min-width : 800px) {  
    p {  
        background-color: chocolate;  
        color: lightyellow;  
    }  
}
```

- media type: **screen**
- logische operator: **and**
- media feature: **width** - **min-width: 800px**
- de **CSS regels** die bij deze query horen worden enkel toegepast als de media query true oplevert

“als het venster waarin de pagina bekeken wordt een breedte heeft die groter of gelijk aan 800px is, dan krijgen p-elementen een ‘chocolate’ achtergrondkleur en ‘lightyellow’ gekleurde tekst”

Media query

- merk op: de user agent gaat media queries **automatisch her-evalueren** indien relevante eigenschappen wijzigen
 - bv. bij re-size van venster worden media queries die gebaseerd zijn op eigenschappen van het venster opnieuw geëvalueerd en kunnen de css-regels dynamisch veranderen

Media query

- merk op: de **volgorde** waarin de media queries geplaatst worden is belangrijk
 - media queries worden van boven naar onder geëvalueerd
 - denk aan regels van cascade & overerving

```
@media screen and (max-width : 800px) {  
  p {  
    background-color: chocolate;  
    color: lightyellow;  
  }  
}  
  
@media screen and (max-width : 1000px) {  
  p {  
    background-color: sandybrown;  
    color: firebrick;  
  }  
}
```

de eerste media query kan true opleveren maar de css regels die erbij horen worden nooit toegepast;

de tweede media query zal, onder de omstandigheden waarvoor de eerste media query true oplevert, immers ook true opleveren, en de bijhorende css regels overschrijven de css regels die bij de vorige media query staan...

Media queries

- twee werkwijzes
 - media queries en css opnemen in 1 bestand
 - 1 groter bestand
 - werkwijze die we zullen hanteren in dit hoofdstuk
 - apart css bestand enkel laden als de media query true oplevert
 - meerdere kleine bestanden

Media query

- media type
 - **all**
 - komt overeen met eender welk device
 - **print**
 - komt overeen met printers en devices in een modus om te printen (bijvoorbeeld een browser in print-preview modus)
 - **screen**
 - komt overeen met alles dat niet overeen komt met print

Note: It is expected that all of the media types will also be deprecated in time, as appropriate [media features](#) are defined which capture their important differences.

Media query

- **media features** zijn omvangrijk en bieden heel veel mogelijkheden

- **viewport/page afmetingen**

- bv. width, height, aspect-ratio, orientation

*wij zullen focussen
op deze media features*

- **kwaliteit**

- bv. via resolution kan je queries maken die rekening houden met de resolutie van het device

- **kleur**

- bv. via color kan je queries maken die rekening houden met de kleurdiepte van het device

- **interactie**

- bv. via pointer kan je queries maken die rekening houden met de accuraatheid van het pointing device (touchscreen vs muis/touchpad)

Media queries – enkele voorbeelden

```
@media (min-width: 500px) {  
    /* CSS toepassen wanneer viewport width >= 500px */  
}
```

gebruik **min-** prefix om een ondergrens in te stellen

```
@media (max-width: 50em) {  
    /* CSS toepassen wanneer viewport size <= 50em */  
}
```

gebruik **max** prefix om een bovengrens in te stellen

```
@media (min-aspect-ratio: 1/1) {  
    /* CSS toepassen wanneer viewport width >= viewport height */  
}
```

aspect ratio: verhouding width/height

Media queries – enkele voorbeelden

```
@media screen and (min-width: 500px) and (max-width: 800px) {  
    /* CSS toepassen wanneer viewport size >= 500px EN viewport size <= 800px is */  
}
```

gebruik logische **and** operator

```
@media screen and (min-width: 500px), screen and (max-width: 800px) {  
    /* CSS toepassen wanneer viewport size >= 500px OF viewport size <= 800px */  
}
```

gebruik , als logische **or**

```
@media not print {  
    /* CSS toepassen voor non-printing */  
}
```

gebruik **not** voor de negatie

Media queries – enkele voorbeelden

```
@media screen and (orientation:landscape) {  
  /* CSS toepassen wanneer viewport width >= viewport height*/  
}
```

orientation vergelijkt width & height

```
/* at time of writing this works only in FireFox! */  
@media screen and (height>500px) {  
  /* CSS toepassen wanneer viewport height >= 500px*/  
}
```

je mag nog grote veranderingen
verwachten voor media queries...

[Media Queries Level 4](#)

[Media Queries Level 5](#)

03_MediaQueriesSlides

Media query – break points

het punt waarop we layout kenmerken van een pagina gaan wijzigen noemen we een breakpoint

