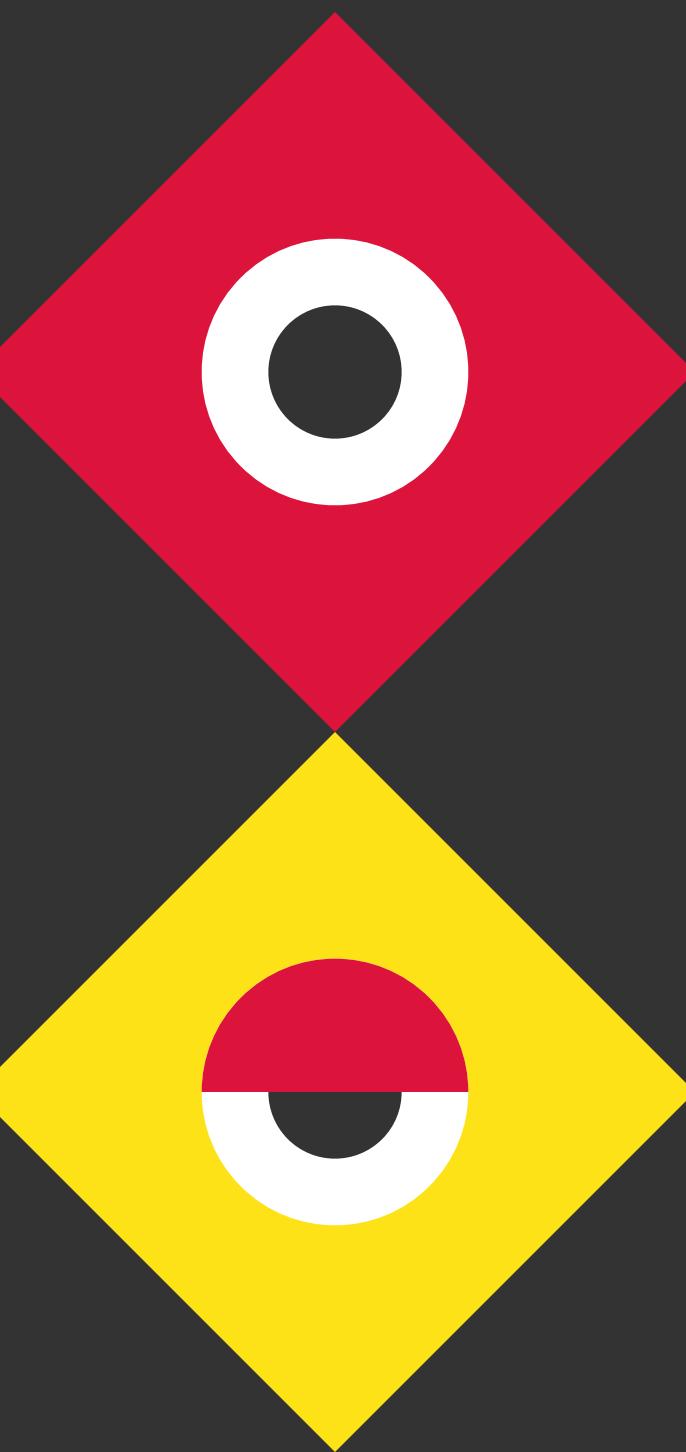




Oefeningen animeren recap



Wat te doen

Om weer helemaal op stoom te komen met CSS animaties en CSS custom properties kun je:

1a. Bronnen doornemen/bekijken

- Video: CSS Animation in 100 Seconds:
<https://youtu.be/HZHHBwzmJLk>
- Video: An introduction to CSS custom properties
https://youtu.be/PHO6TBq_auI

1b. Recap animeren, properties en kleurtjes

2. Oefening maken

Bij de oefening is een uitwerking beschikbaar
(niet meteen spieken ;-)

- Oefening 1: Recap the animation properties



De code voor de oefeningen vind je op codepen.io

- Maak **een account** aan op codepen.
- Volg de **link naar onze code** voor de oefening.

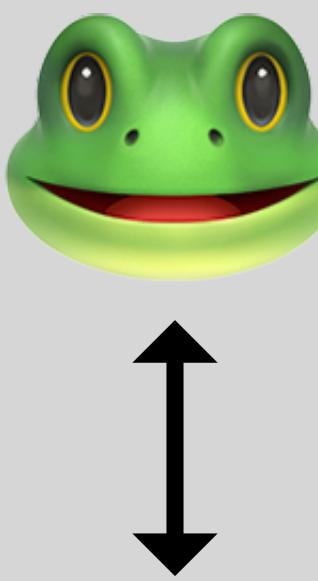
3. Fork de code

Door de code te forken maak je een kopie van ~~onze code~~. Die kopie wordt toegevoegd aan jouw account. Dan kun je je code opslaan, later verder bewerken en in de werkgroep laten zien.

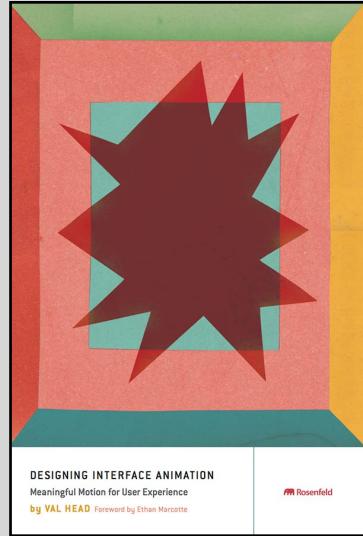
The screenshot shows a CodePen project titled "3D rotate". The preview area displays three 3D cubes. The first cube has faces labeled 3, 4, and 5. The second cube has faces labeled 4, 5, and 6. The third cube has faces labeled 5, 6, and 7. The fourth cube has faces labeled 6, 7, and 8. The fifth cube has faces labeled 7, 8, and 9. The sixth cube has faces labeled 8, 9, and 10. The HTML panel contains code for articles and sections. The CSS panel contains styles for the body and html elements. The JS panel contains a single line of code. The interface also includes buttons for Save, Settings, Change View, and a fork icon.

De "fork" knop zit daar

Animeren recap



Animeren recap



[Val Head](#) heeft het interessante boek 'Designing Interface Animation' geschreven, waar ze een aantal **redenen** geeft om animaties in een user interface toe te passen:

- Relaties tussen elementen leggen

bijv: een filter dat in- en uitschuift zodat je weet hoe 'de wereld' in elkaar zit en samenhangt.

- Aandacht op iets vestigen

bijv. nieuwe content na een refresh met een animatie toevoegen aan een pagina, zodat het even het focuspunt wordt.

- Feedback geven

bijv. een winkelwagentje even groter laten worden als er een product wordt toegevoegd, zodat duidelijk is dat het toevoegen gelukt is en waar het product is terug te vinden.

- iets uitleggen

bijv. gebruikers de eerste keer een animatie van een gesture tonen, zodat ze weten hoe de interactie werkt.

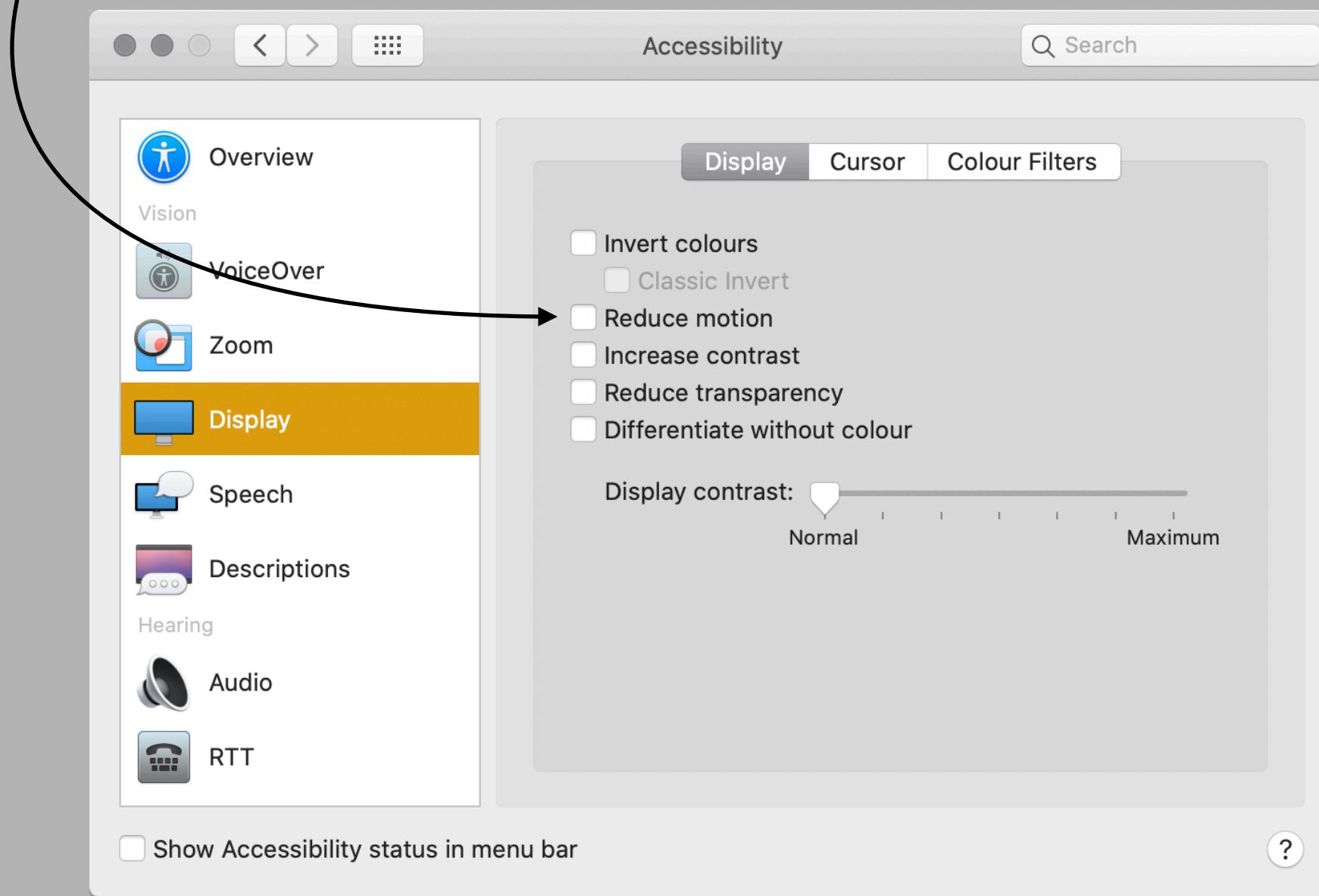
- Het merk karakter geven

De branding ook smoel geven met de wijze van animeren.

Pas ook op

Animaties zijn tof en nuttig (zeker als je ze spaarzaam inzet). Maar er zijn ook mensen die misselijk of duizelig van animaties worden of zelfs epileptische aanvallen krijgen.

Gelukkig is er hulp. In de settings van je device kun je de **optie 'Reduce Motion'** aanzetten. Die setting kun je als developer vervolgens [met een media query uitlezen](#) om een variant van je website te maken met verantwoorde of helemaal geen animaties.



