

# Processing

## Voorwoord

Dit is het Processing boek van de Dojo. Processing is een programmeertaal. Dit boek leert je die programmeertaal.

## Over dit boek

Dit boek heeft een CC-BY-NC-SA licentie.

(C) Dojo Groningen 2016

Het is nog een beetje een slordig boek. Zo staat bijvoorbeeld het plaatje dat eigenlijk op de kaft moet staan op pagina twee. Er zitten tiepvauten in en de opmaak is **niet altijd *even mooi***.

Daarom staat dit boek op een GitHub. Om precies te zijn, op <https://github.com/richelbilderbeek/Dojo>. Hierdoor kan iedereen die dit boek te slordig vindt minder slordig maken.

## Point

Processing is een programmeertaal ontwikkeld voor kunstenaars en erg geschikt om games en mooie dingen mee te maken.

In deze les gaan we leren

- wat pixels zijn
- hoe de pixels op een beeldscherm zitten
- hoe je puntjes tekent

Zo gaat het eruit zien:

## Pixels

Pixels zijn de vierkantjes waaruit je beeldscherm is opgebouwd. Je schermresolutie is het aantal pixels dat je scherm breed en hoog is. Hoe meer pixels je scherm heeft, hoe scherper je beeldscherm eruit ziet

Vroeger hadden computers een lage resolutie. Daarom zien oude spellen er wat blokkiger uit:



Figure 1: Voorblad



Figure 2: Het logo van De Jonge Onderzoekers



Figure 3: Het logo van Codestarter



Figure 4: De licentie van dit boek

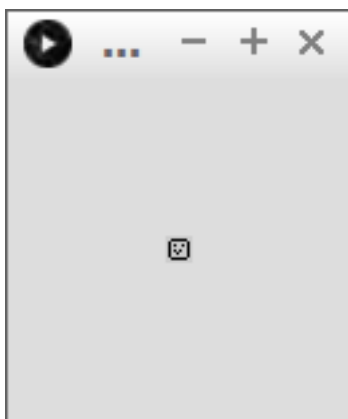


Figure 5: Point

## Vragen

- Wat is de resolutie van jouw beeldscherm?
- Als een beeldscherm een resolutie heeft van 320 bij 200 pixels, hoeveel pixels heeft deze dan?

## Hoe zitten pixels op je beeldscherm

Elke pixel op je beeldscherm heeft een soort postcode. Deze postcode noemen we een coördinaat. Een coördinaat bestaat uit twee getallen. Dit zijn de coördinaten van alle pixels van een beeldscherm van vijf bij vijf pixels:

(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)
(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)
(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

Figure 6: Pixel coördinaat

Je ziet dat de laagste coördinaat (0,0) (zeg: ‘nul komma nul’) is. (0,0) zit in de linkerbovenhoek van je beeldscherm. Dan zie je dat het eerste getal is hoeveel pixels naar recht hiervan is. Zo zit (1,0) rechts van (0,0). Het tweede getal is hoeveel pixels je onder (0,0) zit. Zo zit (0,1) onder (0,0).

Op deze manier kun je elke pixel op je beeldscherm vinden.

## Vragen

- Welke coördinaten heb je nodig om een horizontale lijn te tekenen, die door (0,0) gaat en drie pixels lang is?
- Welke coördinaten heb je nodig om een vertical lijn te tekenen, die door (1,0) gaat en drie pixels hoog is?
- Welke coördinaten heb je nodig om een diagonale lijn te tekenen, die door (0,0) gaat en drie pixels lang is?
- Welke coördinaten heb je nodig om een vierkantje te tekenen, die door (0,0) gaat, twee pixels breed en twee pixels breed is?

## Puntjes tekenen

Dit is de programmeercode die je nodig hebt:

```
void setup()
{
  size(10,10);
}

void draw()
{
  // 0123456789
  // 0 .....
  // 1 ..XXXXX..
  // 2 .X.....X.
  // 3 .X.X..X.X.
  // 4 .X.....X.
  // 5 .X.X.X..X.
  // 6 .X..X...X.
  // 7 .X.....X.
  // 8 ..XXXXX..
  // 9 .....

  // Bovenkant hoofd
  point(2,1);
  point(3,1);
  point(4,1);
  point(5,1);
  point(6,1);
  point(7,1);
```

```

// Rechterkant hoofd
point(8,2);
point(8,3);
point(8,4);
point(8,5);
point(8,6);
point(8,7);

// Onderkant hoofd
point(2,8);
point(3,8);
point(4,8);
point(5,8);
point(6,8);
point(7,8);

// Linkerkant hoofd
point(1,2);
point(1,3);
point(1,4);
point(1,5);
point(1,6);
point(1,7);

// Ogen
point(3,3);
point(6,3);

// Mond
point(3,5);
point(4,6);
point(5,5);
}

```

Dit is wat de code doet:

- `void setup() {}`: de setup functie, de computer voert een keer alles tussen de accolades uit
- `size(10, 10)`: maak een scherm van 10 pixels breed en 10 pixels hoog
- `;`: de puntkomma geeft in Processing het einde aan van een zin. Dit is ongeveer hetzelfde met een punt in schrijftaal.
- `void draw() {}`: de draw functie, de computer voert steeds alles tussen de accolades uit

- `///: commentaar: de rest van de regel wordt niet gelezen door de computer. Hier kun je dingen neerzetten die speciaal bedoelt zijn voor mensen, zoals jezelf. In dit programma zie je dat het poppetje is getekent in commentaar, met de coördinaten aan de zijkant. Ook zie je dat de programmeur zegt wat er getekend wordt: zo kun je de fouten sneller vinden`
- `point(2,1)`: teken een puntje op coördinaat (2,1).

## Vragen

- Start Processing. Kopieer deze code en run het programma
- Laat de smiley een pixel breder lachen
- Draai de lach verticaal om, zodat de smiley sip gaat kijken
- Geef de smiley punk haar door drie pixels bovenop het hoofd erbij te tekenen
- Geef de smiley een sik van een pixel

## Line

Zonder lijnen kun je bijna geen games maken. Een van de allereerste successgames, Asteroids, bestond vooral uit lijnen:

Je kunt een lijn tekenen met een boel puntjes, maar de `line` functie werkt gemakkelijker.

In deze les gaan we leren

- hoe je lijnen tekent

Zo gaat het eruit zien:

Kun je nog geen puntjes tekenen? Ga dan [naar de les waarin je puntjes leert tekenen](#)

## Lijnen

Een lijn bestaat uit pixels. Om een lijn te tekenen, moet je een beginpixel en eindpixel kiezen. Processing tekent dan zelf de pixels ertussenin.