```
@media screen and (max-width : 480px) {  
    /* 480px is a breakpoint */  
}
```

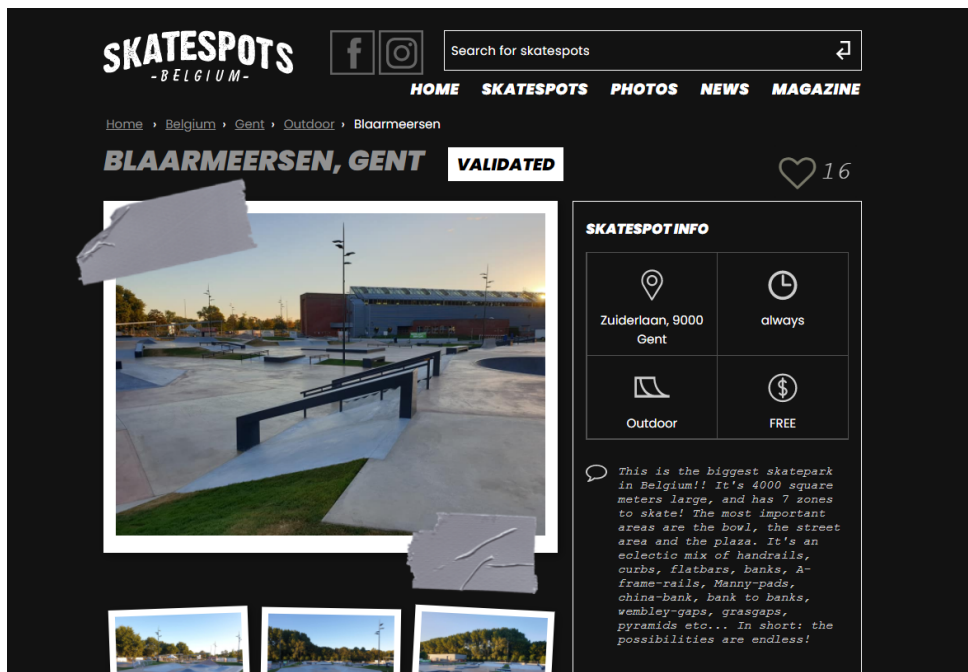
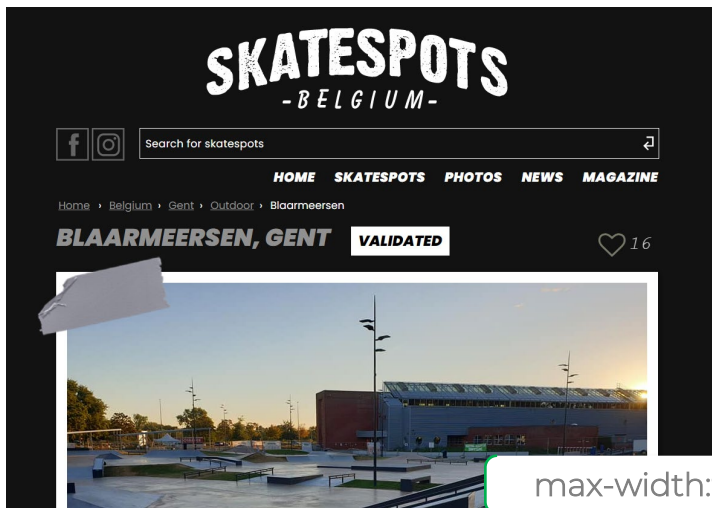
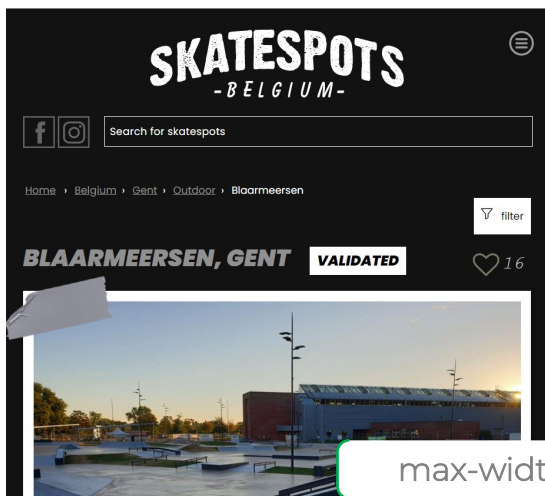
tips voor keuze van breakpoints

320px-480px	Mobile devices
481px-768px	iPads, Tablets
769px-1024px	Small screens, laptops
1025px-1200px	Desktops, large screens
1201px and more	Extra large screens, TV

... maar laat je vooral leiden door de inhoud van de pagina om 'jouw' breakpoints te vinden...

Voorbeeld – skatespots.be

layout wijzigt 3 keer – zie bv. logo;
navigatie; (hamburger)menu; kolommen; ...



Enkele tips

- zorg voor een tap-area van minstens **45 op 45 pixels** voor buttons, links, ... op schermen zonder accuraat pointing device
 - gemiddelde afdruk vinger 40 x 40 pixels
- maak minimaal gebruik van absolute waarden in CSS, maak gebruik van relatieve waarden
 - %, vw, em, rem, ...