Nb. Je ziet **meer opties** waarmee je rekening kunt houden. Die kun je [allemaal uitlezen met media queries](#) (soms nog experimenteel)

Anatomie animatie 1/4

1. Element bepalen dat gaat animeren

2. Animatie definiëren
3. Animatie aan (status van) element koppelen
4. Duur van de animatie bepalen



De HTML

```
<div>❤</div>
```

De CSS

Animatie anatomie 2a/4

1. Element bepalen dat gaat animeren

2. Animatie definiëren

3. Animatie aan (status van) element koppelen

4. Duur van de animatie bepalen



De tweede stap is de animatie definiëren. Dat doe je met @keyframes. Daarachter staat de naam van de animatie. Die mag je zelf bedenken. Handig om de animatie een betekenisvolle naam te geven, zodat je later de code nog snapt (en anderen ook).

De HTML

```
<div>❤</div>
```

Naam van
de animatie

De CSS

```
@keyframes hartjeDraaien {  
}
```

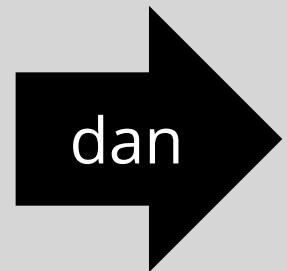
Animatie anatomie 2b/4

1. Element bepalen dat gaat animeren

2. Animatie definiëren

3. Animatie aan (status van) element koppelen

4. Duur van de animatie bepalen



Start

Vervolgens definieer je het startpunt van de animatie met 0% {}.

Daarbinnen zet je de properties met de waarden die het element bij de start van de animatie moet hebben.

Einde

Dan definieer je het eindpunt van de animatie met 100% {}.

Daarbinnen zet je de properties met de waarden die het element bij het einde van de animatie moet hebben.

In het voorbeeld zal het element 180 graden gedraaid gaan worden.

Startpunt van
de animatie

De HTML
`<div>❤</div>`

Eindpunt van
de animatie

De CSS
`@keyframes hartjeDraaien {
 0% { transform: rotate(0deg); }
 100% { transform: rotate(180deg); }
}`

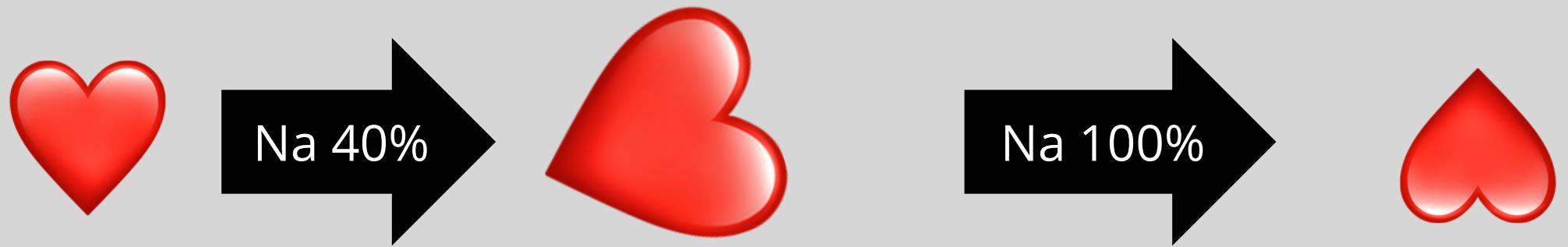
Animatie anatomie 2c/4

1. Element bepalen dat gaat animeren

2. Animatie definiëren

3. Animatie aan (status van) element koppelen

4. Duur van de animatie bepalen



De HTML

```
<div>❤</div>
```

De CSS

```
@keyframes hartjeDraaien {  
    0% {  
        transform: rotate(0deg);  
    }  
    40% {  
        transform: rotate(72deg) scale(2);  
    }  
    100% {  
        transform: rotate(180deg);  
    }  
}
```

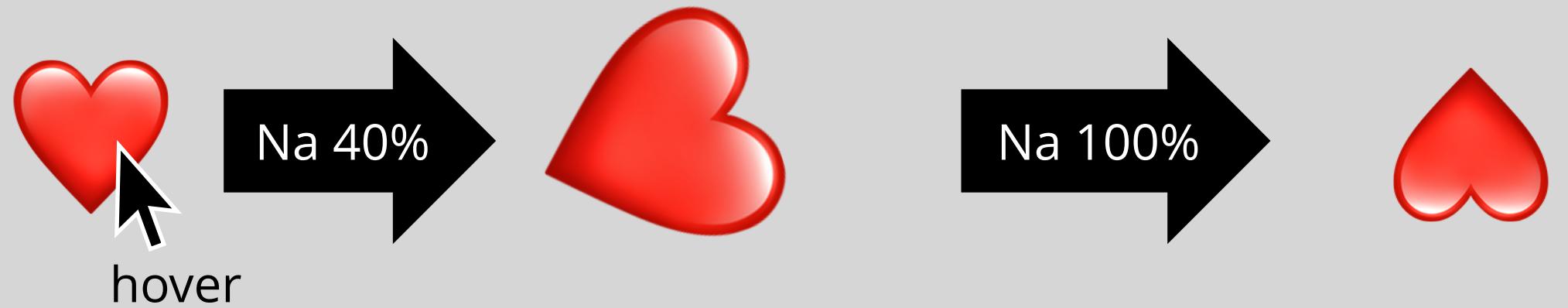
Extra tussenstops

Een animatie kan uitgebreid worden met meer tussenstops. In het voorbeeld wordt het hartje op 40% van de animatie 2x zo groot om dan verder te draaien en weer de normale grootte aan te nemen.

Nb. Niet alle CSS properties kun je animeren. Kijk voor een lijst met properties die je kunt animeren op: developer.mozilla.org/en-US/docs/Web/CSS/CSS_animated_properties

Animatie anatomie 3/4

1. Element bepalen dat gaat animeren
2. Animatie definiëren
- 3. Animatie aan (status van) element koppelen**
4. Duur van de animatie bepalen



Animation-name

De volgende stap is het koppelen van de animatie aan het element dat moet gaan animeren. Dat doe je door het element (of een status van het element) te selecteren. Om daarbinnen met animation-name de animatie te koppelen.

In het voorbeeld wordt de animatie gekoppeld aan de hover-status van de div. De animatie wordt dus geactiveerd als de gebruiker over de div hovert.

De HTML

```
<div>❤</div>
```

De CSS

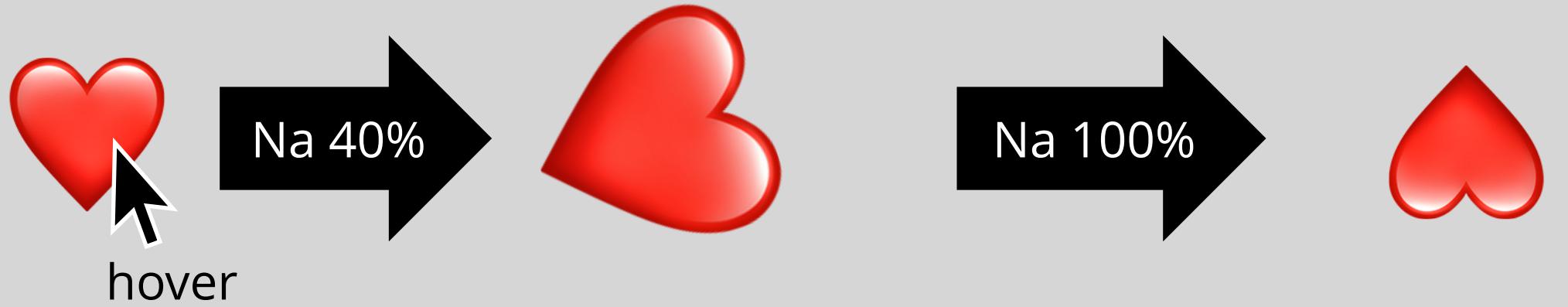
```
@keyframes hartjeDraaien {  
    0% {  
        transform: rotate(0deg);  
    }  
    40% {  
        transform: rotate(72deg) scale(2);  
    }  
    100% {  
        transform: rotate(180deg);  
    }  
}  
  
div:hover {  
    animation-name: hartjeDraaien;  
}
```

De property
animation-name

Animatie anatomie 4a/4

1. Element bepalen dat gaat animeren
2. Animatie definiëren
3. Animatie aan (status van) element koppelen

4. Duur van de animatie bepalen



De HTML

```
<div>❤</div>
```

De CSS

```
@keyframes hartjeDraaien {  
    0% {  
        transform: rotate(0deg);  
    }  
    40% {  
        transform: rotate(72deg) scale(2);  
    }  
    100% {  
        transform: rotate(180deg);  
    }  
}  
  
div:hover {  
    animation-name: hartjeDraaien;  
    animation-duration: 1s;  
}
```

De property
animation-duration

Animation-duration

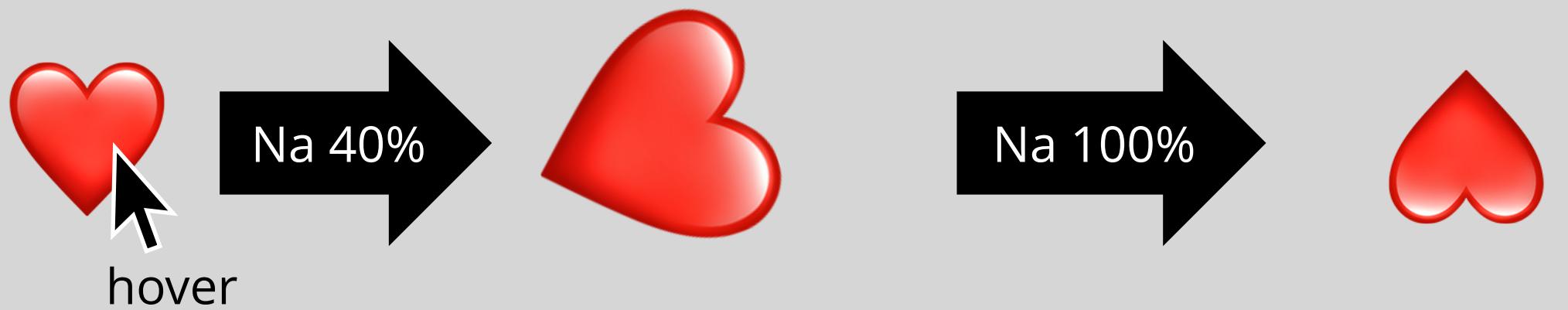
Tenslotte bepaal je hoe lang de animatie gaat duren. Dit doe je met de property animation-duration.

In het voorbeeld is de duur 1 seconde. Het hartje zal bij hoveren dus in 1 seconde 180 graden draaien.

Animatie anatomie 4b/4

1. Element bepalen dat gaat animeren
2. Animatie definiëren
3. Animatie aan (status van) element koppelen

4. Duur van de animatie bepalen



De HTML

```
<div>❤</div>
```

De CSS

```
@keyframes hartjeDraaien {  
    0% {  
        transform: rotate(0deg);  
    }  
    40% {  
        transform: rotate(72deg) scale(2);  
    }  
    100% {  
        transform: rotate(180deg);  
    }  
}
```

```
div:hover {  
    animation-name: hartjeDraaien;  
    animation-duration: 1s;  
    animation-delay: 1s;  
    animation-timing-function: linear;  
    animation-iteration-count: 10;  
}
```

of bijv:

```
    animation-iteration-count: infinite;  
    animation-direction: alternate;
```

De animatie start na 1 seconde

Het hartje draait in een temp
(er is geen versnelling) - dat ziet er meestal wat doods uit

De animatie wordt 10x uitgevoerd.
Het hartje draait 180 graden - en springt dag weer terug - en dan 10x.