Om in Processing een lijn te tekenen, gebruik je de functie `line`. De functie `line` heeft vier getallen nodig. Deze vier getallen zijn twee coördinaten achter elkaar. De eerste twee getallen zijn de coördinaat van het ene eind van de lijn. De laatste twee getallen zijn de coördinaat van het ander eind van de lijn.



Figure 7: Asteroids



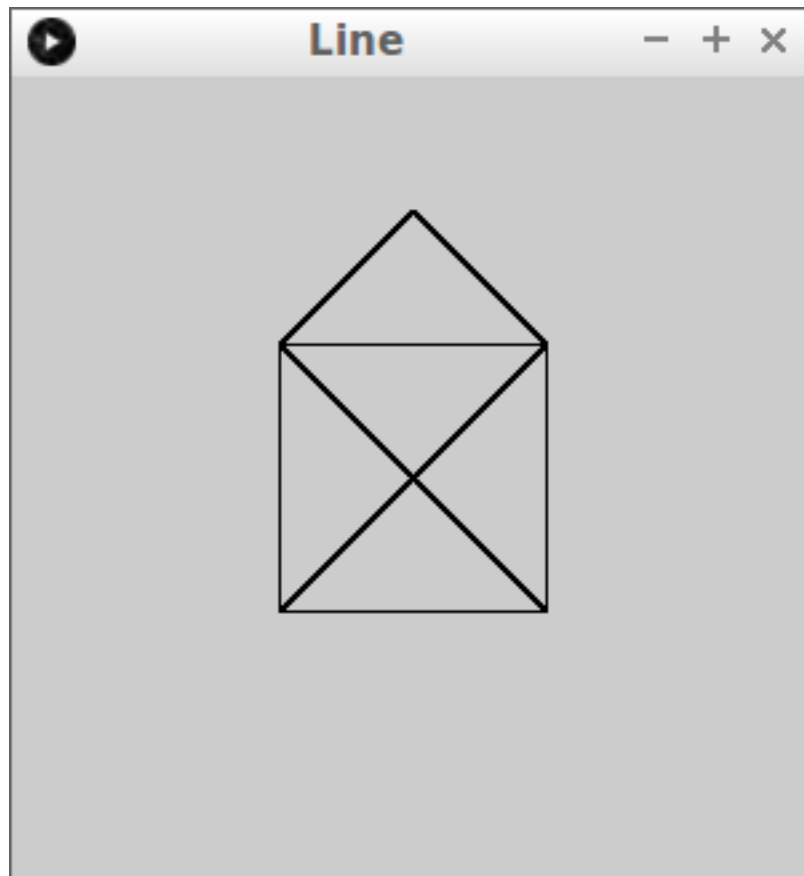


Figure 8: Line

(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)
(0, 2)				(4, 2)
(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

Figure 9: Lijn 1

Hier zie je een lijn die gaat van coördinaat (1,2) naar (3,2):

In Processing teken je deze lijn met:

```
line(1,2,3,2);
```

De volgorde van de twee coördinaten maakt niet uit.

Lijnen kunnen ook schuin gaan.

Hier zie je een lijn die gaat van coördinaat (2,4) naar (1,1):