- gebruik max-width: 100% op img elementen zodat de afbeelding niet uit hun container kunnen vloeien
- layout is geen exacte wetenschap, gebruik tips, baseer je op best practices, maar gebruik ook je **gevoel** om te beslissen of iets al dan niet OK oogt



Responsive Web Design

Responsive Images

De complexiteit van afbeeldingen

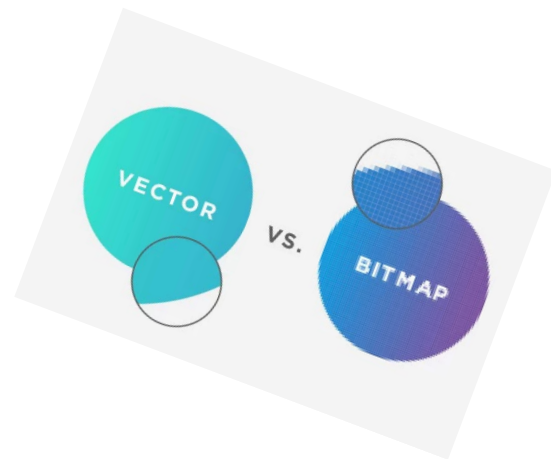
an image is considered **responsive** if it retains its quality on varying device sizes without having an unpleasant impact on performance.

- afbeelding dragen voor meer dan 60% bij in dataverkeer
- afbeeldingen wegen zwaar op resources zoals **geheugen, processor en bandbreedte** en beïnvloeden grotendeels de **laadtijd**
 - amazon.com: elke 100ms extra laadtijd resulteert in 1% verlies aan verkoop
 - als laadtijd van een pagina van 1 naar 3 seconden gaat verhoogt de kans met 32% dat de site verlaten wordt
- **afbeeldingen schalen = verlies aan kwaliteit**