De animatie blijft voor altijd herhalen

De animatie draait heen en draait dan ook weer terug) om-en-om)

Andere animation properties

Uitleg over meer animation properties: developer.mozilla.org/en-US/docs/Web/CSS/CSS_Animations/Using_CSS_animations (MDN)

CSS custom props recap

```
:root {  
  --color-text:crimson;  
}  
  
h1 {  
  color:var(--color-text);  
}
```

CSS custom props recap

Met custom properties kun je waarden (bijv. een kleur) die je vaker wilt gebruikt op één plek definiëren (en wijzigen) en dan overal toepassen.

Deze week gaan we vooral oefenen met kleur en thema's, maar custom properties kun je voor veel meer gebruiken. Je kunt er zelfs mee rekenen.

Een eis voor beide opdrachten is dat je custom properties gebruikt voor kleuren en verloopjes.

Daarnaast moeten beide opdrachten dark en light mode ondersteunen. Ook daar zijn custom properties geweldig voor.

CSS custom properties worden in de volksmond ook wel CSS variabelen genoemd.

CSS custom properties worden meestal in de :root selector gedefinieerd. Het kan ook in andere selectoren, maar voor nu is de :root prima. Ze zijn dan overal beschikbaar. Net als globale variabelen in JS.

1e voorbeeld

```
:root {  
    --hoofdKleur:purple;  
    --accentKleur:darkorange;  
    --achtergrondKleur:white;  
}
```

Een custom property begint met 2 streepjes en dan een zelfgekozen naam. Kies voor betekenisvolle namen. Dat is handig voor jezelf en als anderen met je code aan de slag gaan.

```
h1, h2, h3 {  
    color: var(--hoofdKleur);  
}
```

Vervolgens kun je waarden in de custom properties opslaan. In dit voorbeeld kleuren, maar dat kunnen ook andere waarden zijn.

```
button {  
    color: var(--hoofdKleur);  
    background-color: var(--achtergrondKleur);  
    border:solid 1px var(--hoofdKleur);  
}
```

Vervolgens kun je de properties op meerdere plaatsen in je CSS gebruiken. Dat doe je door ze aan te roepen binnen een var().

```
button:hover {  
    color: var(--accentKleur);  
    border-color: var(--accentKleur);  
}
```

Heading 1
Heading 2
Heading 3

button

hover

Meer dan kleur

Je gaat custom properties i.i.g. voor kleuren en verloopjes gebruiken. Custom properties kun je echter benutten voor veel meer: bijvoorbeeld afmetingen, fonts en strings.

Een custom property waar een afmeting in wordt opgeslagen.

2e voorbeeld

```
:root {  
  --hoofdKleur:purple;  
  --accentKleur:darkorange;  
  --achtergrondKleur:white;  
  --witruimte:1.5em;  
}
```

Die kun je vervolgens gebruiken voor een padding.

```
body {  
  padding: var(--witruimte);  
}
```

```
h1, h2, h3 {  
  color: var(--hoofdKleur);  
}
```

Je kunt met calc() ook rekenen met custom properties. Zo kun je bijv. alle paddings in verhouding baseren op 1 afmeting.

```
button {  
  padding: calc( var(--witruimte) / 2 );  
  
  color: var(--hoofdKleur);  
  background-color: var(--achtergrondKleur);  
  border:solid 1px var(--hoofdKleur);  
}
```

```
button:hover {  
  color: var(--accentKleur);  
  border-color: var(--accentKleur);  
}
```

Heading 1
Heading 2
Heading 3

button

hover

Properties in properties

Je kunt een custom property ook vullen met een andere custom property. Dat kan bijvoorbeeld handig zijn voor verloopjes.

Voor de color-stops in de gradient kun je eerder gedefinieerde custom properties gebruiken.

3e voorbeeld

```
:root {  
  --hoofdKleur:mediumpurple;  
  --titelKleur:purple;  
  --achtergrondGradient:linear-gradient(  
    var(--hoofdKleur),  
    var(--titelKleur)  
  );  
}
```

```
body {  
  color: var(--hoofdKleur);  
}
```

```
h1, h2, h3 {  
  color: var(--titelKleur);  
}
```

```
header {  
  color: white;  
  background-image: var(--achtergrondGradient);  
}
```

```
button {  
  color: white;  
  background-image: var(--achtergrondGradient);  
}
```

Je kunt ook een gradient opslaan in een custom property.

De custom property met daarin de gradient kun je vervolgens weer op meerdere plekken gebruiken.

Header

Heading 1

Heading 2

Heading 3

button

Dark/Light mode

Geheel in de trend van apps ondersteunen veel websites tegenwoordig ook light en dark mode.

Media query:

Met `@media (prefers-color-scheme:dark)` kun je checken of de gebruiker dark (of light) heeft ingesteld als voorkeur op een device.

Meer info: [prefers-color-scheme op MDN](#).

Dark als default

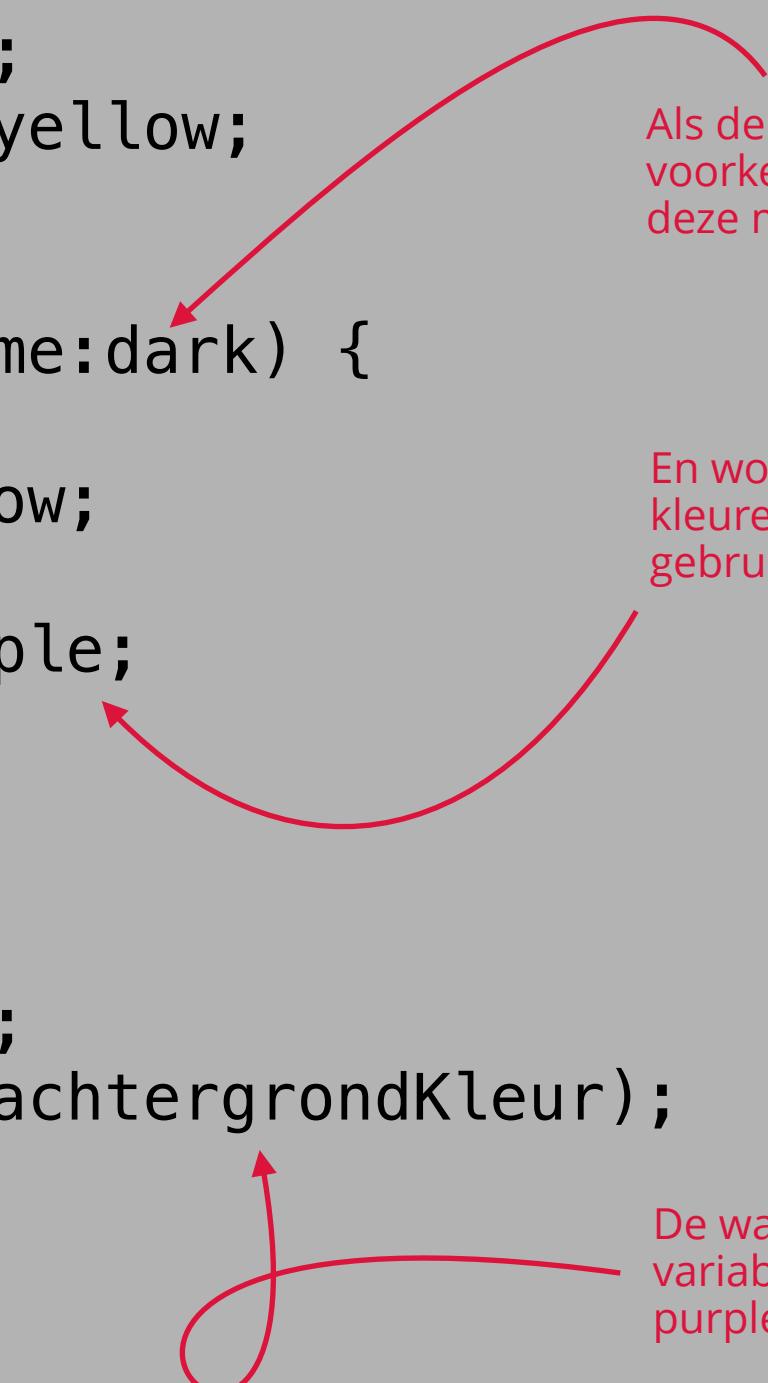
Tegenwoordig wordt zelfs steeds vaker dark als default gebruikt en wordt alleen gewisseld naar light als de gebruiker dat instelt. [Leuk artikel over Dark themes](#).

Knopje op je website

Veel gebruikers willen niet alle websites/apps dark of light. Ze willen dat per website/app kunnen instellen. Daarvoor kun je een knopje aanbieden. Die keuze natuurlijk wel bewaren voor de volgende keer (dat kan met local storage of in het profiel van de gebruiker als hij dat heeft).

4e voorbeeld

```
:root {  
  --hoofdKleur:purple;  
  --accentKleur:darkorange;  
  --achtergrondKleur:lightyellow;  
}  
  
@media (prefers-color-scheme:dark) {  
  :root {  
    --hoofdKleur:lightyellow;  
    --accentKleur:orchid;  
    --achtergrondKleur:purple;  
  }  
}  
  
body {  
  color: var(--hoofdKleur);  
  background-color: var(--achtergrondKleur);  
}
```



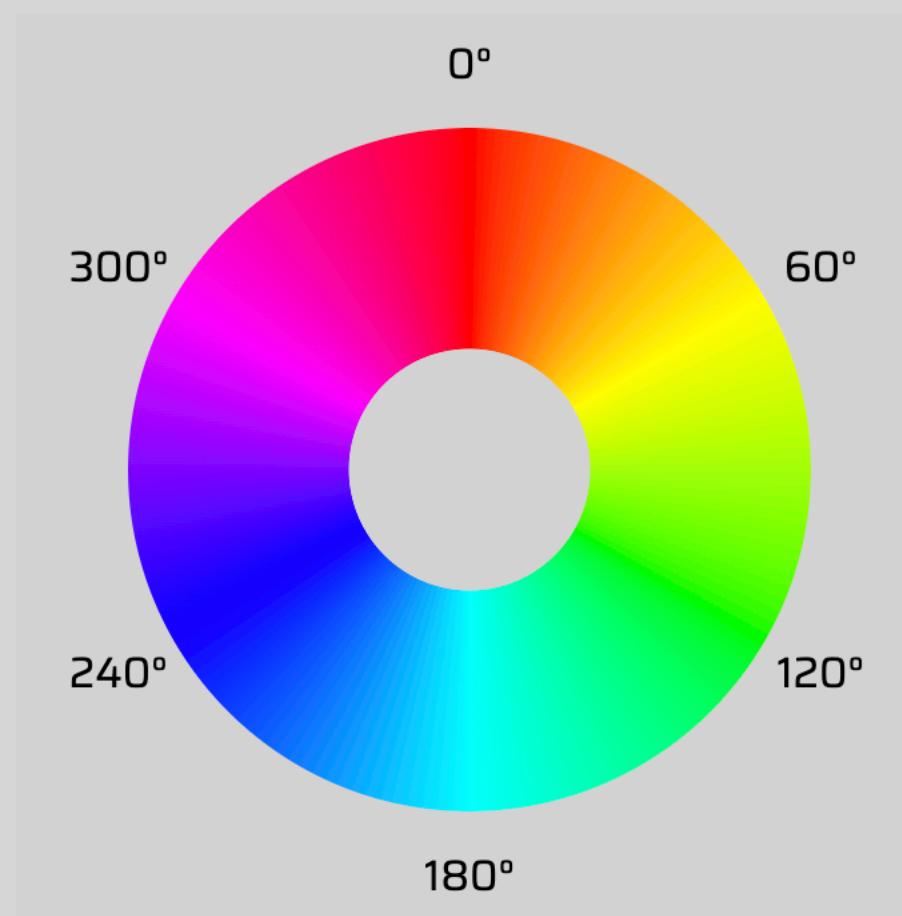
Als de gebruiker dark als voorkeur heeft ingesteld is deze media query true.