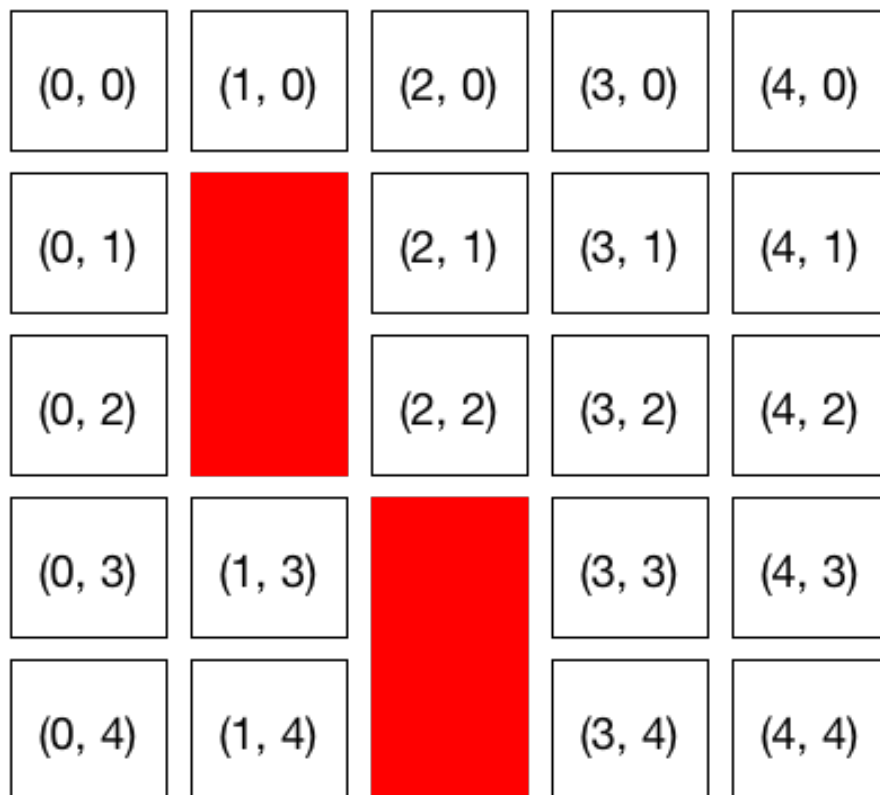


Figure 10: Lijn 2

In Processing teken je deze lijn met:

```
line(2,4,1,1);
```

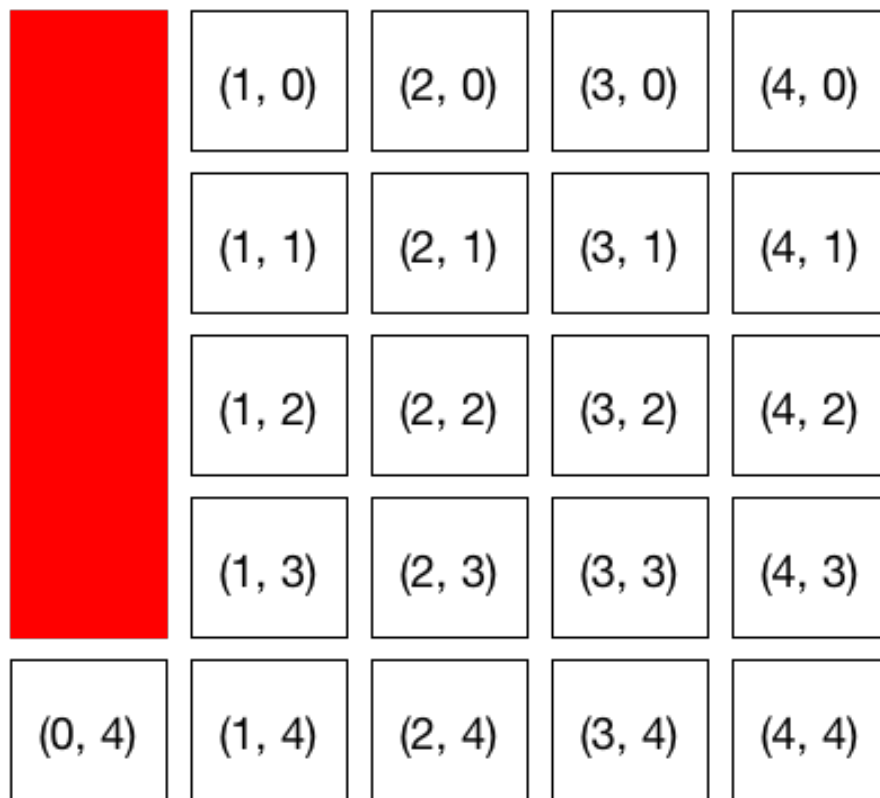


Figure 11: Lijn 3

## Vragen

- 1. Hierboven staat een lijn. Wat zijn de begin- en eindcoördinaat van die lijn?

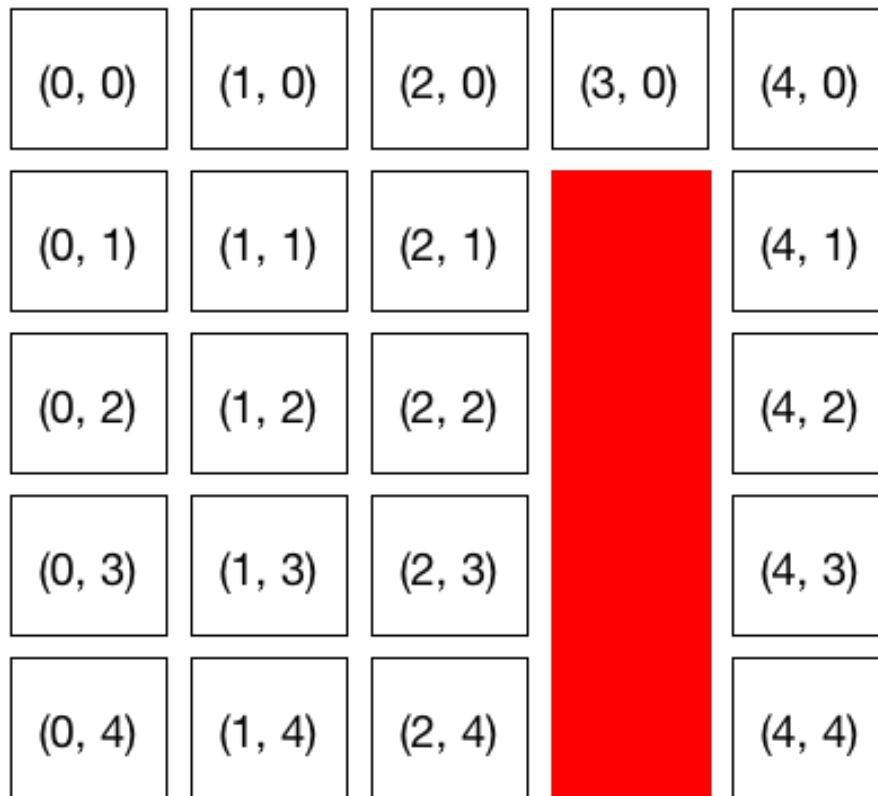


Figure 12: Lijn 4

- 2. Hierboven staat een lijn. Wat zijn de begin- en eindcoördinaat van die lijn?
- 3. Hierboven staat een lijn. Wat zijn de begin- en eindcoördinaat van die lijn?
- 4. Een lijn gaat van coördinaat (0,0) naar (10,0). In welke richting gaat de lijn? Wat is het Processing commando?
- 5. Een lijn gaat van coördinaat (0,0) naar (0,10). In welke richting gaat de lijn? Wat is het Processing commando?

(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)
(0, 1)	(1, 1)		(3, 1)	(4, 1)
(0, 2)		(2, 2)	(3, 2)	(4, 2)
	(1, 3)	(2, 3)	(3, 3)	(4, 3)
(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)

Figure 13: Lijn 5

- 6. Een lijn gaat van coördinaat (0,0) naar (10,10). In welke richting gaat de lijn? Wat is het Processing commando?
- 7. Een lijn gaat van coördinaat (30,20) naar (20,20). In welke richting gaat de lijn? Wat is het Processing commando?
- 8. Een lijn gaat van coördinaat (10,20) 20 pixels naar rechts. Welke coördinaat heeft het eindpunt? Wat is het Processing commando?
- 9. Een lijn gaat van coördinaat (10,30) 10 pixels naar rechtsomhoog. Welke coördinaat heeft het eindpunt? Wat is het Processing commando?