 - als afbeeldingsbestanden te groot zijn voor het gewenste formaat op scherm verkwist je dus bandbreedte, geheugen en processor tijd (de browser moet schalen!) maar bovendien verlies je ook aan kwaliteit...
 - als afbeeldingsbestanden te klein zijn heb je grote kans op een pixelated afbeelding op de pagina



Formaten

- **bitmap** based
 - afbeelding is raster van pixels
 - elke pixel is een waarde die een kleur voorstelt
 - foto realisme
 - bij schalen verlies je aan kwaliteit
 - bestandsgrootte hangt af van
 - resolutie, kleurdiepte, compressie-techniek
- **vector** based
 - coördinaten en geometrische vormen
 - niet foto realistisch
 - kleine bestandsgrootte
 - schalen zonder verlies aan kwaliteit



Formaten

- vergelijking
 - zie bv. <https://socialcompare.com/en/comparison/image-file-formats>
 - ook dit is evolving science, er zijn enkele veel belovende “nieuwere” formaten
 - betere kwaliteit, kleurdynamiek, compressie, bestandsgrootte dan de “traditionele” formaten
 - **webp**
 - **avif**

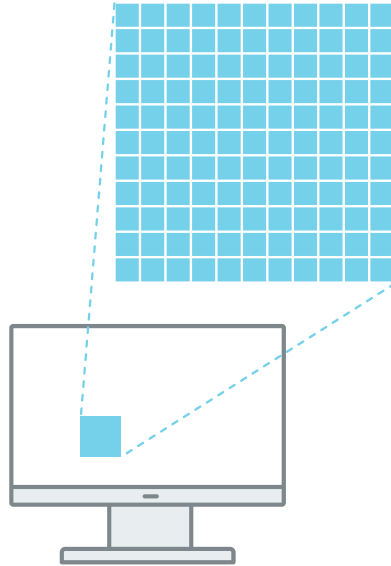