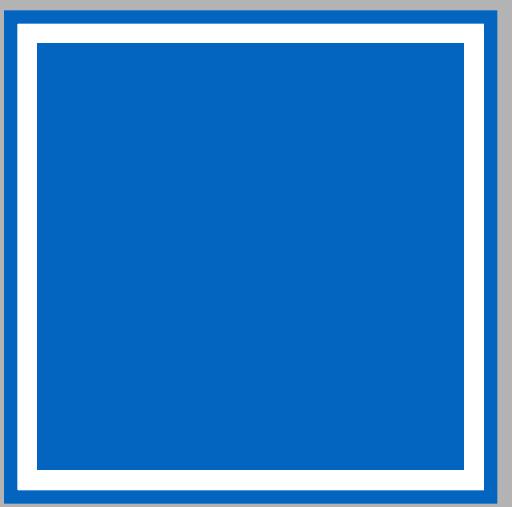
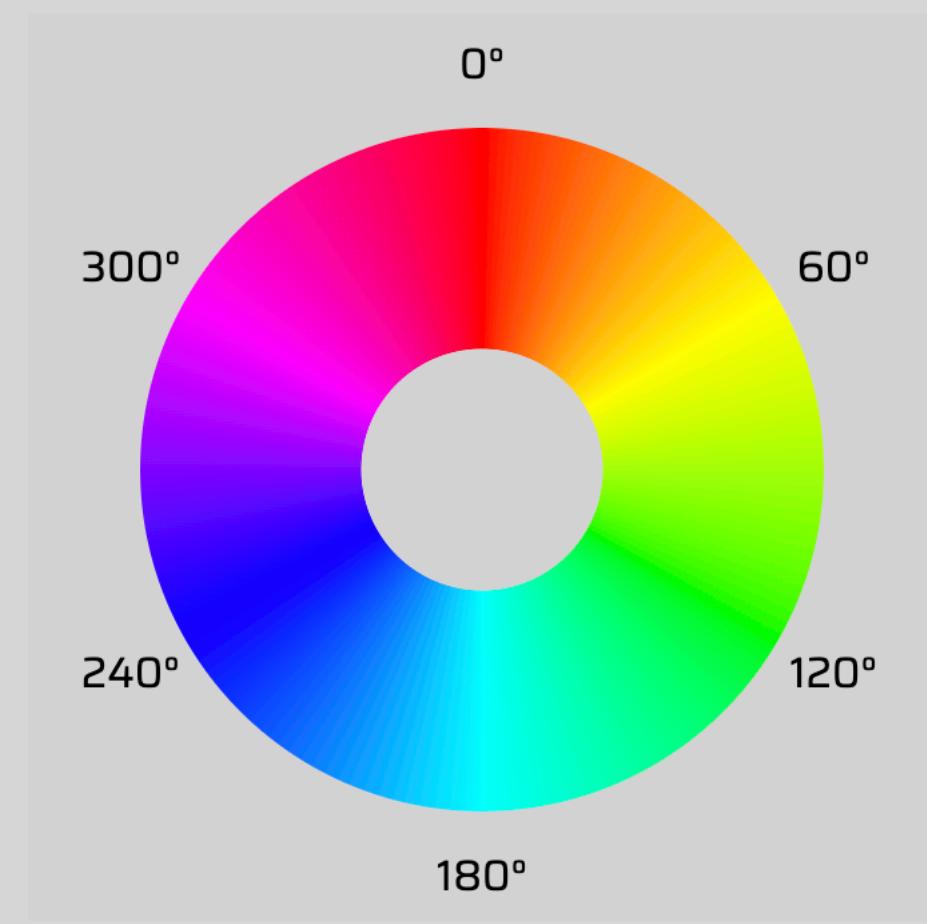
En wordt dit kleurenschema gebruikt.

De waarde van deze variabele is dan dus purple.

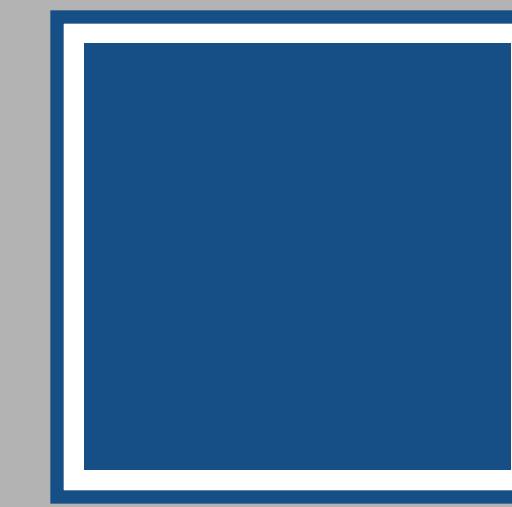


Recap kleurtjes

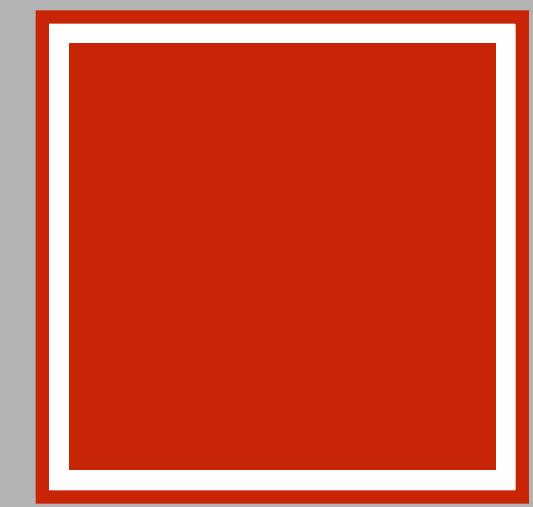




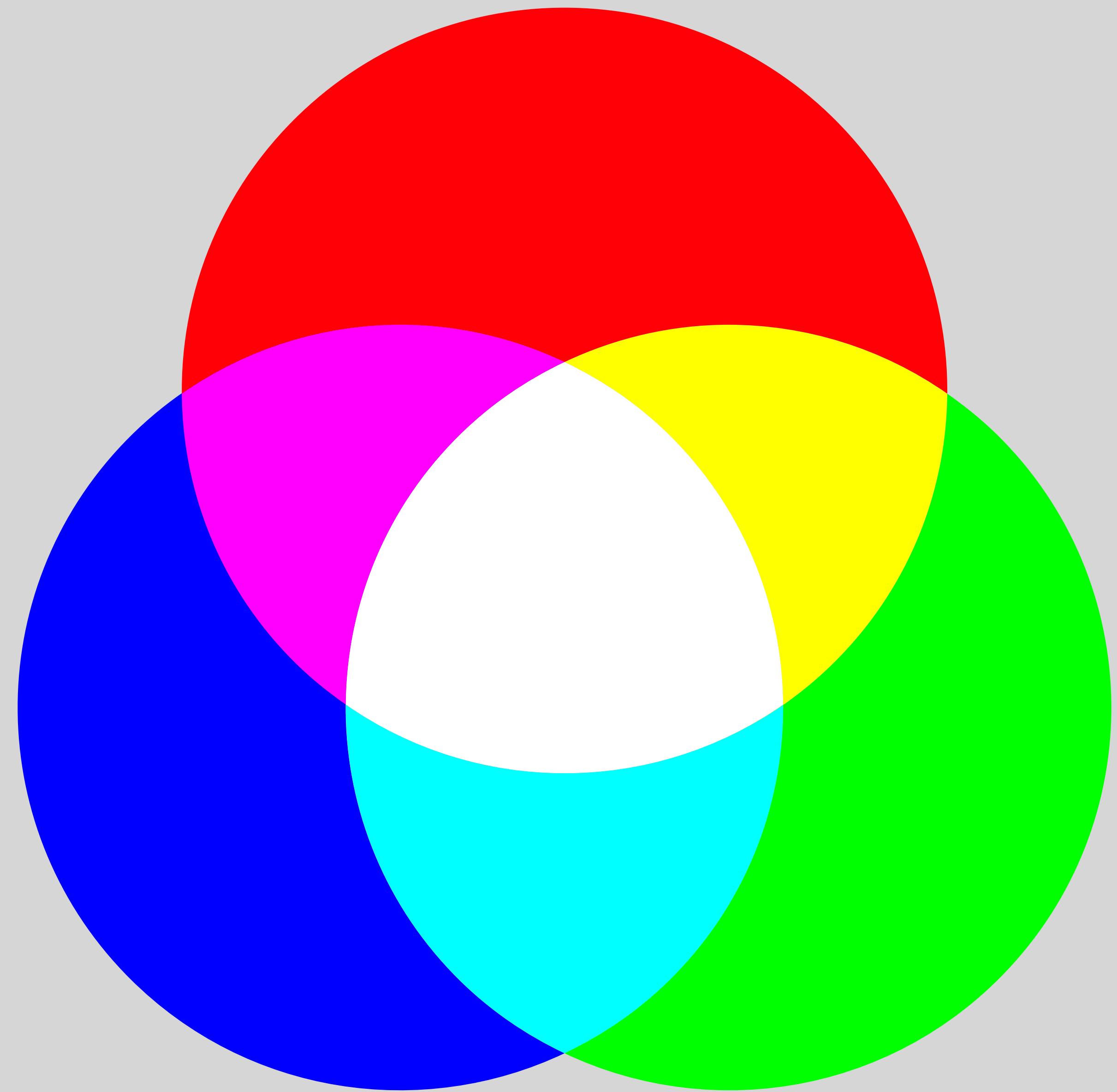
#0365C0
rgb(3 101 191)



#164F86
rgb(22 79 134)

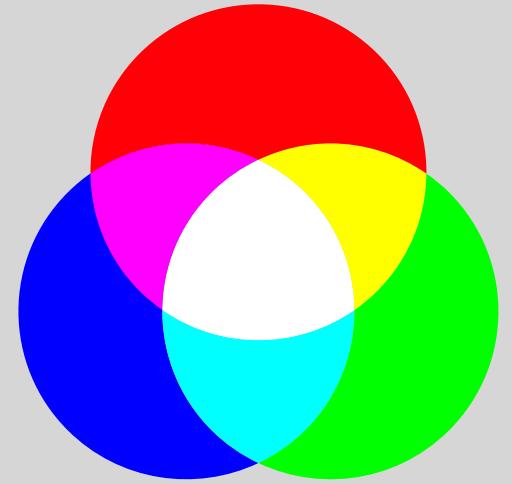


#C82506
rgb(200 37 6)



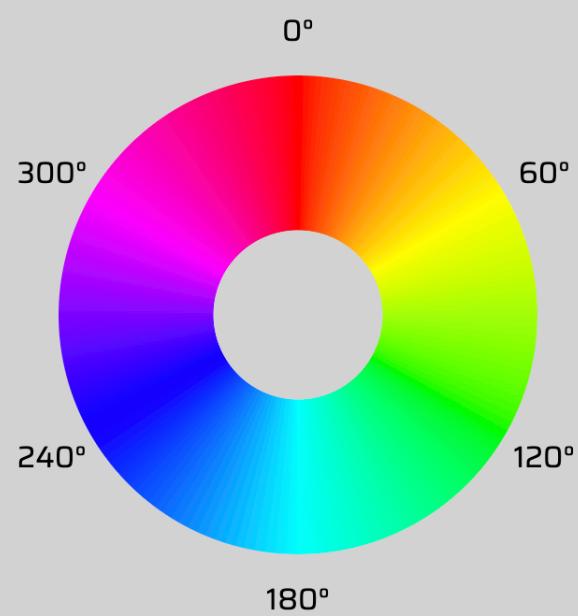
Kleurtjes 1/5

Schermen bestonden lang uit 3 kleuren fosforpuntjes: rood, groen en blauw. Alledrie uit is zwart, alledrie aan is wit en combinaties van de puntjes voor andere kleuren.