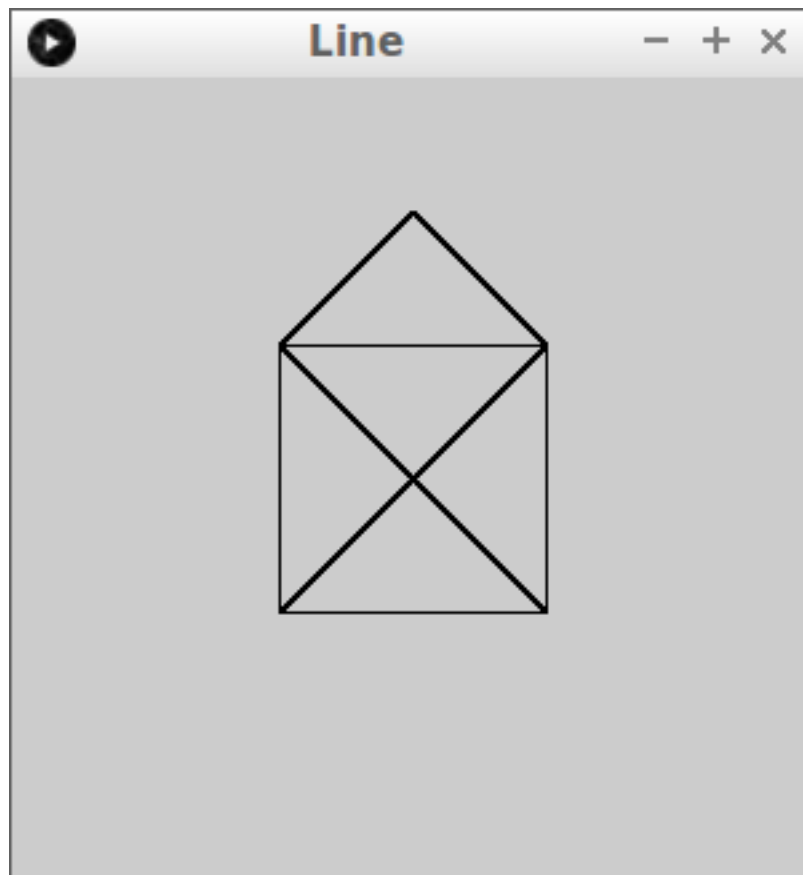


Figure 14: Line

- 10. Hierboven staat een tekening. Maak deze tekening na in Processing

## Oplossing

- 1. (0,0) en (0,3)
- 2. (3,1) en (3,4)
- 3. (2,1) en (0,3)
- 4. Van links naar rechts/horizontaal. `line(0,0,10,0)`
- 5. Van onder naar boven/verticaal. `line(0,0,0,10)`
- 6. Schuin/diagonaal. `line(0,0,0,10)`
- 7. Van rechts naar links/horizontaal. `line(30,20,20,20)`
- 8. (30,20). `line(10,20,30,20)`
- 9. (20,20). `line(10,30,20,20)`
- 10. Zie hieronder:

```
void setup()
{
  size(300,300);
}

void draw()
{
  //
  //      a
  //    / \
  //  e---b
  //  |\  /|
  //   |X|
  //  |/\|
  //  d---c
  //
  // a: (150, 50)
  // b: (200,100)
  // c: (200,200)
  // d: (100,200)
  // e: (100,100)

  //Van a naar b
  line(150,50,200,100);
  //Van b naar c
  line(200,100,200,200);
  //Van c naar d
```



```

line(200,200,100,200);
//Van d naar e
line(100,200,100,100);
//Van e naar a
line(100,100,150,50);
//Van b naar d
line(200,100,100,200);
//Van b naar e
line(200,100,100,100);
//Van c naar e
line(200,200,100,100);
}

```

## backGround

Zonder kleur zien games er minder mooi uit.

Een van de allereerste games met kleur heette Moria:

Om goed kleuren te kunnen gebruiken, moeten we leren hoe kleuren werken.

In deze les gaan we leren

- hoe kleuren werken

Zo gaat het eruit zien:

Je hoeft maar weinig te weten, behalve hoe je [een mooi programma maakt](#)

## Kleuren

Als je goed naar een beeldscherm kijkt, zie je dat elke pixel uit drie lampjes bestaat:

De lampjes hebben de kleuren rood, groen en blauw.

Omdat de lampjes zo klein zijn, ziet ons oog van afstand de menging van deze drie lampjes. Zo zien de lampjes rood en groen er samen uit als geel. Hier zie je hoe de kleuren mengen:

Om wit te krijgen, heb je alledrie de kleuren nodig.

## Vragen

- 1. Welke drie kleuren lampjes heeft een pixel?