Pixels

- device pixel - hardware pixel
- CSS pixel - logische pixel
- vroeger vielen device pixels en CSS pixels 1 op 1 samen
- met de komst van retina (high density) schermen is er een gap ontstaan tussen de twee

Pixels

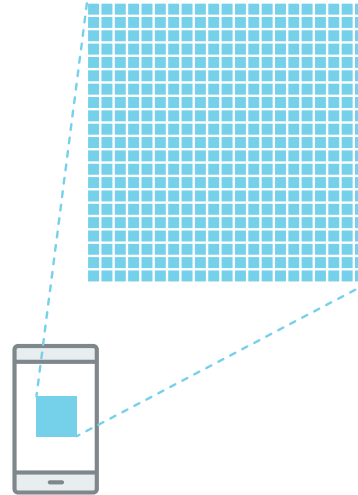
Low/medium density screens

Common desktop, laptop,
and older phone screens



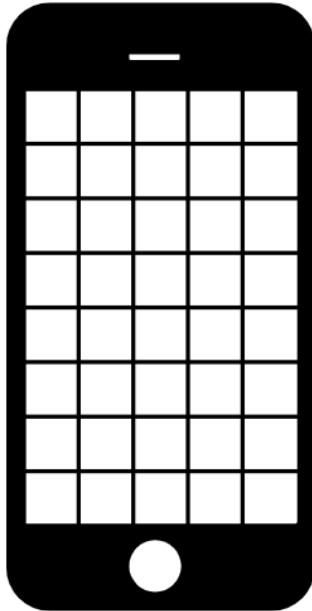
HDPI or 'Retina' screens

Most modern smart phones, tablets
and high-end laptops

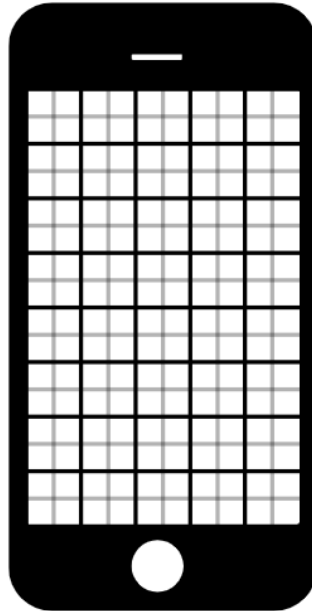


Pixels

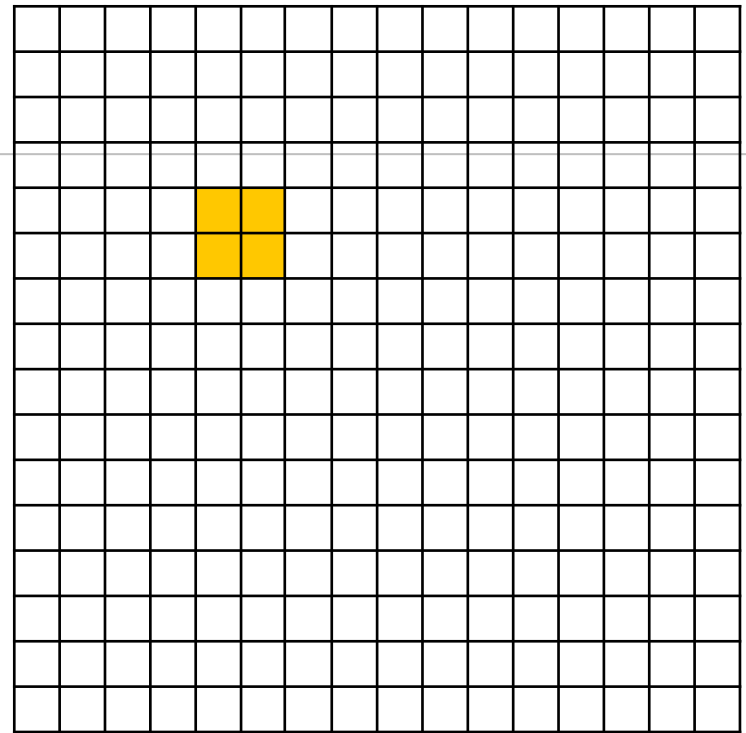
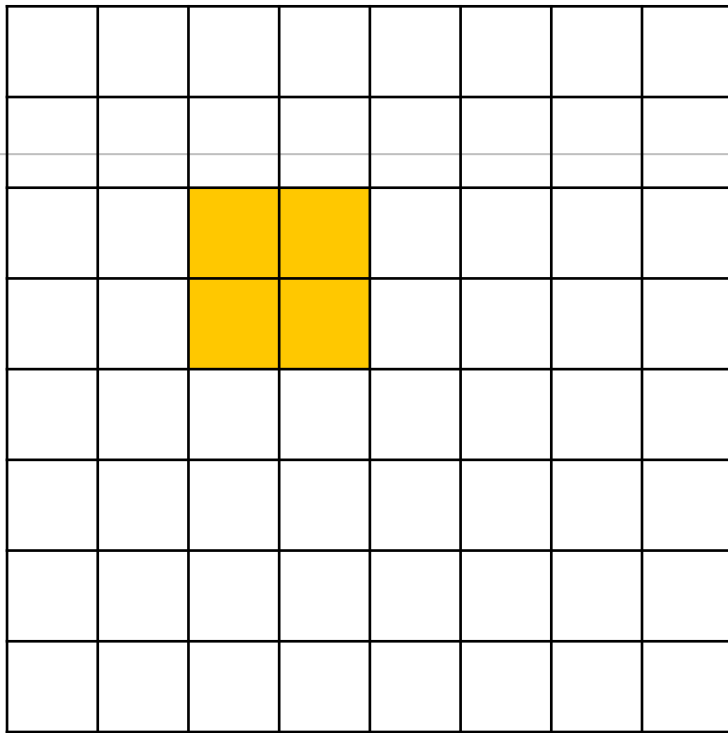
iPhone 3



iPhone 4

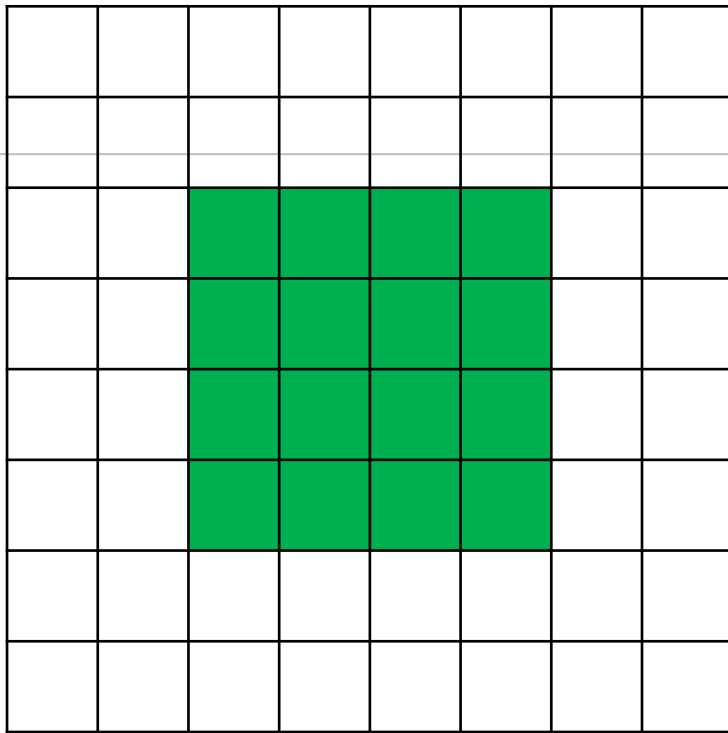


check de device pixel ratio voor
jouw scherm via
<https://johankj.github.io/devicePixelRatio/>

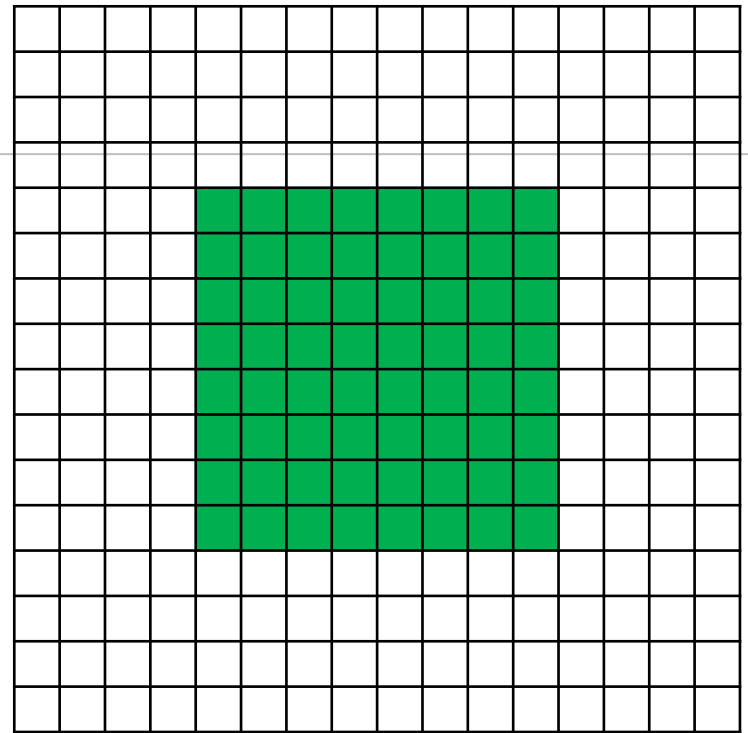


afbeelding van 2 x 2 pixels vult veel kleiner deel op een HD screen;

om **dezelfde grootte** te behouden en de **scherpte van het HD screen** te benutten moeten we het aantal pixels van de afbeelding verhogen naar 4 x 4



low density screen
1 device pixel \approx 1 CSS pixel
1x device pixel ratio



high density screen
2 x 2 device pixel \approx 1 CSS pixel
2x device pixel ratio

Pixels



6 android phones met
verschillende pixel density



6 afbeeldingen met
verschillende resolutie om
gelijke grootte op phone te
bekomen

Afbeelding kiezen

only one image should load, even if many are specified

- een afbeelding moet scherp zijn
 - schermen met hoge resolutie -> afbeeldingen met hoge resolutie
 - schermen met lage resolutie -> afbeeldingen met lage resolutie zodat geen bandbreedte verkwist wordt
- afbeeldingen moeten kunnen krimpen en uitzetten
- afbeeldingen moeten soms bijgesneden worden
 - art direction
- afbeeldingen kunnen aangeboden worden in verschillende formaten
 - zorg ervoor dat alternatieven voorhanden zijn indien formaat niet ondersteund wordt door browser

Do Not...