Kleuren definiëren in CSS
is/was een combi van **RGB**.

Moderne schermen geven kleur op een andere manier weer. Elke pixel straalt een bepaalde golflengte straling uit → daarmee wordt de kleur bepaald (de hue).



In lijn daarmee kun je **nu ook** de **hue** (kleur) definiëren en daarna de **saturation** (verzadiging) en **luminance** (helderheid).

Named colors:

color: Red;

RGB function:

color: rgb(255, 0, 0);

Hexadecimal values:

color: FF0000;

HSL function:

color: hsl(0deg, 100%, 50%);

Color function:

color: color(display-p3 1 0 0);



Kleurtjes 2a/4

De RGB function



De oude manier

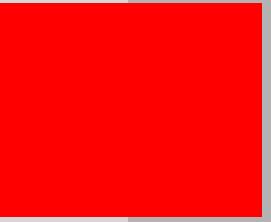
color: rgb(255, 0, 0);



Kleurtjes 2b/4

De RGB function

De oude manier
color: rgb(255, 0, 0);



De nieuwe manier
color: rgb(255 0 0);

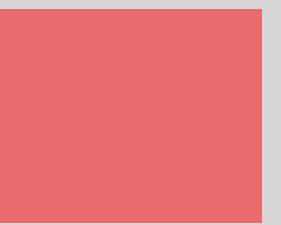


Geen komma's meer.



Kleurtjes 2c/4

De RGB function



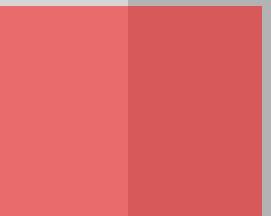
De oude transparantie
`color: rgba(255, 0, 0, .5);`



Kleurtjes 2d/4

De RGB function

De oude transparantie
color: rgba(255, 0, 0, .5);



De nieuwe transparantie
color: rgb(255 0 0 / .5);

Geen 'a' meer. De waarde voor het alpha channel is optioneel bij het aanroepen van de rgb() function.

Het alpha channel wordt gedefineerd door een '/' gevolgd door een waarde tussen 0 (doorzichtig) en 1 (zichtbaar).



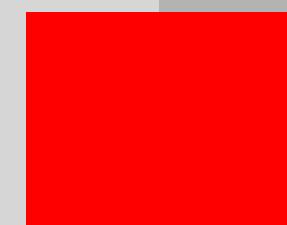
Kleurtjes 2e4

De RGB function

nb. Geen komma's en een optionele waarde voor het alpha channel geldt voor alle color() functions.

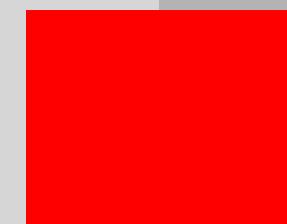
De oude manier

```
color: rgb(255, 0, 0);
```

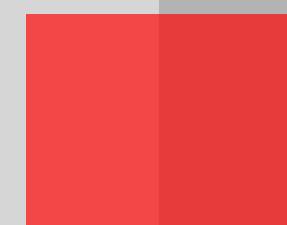


Oude (semi)-transparant

```
color: rgba(255, 0, 0, 1);
```



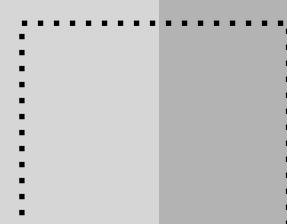
```
color: rgba(255, 0, 0, .67);
```



```
color: rgba(255, 0, 0, .33);
```



```
color: rgba(255, 0, 0, .0);
```



De nieuwe manier

```
color: rgb(255 0 0);
```

Oude (semi)-transparant

```
color: rgb(255 0 0 / 1);
```

```
color: rgb(255 0 0 / .67);
```

```
color: rgb(255 0 0 / .33);
```

```
color: rgb(255 0 0 / 0);
```



quiz 1/2

Welke kleur is dit?

rgb(0 255 0);



quiz 1/2

Welke kleur is dit?

```
rgb(128 0 255 / .5);
```

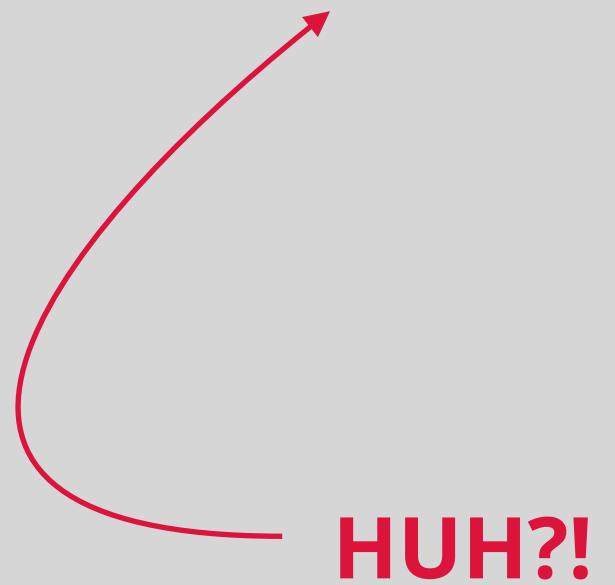


Kleurtjes 3a/5

RGB met HEX



#FF0099;



HUH?!



Kleurtjes 3b/5

RGB met HEX

rgb(255 0 153);
red green blue



#FF0099;
red green blue

3x een waarde
tussen 00 en FF.

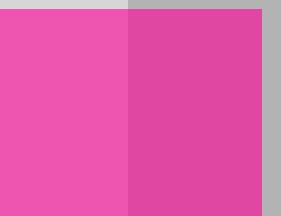
- 00 → 0
- 33 → 51
- 66 → 102
- 99 → 153
- CC → 204
- FF → 255



Kleurtjes 3c/5

RGB met HEX

rgb(255 0 153 / .6);
red green blue
alpha



#FF009999;
red green blue alpha

optionele waarde
tussen 00 en FF.
- 00 → 0
- 33 → 0.2
- 66 → 0.4
- 99 → 0.6
- CC → 0.8
- FF → 1



quiz 1/2

Welke kleur is dit?

#0000ff;



quiz 2/2

Welke kleur is dit?

#008f00;



Kleurtjes 4a/5

HSL



`hsl(270deg 100% 50%);`



degrees?!



Kleurtjes 4b/5

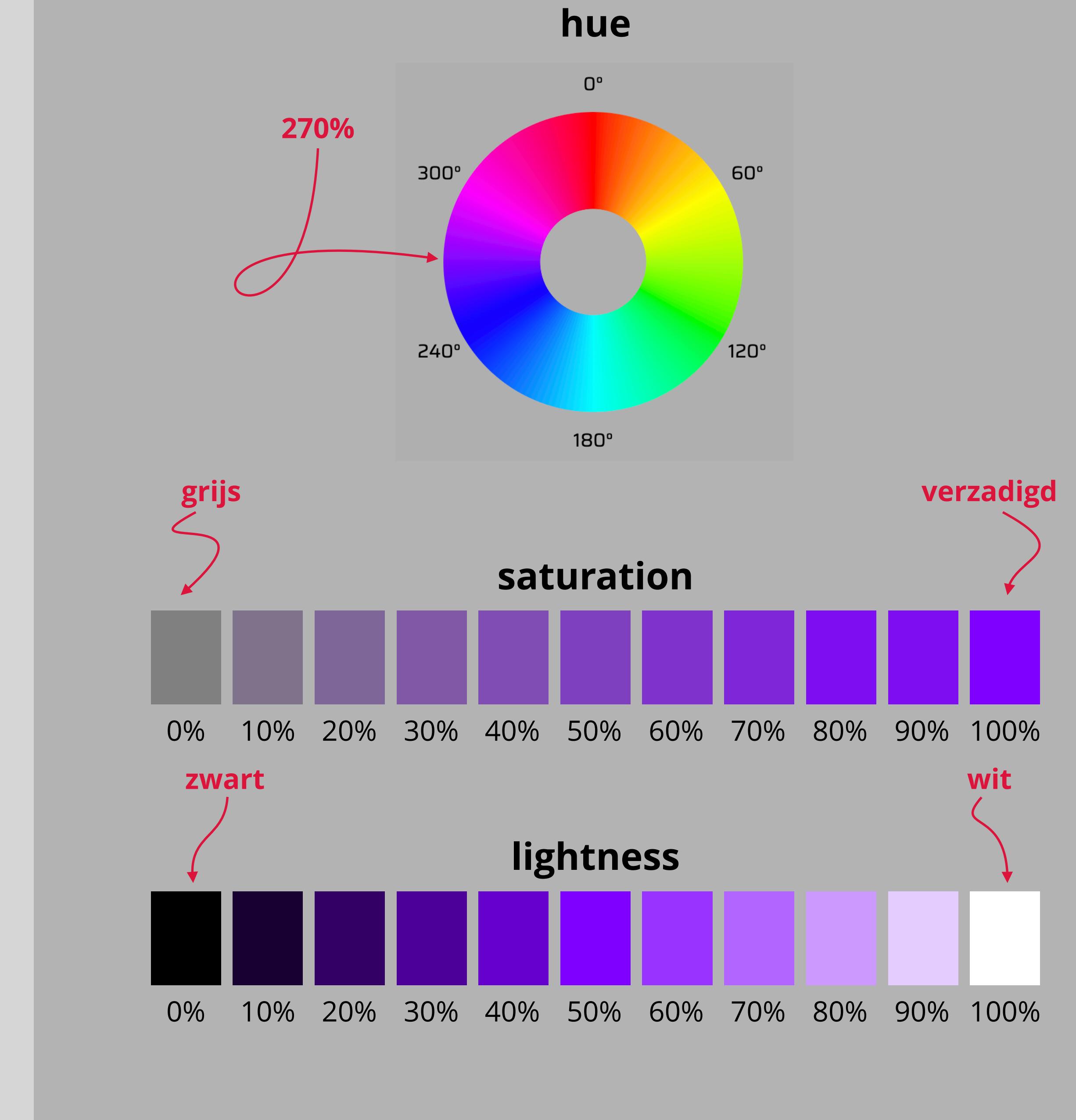
HSL



`hsl(270deg 100% 50%)` ;
_____ hue _____ saturation _____ lightness

bronnen:

smashingmagazine.com/2021/07/hsl-colors-css/
hslpicker.com





Kleurtjes 4c/5

HSL



`hsl(270deg 100% 50%) ;`



`hsl(270deg 100% 60%) ;`



`hsl(270deg 100% 50%) ;`



`hsl(270deg 100% 40%) ;`

bronnen:

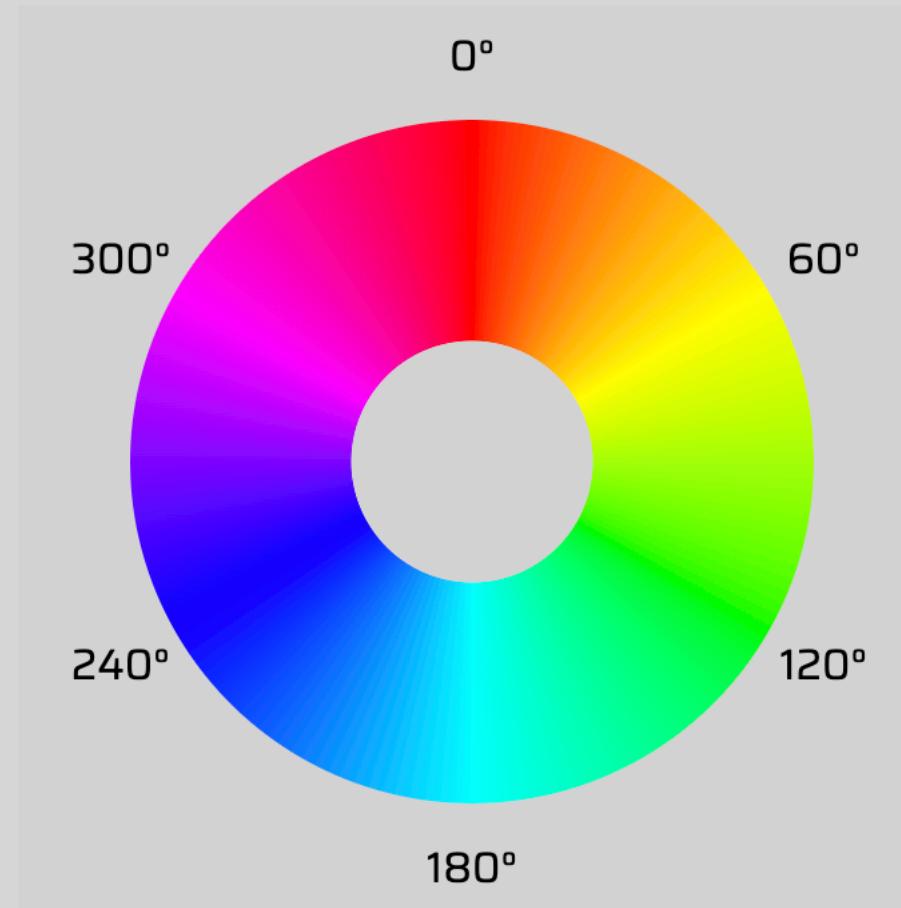
smashingmagazine.com/2021/07/hsl-colors-css/
hspicker.com

HSI is handig
voor tinten



quiz 1/2

Welke kleur is dit?

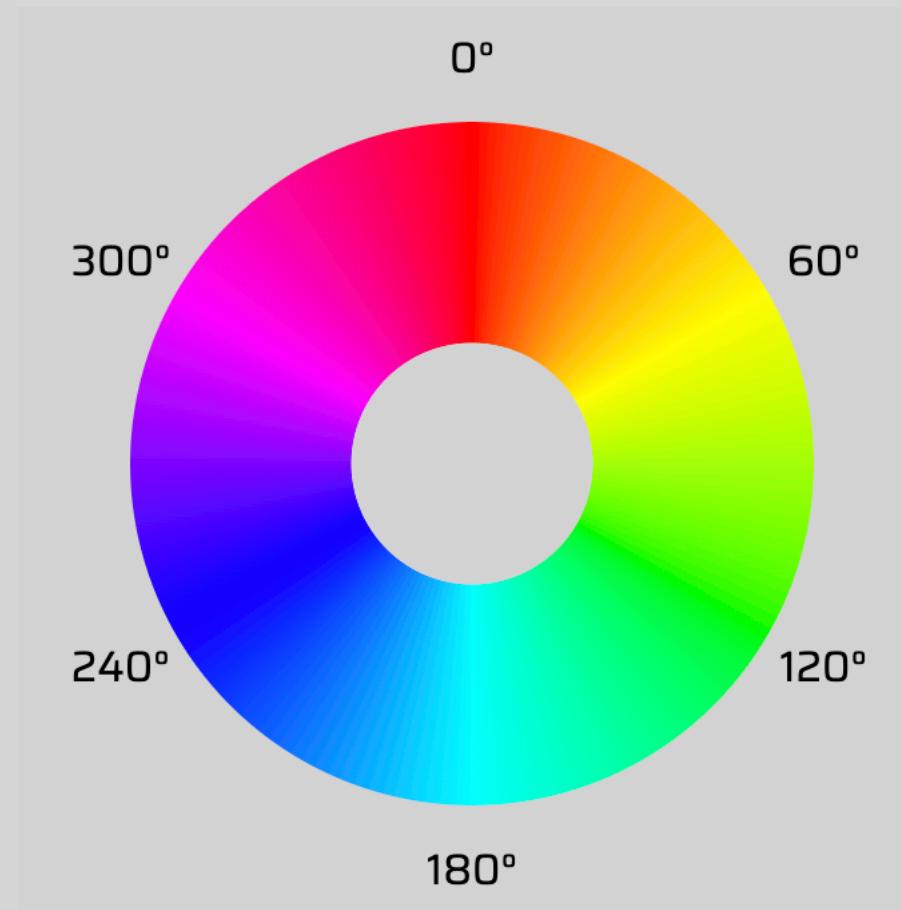


`hsl(90deg 100% 50%) ;`



quiz 2/2

Welke kleur is dit?