Figure 15: Moria

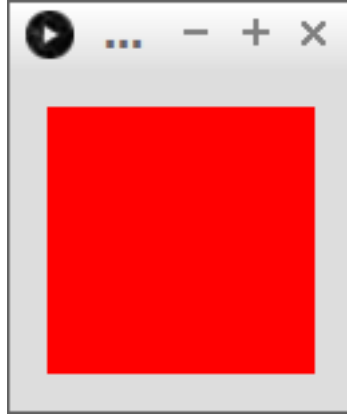


Figure 16: background

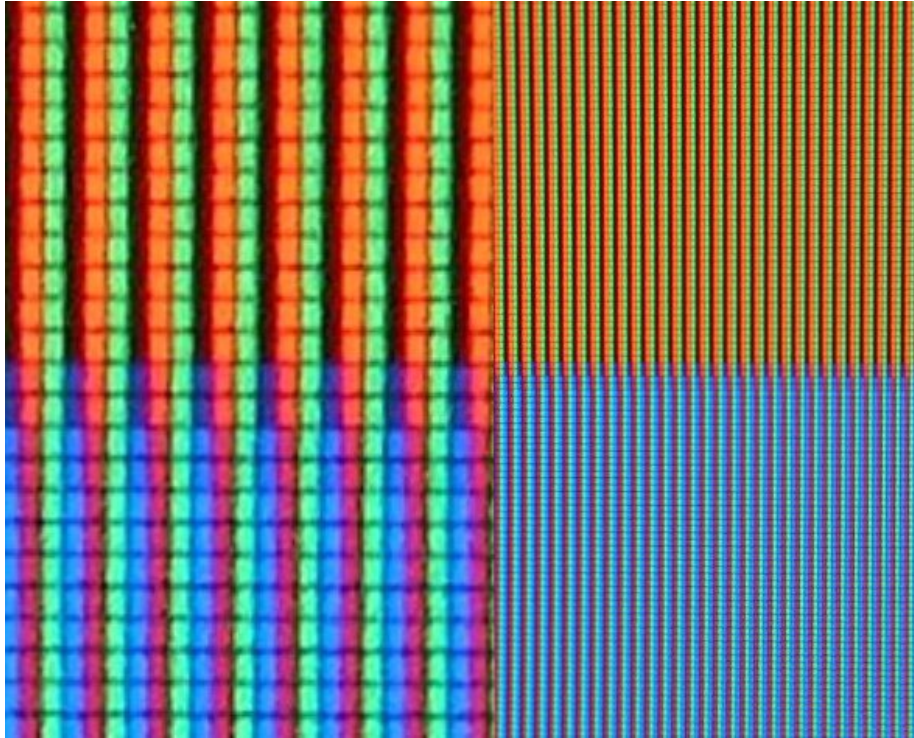


Figure 17: RGB pixels

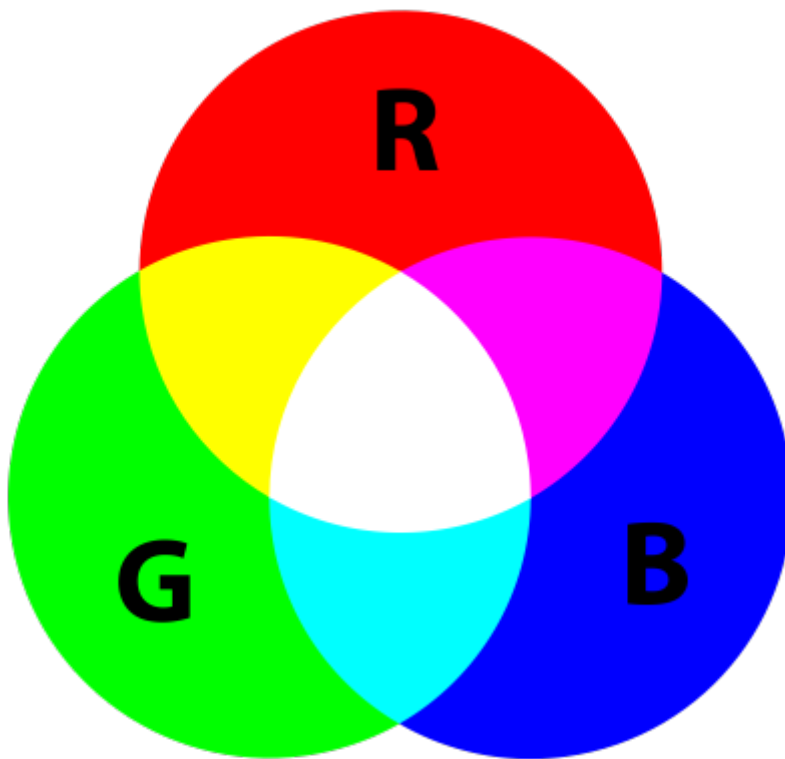


Figure 18: Additieve kleuren

- 2. Met welke lampjes samen maak je geel?
- 3. Met welke lampjes samen maak je cyaan/lichtblauw?
- 4. Met welke lampjes samen maak je magenta/paars?
- 5. Met welke lampjes samen maak je wit?
- 6. Met welke lampjes samen maak je zwart?
- 7. Met welke lampjes samen maak je grijs?
- 8. Met welke lampjes samen maak je oranje?

## Antwoorden

- 1. Rood, groen en blauw
- 2. Rood en groen
- 3. Groen en blauw
- 4. Rood en blauw
- 5. Rood en groen en blauw
- 6. Met geen lampjes: als alle lampjes uit zijn, is het zwart
- 7. Met rood en groen en blauw, maar dan moeten ze niet op hun hardst branden
- 8. Met rood op z'n hardst en groen op halve kracht

## background

In Processing is er een functie om de achtergrond een kleur te geven. Deze functie heet **background**. **background** is Engels voor 'achtergrond'. **background** is een functie die drie getallen nodig heeft. Deze drie getallen bepalen hoe hard de rode, groene en blauwe lampjes gaan schijnen. Deze drie getallen noemen we de RGB waarde. Met het getal nul zeg je dat een lampje uit staat. Met het getal 255 zeg je dat een lampje aan staat. Met getallen tussen nul en 255 kun je het lampje ertussenin laten branden.

Met deze Processing code krijg je een rode achtergrond:

```
void setup()
{
  size(100,100);
}
```

```
void draw()
{
  background(255,0,0);
}
```

## Opdracht

- 1. Kopieer deze code in Processing en start de code
- 2. Wat is een RGB waarde?
- 3. Wat is de RGB waarde van groen? Maak in Processing een groene achtergrond
- 4. Wat is de RGB waarde van blauw? Maak in Processing een blauwe achtergrond
- 5. Wat is de RGB waarde van geel? Maak in Processing een gele achtergrond
- 6. Wat is de RGB waarde van cyaan/lichtblauw? Maak in Processing een cyane/lichtblauwe achtergrond
- 7. Wat is de RGB waarde van magenta/paars? Maak in Processing een magenta/paarse achtergrond
- 8. Wat is de RGB waarde van wit? Maak in Processing een witte achtergrond
- 9. Wat is de RGB waarde van zwart? Maak in Processing een zwart achtergrond
- 10. Wat is de RGB waarde van grijs? Maak in Processing een grijze achtergrond
- 11. Wat is de RGB waarde van donkerrood? Maak in Processing een donkerrode achtergrond
- 12. Wat is de RGB waarde van oranje? Maak in Processing een oranje achtergrond

## Oplossingen

- 1. OK
- 2. De Rood-Groen-Blauw waarde. Dit zijn drie getallen van nul tot en met 255 die bepalen hoe hard de drie kleurenlampjes branden

- 3. `background(0,255,0)`
- 4. `background(0,0,255)`
- 5. `background(255,255,0)`
- 6. `background(0,255,255)`
- 7. `background(255,255,0)`
- 8. `background(255,255,255)`
- 9. `background(0,0,0)`
- 10. `background(128,128,128)`, maar andere getallen tussen 0 en 255 zijn ook goed. Als ze maar alledrie gelijk zijn
- 11. `background(128,0,0)`, maar het eerste getal mag ook een ander getal tussen de 0 en 255 zijn
- 12. `background(255,128,0)`, maar het tweede getal mag ook een ander getal in de buurt van 128 zijn

## stroke

Zonder kleur zien games er minder mooi uit.

Een van de allereerste games met kleur heette Moria:

In deze les gaan we leren

- hoe je een lijn een kleur kan geven.

Zo gaat het eruit zien:

Weet je nog niet hoe kleuren werken, ga dan naar de les [background](#)

## stroke

In Processing is er een functie om lijnen een kleur te geven. Deze functie heet **stroke**. **stroke** is Engels voor ‘(penseel)streek’. **stroke** is een functie die drie getallen nodig heeft. Deze drie getallen zijn de RGB waarden.

Met deze Processing code krijg je een rode lijn:



Figure 19: Moria



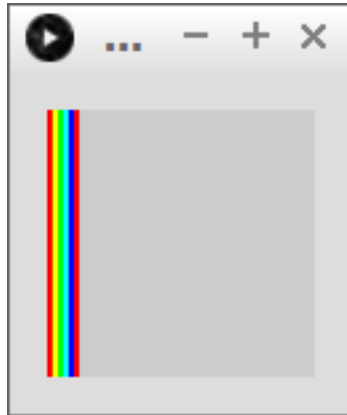


Figure 20: Stroke

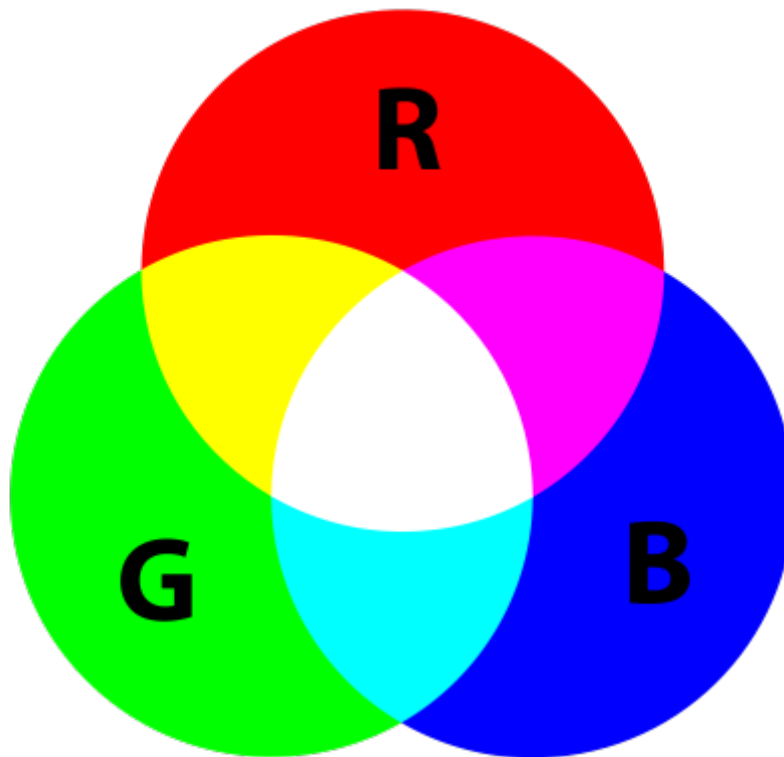


Figure 21: Kleurencirkel

```

void setup()
{
  size(100,100);
}

void draw()
{
  stroke(255,0,0);
  line(10,20,30,40);
}

```

Met **stroke** zeg je: ‘vanaf nu wil ik deze lijnkleur’. Hieronder zie je hoe je twee groene en een blauwe lijn tekent:

```

void setup()
{
  size(100,100);
}

void draw()
{
  stroke(0,255,0);
  line(10,20,30,40);
  line(50,60,70,80);
  stroke(0,0,255);
  line(90,10,20,30);
}

```

## Opdracht

- 1. Maak in Processing bovenstaande tekening na. Het venster is honderd bij honderd pixels. Elke kleur is met twee lijnen getekend. De kleuren zijn rood, geel, groen, cyaan, blauw, magenta

## Oplossing

```

void setup()
{
  size(100,100);
}

void draw()
{
  stroke(255,0,0);

```

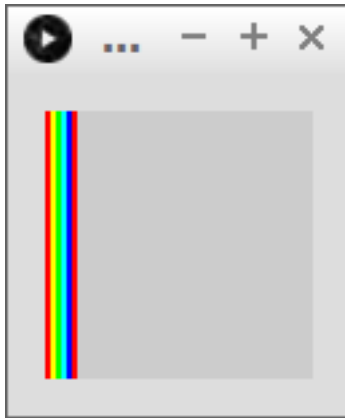


Figure 22: Stroke

```
line(0,0,0,100);  
line(1,0,1,100);  
stroke(255,255,0);  
line(2,0,2,100);  
line(3,0,3,100);  
stroke(0,255,0);  
line(4,0,4,100);  
line(5,0,5,100);  
stroke(0,255,255);  
line(6,0,6,100);  
line(7,0,7,100);  
stroke(0,0,255);  
line(8,0,8,100);  
line(9,0,9,100);  
stroke(255,0,0);  
line(10,0,10,100);  
line(11,0,11,100);  
}
```

## Rect

Vierkanten worden veel gebruikt in games.

Hier zie je een van de beroemdste games ooit:

Je kunt een vierkant tekenen met vier lijnen, maar de `rect` functie werkt gemakkelijker.

In deze les gaan we leren

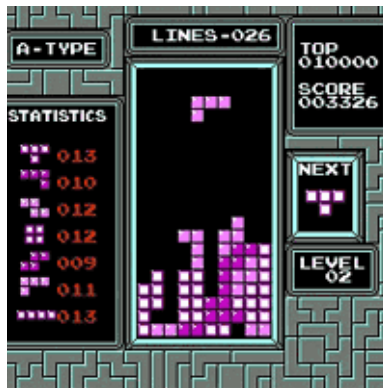


Figure 23: Tetris

- hoe je lijnen tekent

Zo gaat het eruit zien:

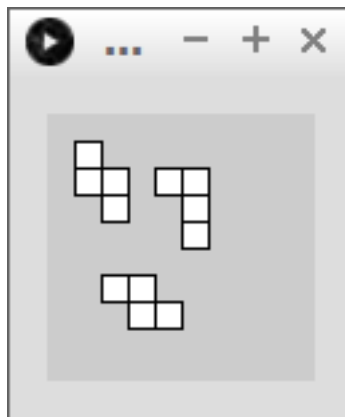


Figure 24: Rect

Kun je nog geen lijnen tekenen? Ga dan [naar de les waarin je lijnen leert tekenen](#)

## Rechthoeken

Een rechthoek bestaat uit vier lijnen. Om een rechthoek te tekenen, moet je een coördinaat, breedte en hoogte geven.

Om in Processing een rechthoek te tekenen, gebruik je de functie `rect`. De functie `rect` heeft vier getallen nodig. De eerste twee getallen zijn de coördinaat

van de linkerbovenhoek van de rechthoek. Het derde getal is de breedte van de rechthoek. Het vierde getal is de hoogte van de rechthoek.

Hier zie je een rechthoek met coördinaat (1,2), een breedte van drie pixels en hoogte van vier pixels:

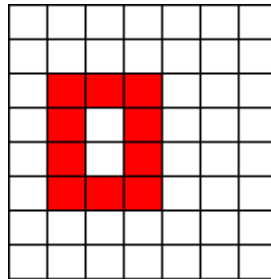


Figure 25: Rechthoek 1

In Processing teken je deze rechthoek met:

```
rect(1,2,3,4);
```

Hier is nog een rechthoek:

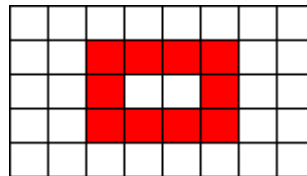


Figure 26: Rechthoek 2

De linkerbovenhoek heeft coördinaat (2,1), hij is vier pixels breed en drie pixels hoog.

## Vragen

- 1. Hierboven staat een rechthoek. Wat is de coördinaat van de linkerbovenhoek? Hoe breed is de rechthoek? Hoe hoog is de rechthoek?
- 2. Hierboven staat een rechthoek. Wat is de coördinaat van de linkerbovenhoek? Hoe breed is de rechthoek? Hoe hoog is de rechthoek?