```
<body>
  <div class="fake-responsive">
    
    
  </div>
</body>
```

```
@media (max-width: 500px) {
  .big {
    display: none;
  }
}

@media (min-width: 501px) {
  .tiny {
    display: none;
  }
}
```



Naam	Status	Type	Initiator	Grootte	Tijd	Waterval
index.html	200	document	Anders	2.4 kB	2.05 s	
style.css	200	stylesheet	index.html	496 B	2.01 s	
tiny.jpg	200	jpeg	index.html	30.1 kB	3.21 s	
big.jpg	200	jpeg	index.html	370 kB	10.02 s	
ws	101	websocket	index.html:39	0 B	In behand...	
extn-utils.html	200	document	jquery-3.1.1...	1.1 kB	4 ms	
extn-utils.js	200	script	chrome-exten...	1.2 kB	2 ms	

op deze manier worden **steeds de 2 afbeeldingen** geladen

Resolutie?

04_IntroductionImages

```
<body>  
    
</body>
```

```
img {  
  max-width: 100%;  
}
```

- de afbeelding schaalt mee met venster



- mogelijks maak je gebruik van een relatief groot bestand om kwaliteit te garanderen op een hoog resolutie scherm, en...
 - ... wordt ditzelfde 'groot' bestand ook bij lage resolutie schermen geladen en verkwist je bandbreedte
- mogelijks maak je gebruik van een relatief klein bestand om laadtijd in te korten, en...
 - ... wordt je afbeelding niet mooi getoond op een hoog resolutie scherm



Resolution switching!

- wij bieden een aantal versies van eenzelfde afbeelding aan
- de browser gaat zelf kiezen welke versie geladen wordt
- wij helpen de browser door wat extra informatie te voorzien bij elke versie
 - pixel density, width/sizes
- de browser kan naast onze informatie ook gebruik maken van andere informatie, we kunnen dus niet voorspellen welke versie geladen zal worden
 - viewport afmetingen
 - kwaliteit van het netwerk
 - voorkeursinstellingen van de gebruiker

Resolution switching

- concreet
 - gebruik maken van srcset attribuut bij het **img-element**
 - op deze manier bied je verschillende versies aan
 - extra informatie voorzien bij elke versie:
 - optie 1: pixel density beschrijven
 - optie 2: grootte beschrijven

img met **srcset** attribuut

- voorbeeld 1 – **pixel density descriptor**

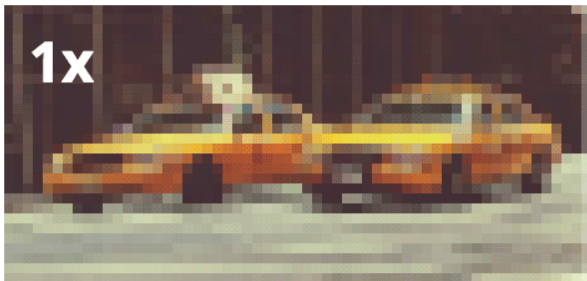
```

```

- **srcset** attribuut heeft als waarde een **lijst van image URLs**
- bij elke URL staat ook een **x-descriptor** die aangeeft voor welke pixel density het bestand bedoeld is
- de URL in het **src** attribuut wordt gebruikt door browsers die srcset niet ondersteunen

nadeel is dat geen rekening wordt gehouden
met de grootte waarop de afbeelding zal gerendered worden

srcset met pixel density descriptor



Pixels exaggerated
for effect



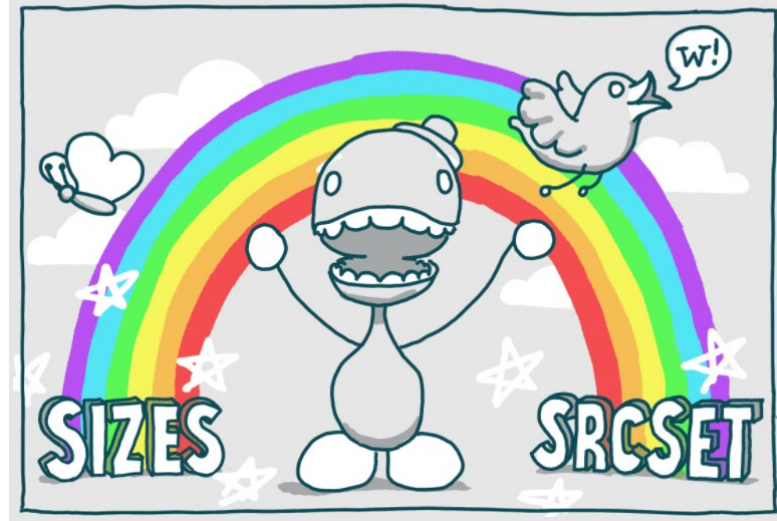
nadeel is dat geen rekening wordt gehouden
met de grootte waarop de afbeelding zal gerendered worden

**HO
GENT**

Let's share our knowledge!

VARIABLE	KNOWN BY AUTHOR WHEN SHE'S WRITING THE CODE?	KNOWN BY BROWSER WHEN IT'S LOADING THE PAGE?
viewport dimensions	no	yes
image size relative to the viewport	yes	no
screen density	no	yes
source files' dimensions	yes	no

VARIABLE	KNOWN BY AUTHOR WHEN SHE'S WRITING THE CODE?	KNOWN BY BROWSER WHEN IT'S LOADING THE PAGE?
viewport dimensions	no	yes
image size relative to the viewport	yes	no yes! via sizes !
screen density	no	yes
source files' dimensions	yes	no yes! via srcset !



img met **srcset** attribuut

06_ResolutionSwitching_Width

- voorbeeld 2 – **width descriptor**

```

```

- **srcset**: bij elke URL staat nu een w-descriptor die de exacte breedte (in pixels) aangeeft van de afbeelding
- het **sizes attribuut** verduidelijkt voor de browser hoeveel pixels er effectief nodig zijn door de breedte van de gerenderde image aan te geven

op deze manier kent de browser de eigenschappen van de afbeelding EN de grootte van de uiteindelijk gerenderde afbeelding

img met **srcset** attribuut

- voorbeeld 2 – **het sizes attribuut**