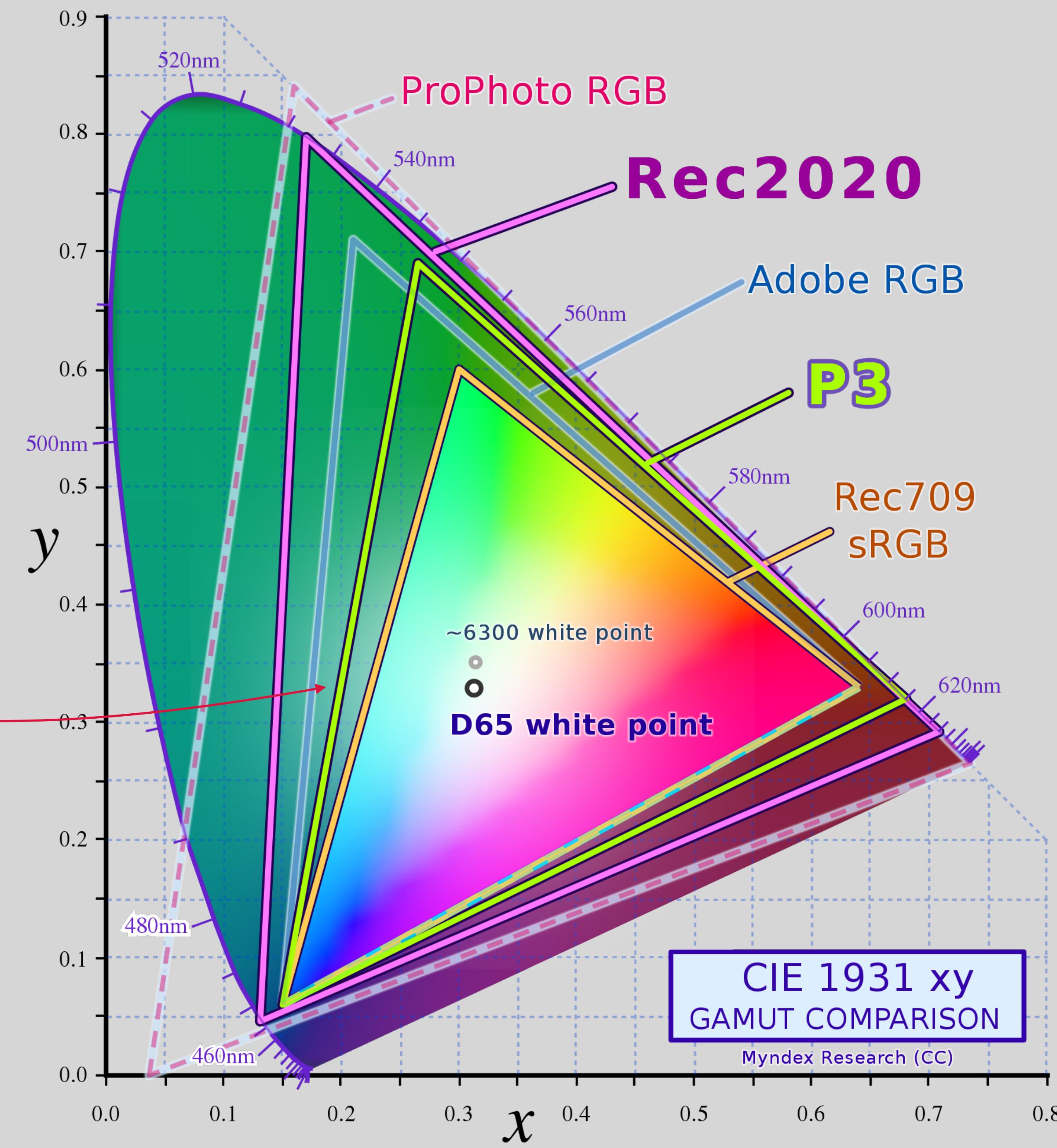
`hsl(90deg 50% 50%) ;`



Advanced

Kleurtjes 5a/5

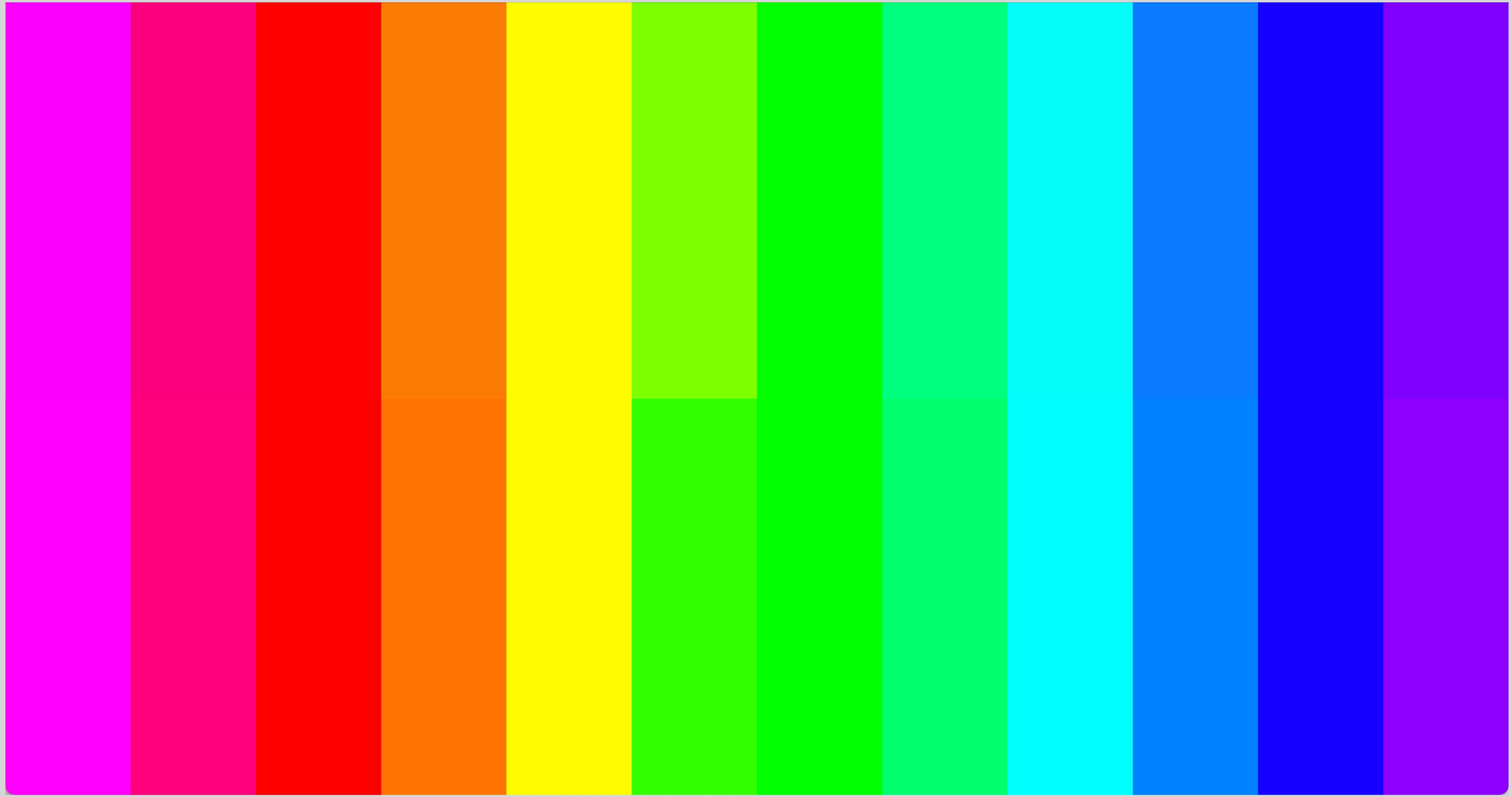
display-p3





rgb

display-p3



screenshot gemaakt in Safari



Advanced

Kleurtjes 5b/5

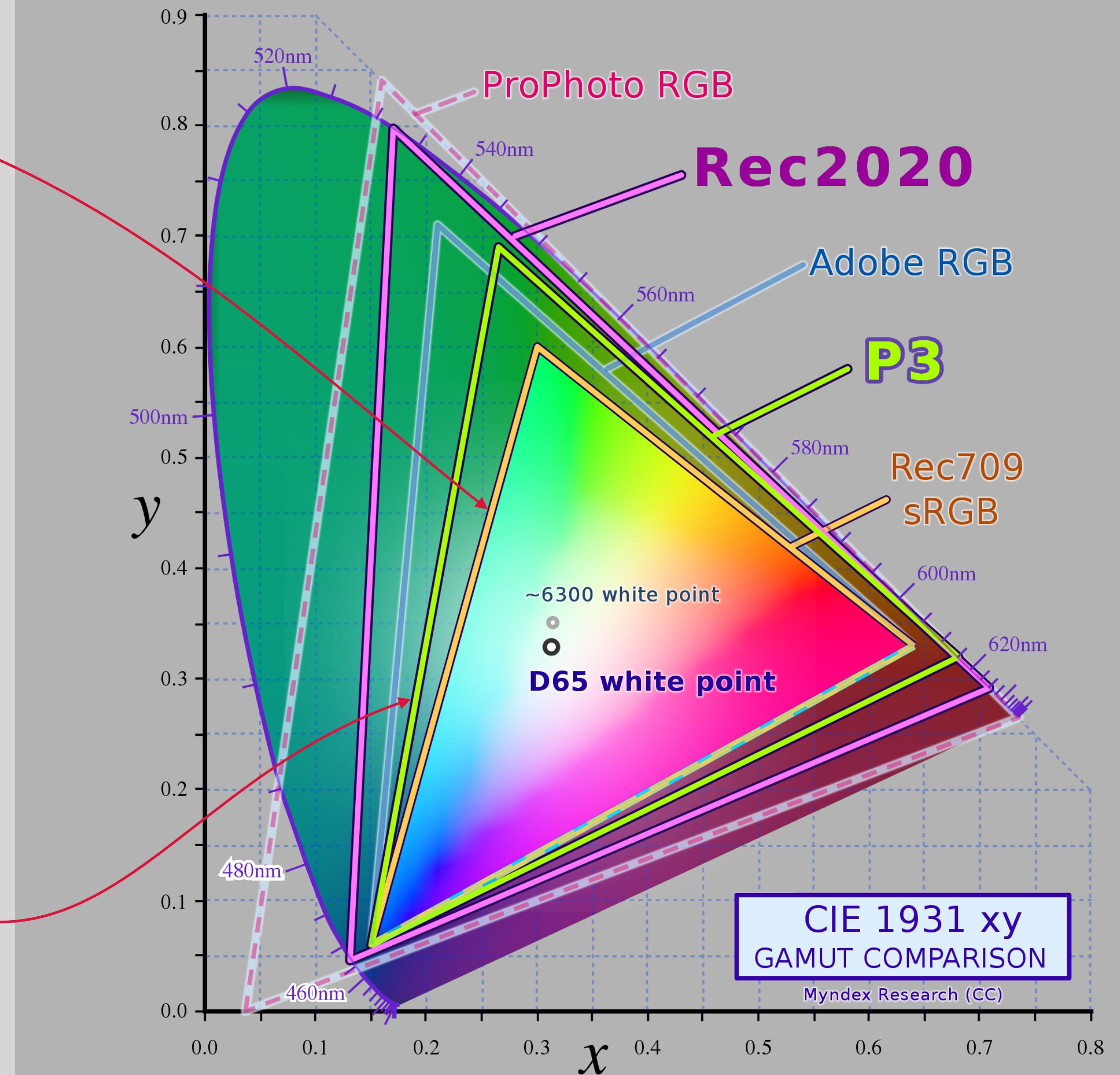
display-p3

Met het **RGB kleurprofiel** kun je niet alle kleuren definiëren. Dat was lang geen probleem omdat beeldschermen ook maar een beperkt aantal kleuren konden weergeven.

Moderne schermen kunnen meer (verzadigde) kleuren weergeven. Die kun je gebruiken door een ander kleurprofiel te kiezen. Bijv. **display-p3** - dat voor nu door Safari ondersteund wordt.

bron:

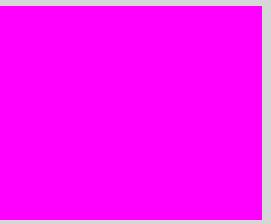
webkit.org/blog/10042/wide-gamut-color-in-css-with-display-p3/





Kleurtjes 5c/5

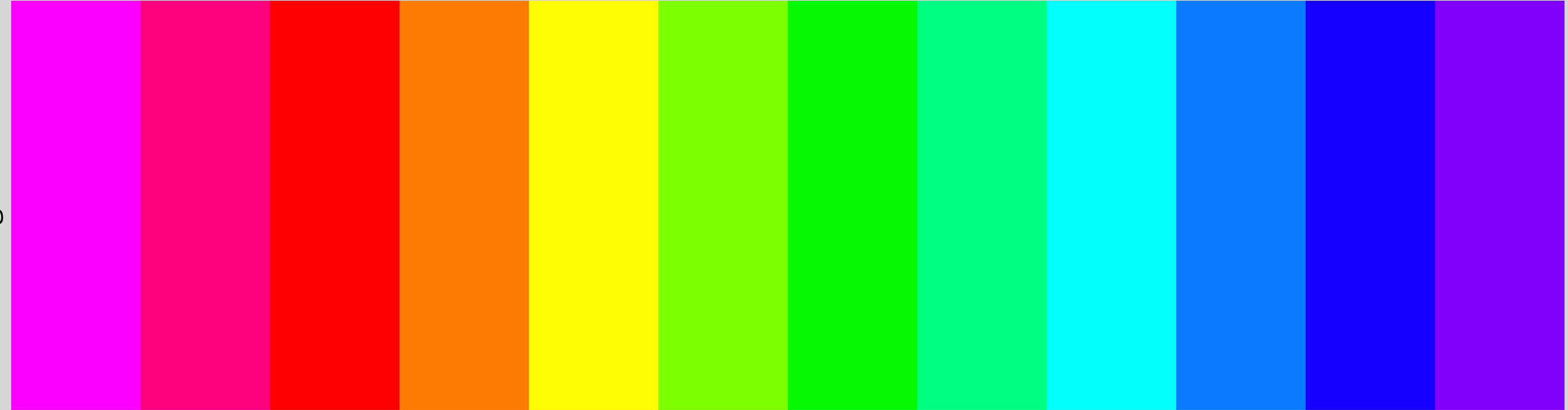
display-p3



color(display-p3 $\frac{1}{R}$ $\frac{0}{G}$ $\frac{1}{B}$);
color-profile



rgb



display-p3

Geen kleurtjes als de browser display-p3 niet ondersteund :-(

screenshot gemaakt in Chrome



Kleurtjes 5d/5

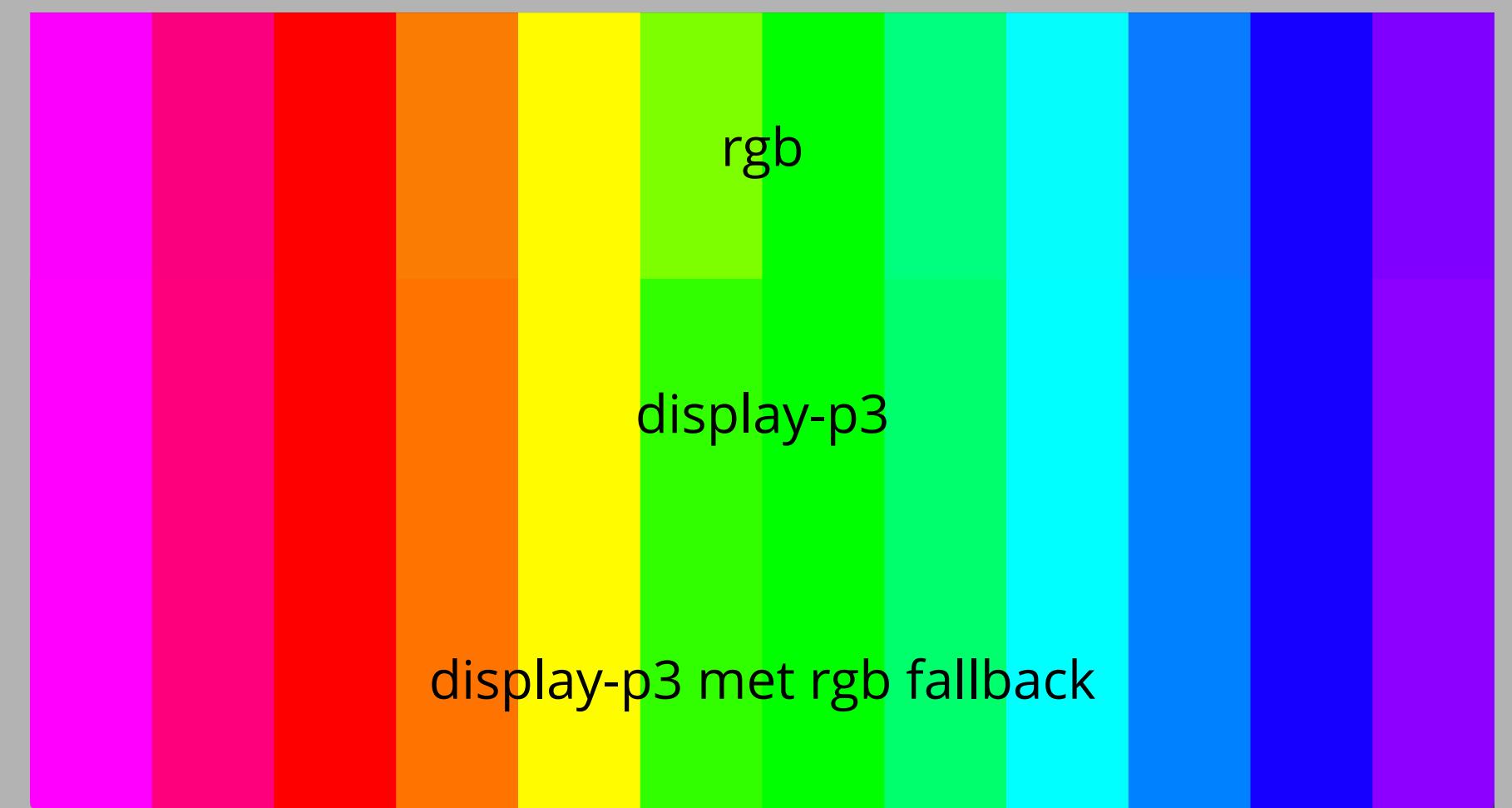
display-p3



```
rgb(255 0 255);  
color(display-p3 1 0 1);
```