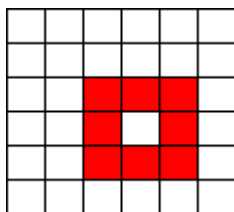


Figure 27: Rechthoek 3

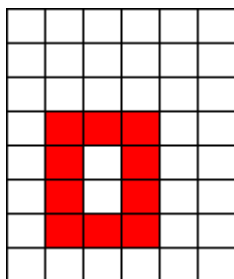


Figure 28: Rechthoek 4

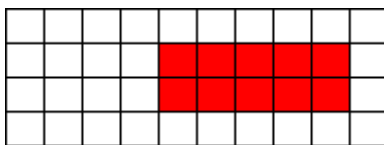


Figure 29: Rechthoek 5

- 3. Hierboven staat een rechthoek. Wat is de coördinaat van de linkerbovenhoek? Hoe breed is de rechthoek? Hoe hoog is de rechthoek?
- 4. Een rechthoek heeft als coördinaat (0,0), is twee pixels breed en drie pixels hoog. Wat is het Processing commando?
- 5. Een rechthoek heeft als coördinaat (1,2), is drie pixels breed en vier pixels hoog. Wat is het Processing commando?
- 6. Een rechthoek heeft als coördinaat (10,20), is dertig pixels breed en veertig pixels hoog. Wat is het Processing commando?

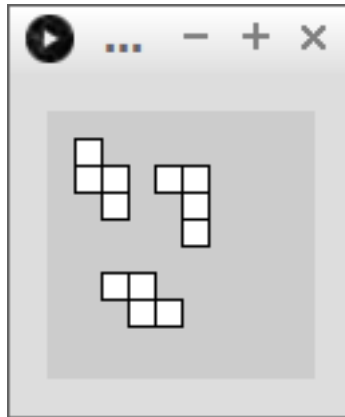


Figure 30: Rect

- 10. Hierboven staat een tekening. Maak deze tekening na in Processing

## Oplossing

- 10. Zie hieronder:

```
void setup()
{
  size(100,100);
}

void draw()
{
  rect(10,10,10,10);
  rect(10,20,10,10);
  rect(20,20,10,10);
}
```

```

rect(20,30,10,10);

rect(40,20,10,10);
rect(50,20,10,10);
rect(50,30,10,10);
rect(50,40,10,10);

rect(20,60,10,10);
rect(30,60,10,10);
rect(30,70,10,10);
rect(40,70,10,10);
}

```

## Ellipse

Cirkels en ovalen worden veel gebruikt in games.

Hier zie je een beroemde game, Bubble Bobble, dat veel met cirkels werkt:



Figure 31: Bubble Bobble

Je kunt een ovaal tekenen met heel veel puntjes, maar de `ellipse` functie werkt gemakkelijker.

In deze les gaan we leren



- hoe je ovalen tekent

Zo gaat het eruit zien:

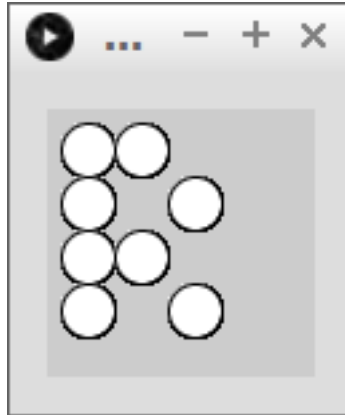


Figure 32: Ellipse

Kun je nog geen rechthoeken tekenen? Ga dan [naar de les waarin je rechthoeken leert tekenen](#)

## Ovalen

Een ovaal heeft een middelpunt, breedte en hoogte. Om een ovaal te tekenen, moet je een coördinaat, breedte en hoogte geven.

Om in Processing een ovaal te tekenen, gebruik je de functie `ellipse`. De functie `ellipse` heeft vier getallen nodig. De eerste twee getallen zijn de coördinaat van het midden van de ovaal. Het derde getal is de breedte van de ovaal. Het vierde getal is de hoogte van de ovaal.

Hier zie je een ovaal met middelpunt (3,2), een breedte van vijf pixels en hoogte van drie pixels:

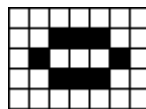


Figure 33: Ovaal 1

In Processing teken je deze ovaal met:

```
ellipse(3,2,5,3);
```

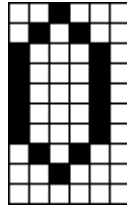


Figure 34: Ovaal 2

Hier is nog een ovaal:

Het middelpunt heeft coördinaat (2,4), hij is vijf pixels breed en negen pixels hoog.

## Vragen

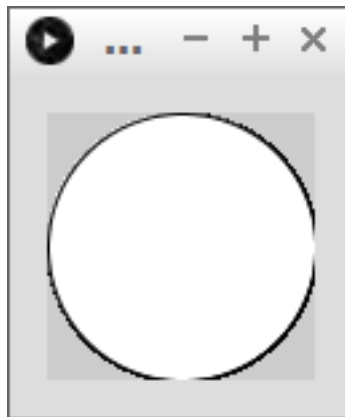


Figure 35: Ovaal 3

- 1. Je wilt bovenstaande plaatje namaken. Het venster is 100 pixels breed en 100 pixels hoog. Wat is het middelpunt van de cirkel? Hoe breed is de cirkel? En hoe hoog? Hoe maak je dit in Processing?
- 2. Je wilt bovenstaande plaatje namaken. Het venster is 200 pixels breed en 100 pixels hoog. Wat is het middelpunt van de cirkel? Hoe breed is de cirkel? En hoe hoog? Hoe maak je dit in Processing?
- 3. Je wilt bovenstaande plaatje namaken. Het venster is 200 pixels breed en 100 pixels hoog. Wat zijn de middelpunten van de cirkels? Hoe breed zijn de cirkels? En hoe hoog? Hoe maak je dit in Processing?