```

```

```
sizes="[media query] [length],  
       [media query] [length],  
       etc...  
       [default length]"
```

- het **sizes** attribuut bevat een **breedte** (CSS lengte)
 - klassieke eenheden: 55px, 10em, ...
 - of relatief t.o.v. de breedte van de viewport; eenheid: **vw**
 - 33,3vw, i.e. een derde van de breedte van de viewport
 - 100vw, i.e. de volledige breedte van de viewport
- de lengte wordt gekoppeld aan een **media query**, deze worden sequentieel overlopen tot de eerst passende gevonden wordt

I want control!

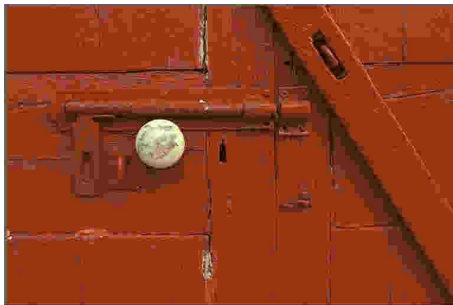
- bij gebruik van **srcset** heb je zelf geen controle over de versie van de afbeelding die de browser zal laden
 - enkel te gebruiken voor verschillende versies van 1 en dezelfde afbeelding!
- soms wil je **zelf controle** over de afbeelding die geladen wordt
 - kiezen uit verschillende afbeeldingen, uit anders uitgesneden afbeeldingen, uit afbeeldingen in verschillende formaten
- maak in deze gevallen gebruik van het **picture element**

Het picture element – use cases

- art direction



- alternatieve afbeeldingsformaten



.jpeg (14 KByte)



.avif (4 KByte)

Het picture element

- **art direction:** jij bepaalt welke afbeelding er onder welke condities moet gebruikt worden
 - je kan totaal verschillende afbeeldingen aanbieden
 - aangepast aan de viewport
 - je kan afbeeldingen aanbieden die op verschillende manieren uitgesneden zijn
 - focus op element van afbeelding behouden
 - typische toepassing: hero image
- aanbieden **alternatieve afbeeldingsformaten**
 - **.webp** en **.avif** zijn niet voor niets upcoming...

Het picture element

07_ArtDirection

```
<picture>
  <source srcset="images/runner-wide.jpg" media="(min-width: 1200px)">
  <source srcset="images/runner-narrow.jpg" media="(min-width: 700px)">
  
</picture>
```

- **source** elementen
 - **srcset** attribuut met URL naar een afbeelding en
 - **media** attribuut met query die bepaalt wanneer deze afbeelding gebruikt wordt
 - volgorde is belangrijk!
- het **img** element wordt gebruikt om het geselecteerde element te tonen
 - de URL van het src attribuut wordt enkel gebruikt indien picture niet ondersteund is door browser, of indien geen enkele media query voldoet

Het picture element

- gebruik “nieuwe” afbeeldingsformaten zoals **.webp** en **.avif**
 - betere compressie (kleinere bestanden)
 - betere kleurdiepte, dynamiek, ...
 - animatie, transparantie, ...
- maar voorziet via het picture element een **fallback**
 - voor browsers die het formaat niet ondersteunen

Het picture element

08_NewFileFormats

```
<picture>
  <source srcset="example.avif" type="image/avif">
  <source srcset="fallback.webp" type="image/webp">
  
</picture>
```

- via het **srcset- & type-attribuut** kan je in verschillende formaten voorzien
- via het **src-attribuut** kan je steeds een fallback voor de browser voorzien

Het picture element

```
<picture>
  <source srcset="images/runner-wide-2x.jpg 2x, images/runner-wide.jpg"
    media="(min-width: 1200px)">
  <source srcset="images/runner-narrow-2x.jpg 2x, images/runner-narrow.jpg"
    media="(min-width: 700px)">
  
</picture>
```

- in het **srcset attribuut** kan je ook weer gebruik maken van een lijst van URL's met **pixel density/width descriptors**
 - **combinatie van art direction & resolution switching**
 - wordt serieus complex ☹

Images samengevat

`<picture>`



Image optimized for **content reasons** matters more than technical reasons.



Author chooses the best image.



Images may be very different in composition and dimension.

`src, srcset`



Image optimized for **technical reasons** matters more than content reasons.



Browser chooses the best image.



Images vary in dimension but not composition.

**HO
GENT**