De fallback voor als de
browser display-p3
niet ondersteund

Safari

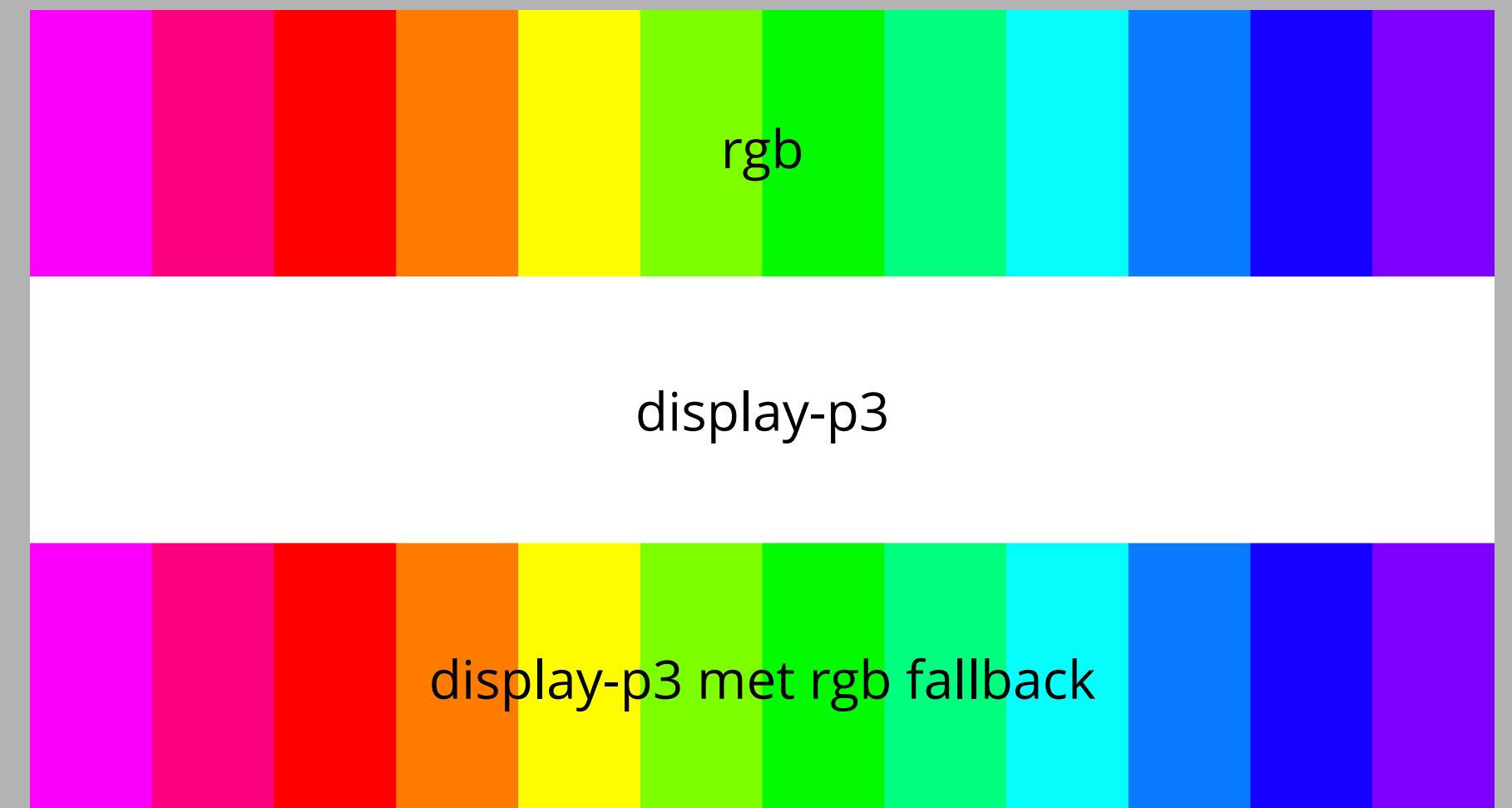


rgb wordt
getoond

display-p3
wordt
getoond

display-p3 met rgb fallback

Chrome



rgb wordt
getoond

display-p3

display-p3 met rgb fallback

display-p3
wordt **niet**
getoond

rgb fallback
wordt **wel**
getoond

Animeren - Recap oefening

Recap the animation properties 1/4

Om op stoom te komen 9 animatiesommetjes. Maak de 9 animaties na - van herhaling/goed te doen tot diep zwart.

De code voor jou:

codepen.io/shooft/pen/bGjJWwP

Uitwerking en voorbeeld:

codepen.io/shooft/pen/XWBQRKp



Animeren - Recap oefening

Recap the animation properties 2/4

Maak de 9 animaties na - van goed te doen tot diep diep zwart.

De code voor jou:

codepen.io/shooft/pen/bGjJWwP

Uitwerking en voorbeeld:

codepen.io/shooft/pen/XWBQRKp

Als de gebruiker over de section.lord hovert.

Draai de emoji 180deg in 1 seconde.

En blijft omgedraaid, zolang de gebruiker hovert.

De truc is om het hartje in 30% van de animatie 2x te laten kloppen met transform:scale(), om het daarna tot 100% dezelfde maat te houden.

Hierdoor ontstaat een pauze tussen de hartslagen. De duur is 2s.



Met transform-origin kun je bepalen om welk punt de palm ronddraait. Met transform-origin: left top; draait de palm bijvoorbeeld om de linkerbovenhoek.

Animeren - Recap oefening

Recap the animation properties 3/4

Maak de 9 animaties na - van goed te doen tot diep diep zwart.

De code voor jou:

codepen.io/shooft/pen/bGjJWwP

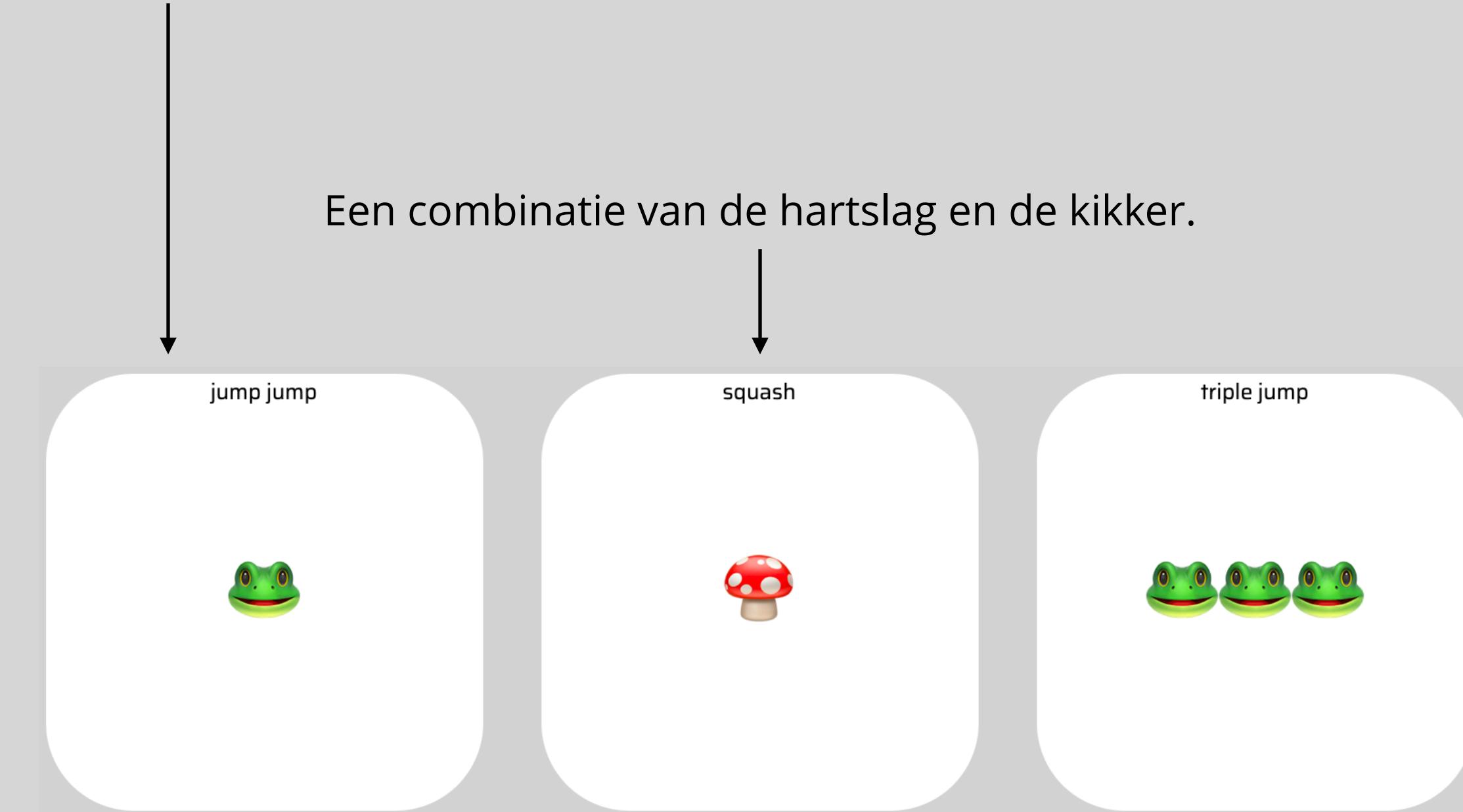
Uitwerking en voorbeeld:

codepen.io/shooft/pen/XWBQRKp

Om de versnelling goed te krijgen moet je aan de slag met cubic-bezier().

Met easing spelen: <https://matthewlein.com/tools/ceaser>

Met animation-direction:alternate kun je de animatie om-en-om heen en weer laten gaan.



Een combinatie van de hartslag en de kikker.

Met animation-delay kun je de animatie van een element later laten starten. Met div:nth-of-type(2) {animation-delay:1s;} start de tweede kikker bijvoorbeeld 1s later met springen.

Animeren - Recap oefening

Recap the animation properties 2/4

Maak de 9 animaties na - van goed te doen tot diep diep zwart.

De code voor jou:

codepen.io/shooft/pen/bGjJWwP

Uitwerking en voorbeeld:

codepen.io/shooft/pen/XWBQRKp

De 3 palmen staan boven op elkaar. De bovenste/derde start direct en is helemaal zichtbaar. De middelste/tweede start iets later en is half transparant. De onderste/eerste start nog wat later en is driekwart transparant.

Met rotateX() draait de palm om de x-as. Door ook nog perspective() toe te voegen creëer je een 3D-effect.



It's all yours. Tips:

- Het zijn meerder animaties die na elkaar starten
- Voor de pirouette is rotate3D gebruikt

ანიმელის ჩერხ
ორგანიზაცია