Figure 36: Ovaal 4

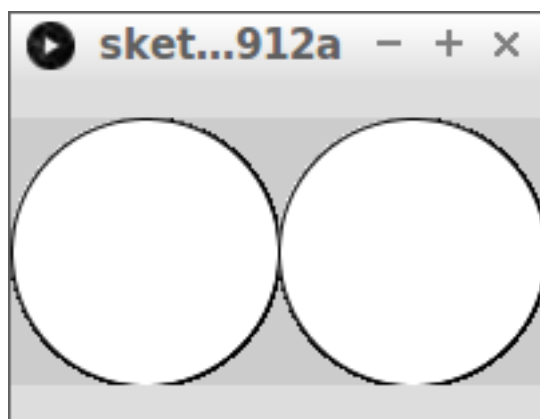


Figure 37: Ovaal 5

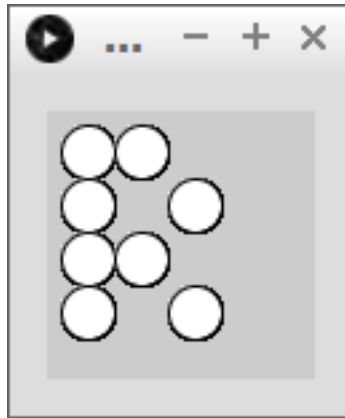


Figure 38: Ellipse

- 4. Hierboven staat een tekening. Maak deze tekening na in Processing

## Oplossing

- 1. Zie hieronder:

```
void setup() {
  size(100, 100);
}

void draw() {
  ellipse(50, 50, 100, 100);
}
```

- 2. Zie hieronder:

```
void setup() {
  size(200, 100);
}

void draw() {
  ellipse(100, 50, 200, 100);
}
```

- 3. Zie hieronder:

```
void setup() {
```

```

    size(200, 100);
}

void draw() {
    ellipse( 50, 50, 100, 100);
    ellipse(150, 50, 100, 100);
}

```

- 4. Zie hieronder:

```

void setup()
{
    size(100, 100);
}

void draw()
{
    ellipse(15,15, 20, 20);
    ellipse(15,35, 20, 20);
    ellipse(15,55, 20, 20);
    ellipse(15,75, 20, 20);

    ellipse(35, 15, 20, 20);
    ellipse(55, 35, 20, 20);
    ellipse(35, 55, 20, 20);
    ellipse(55, 75, 20, 20);
}

```

## fill

Een rechthoek of een ovaal kan ook ingekleurd worden.

Een van de beroemdste games ooit heeft bijvoorbeeld veel ingekleurde vierkanten:

In deze les gaan we leren

- hoe we de invulkleur van vierkanten en ovalen instellen

Zo gaat het eruit zien:

Weet je nog niet hoe je een lijn een kleur kan geven? Ga dan naar [stroke](#)

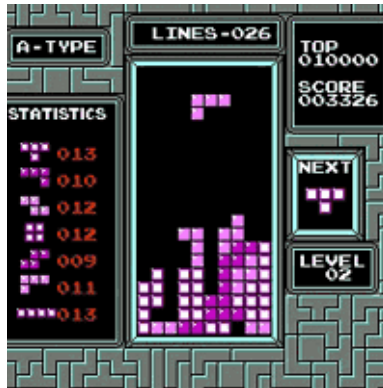


Figure 39: Tetris

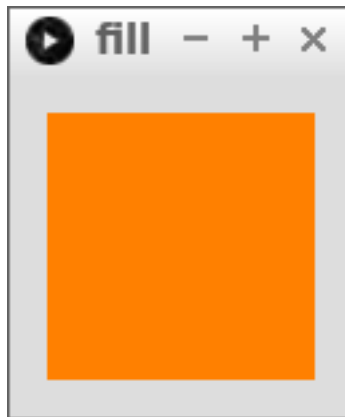


Figure 40: Fill

## Kleuren

De lampjes hebben de kleuren rood, groen en blauw.

Omdat de lampjes zo klein zijn, ziet ons oog van afstand de menging van deze drie lampjes. Zo zien de lampjes rood en groen er samen uit als geel. Hier zie je hoe de kleuren mengen:

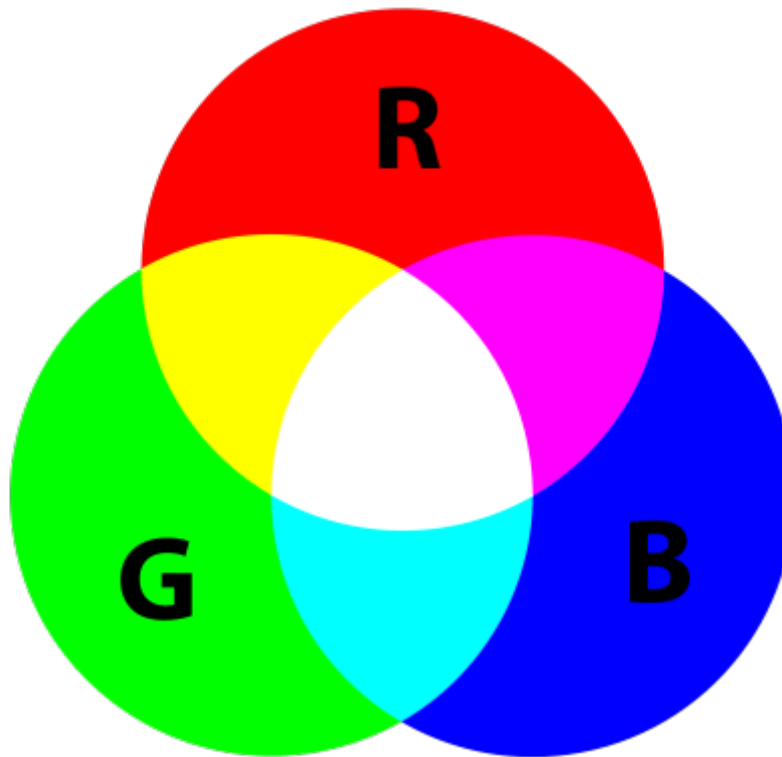


Figure 41: Additieve kleuren

Om wit te krijgen, heb je alledrie de kleuren nodig.

## Vragen

- 1. Welke drie kleuren lampjes heeft een pixel?
- 2. Met welke lampjes samen maak je geel?
- 3. Met welke lampjes samen maak je cyaan/lichtblauw?
- 4. Met welke lampjes samen maak je magenta/paars?

- 5. Met welke lampjes samen maak je wit?
- 6. Met welke lampjes samen maak je zwart?
- 7. Met welke lampjes samen maak je grijs?
- 8. Met welke lampjes samen maak je oranje?

## Antwoorden

- 1. Rood, groen en blauw
- 2. Rood en groen
- 3. Groen en blauw
- 4. Rood en blauw
- 5. Rood en groen en blauw
- 6. Met geen lampjes: als alle lampjes uit zijn, is het zwart
- 7. Met rood en groen en blauw, maar dan moeten ze niet op hun hardst branden
- 8. Met rood op z'n hardst en groen op halve kracht

## background

In Processing is er een functie om de achtergrond een kleur te geven. Deze functie heet **background**. **background** is Engels voor 'achtergrond'. **background** is een functie die drie getallen nodig heeft. Deze drie getallen bepalen hoe hard de rode, groene en blauwe lampjes gaan schijnen. Deze drie getallen noemen we de RGB waarde. Met het getal nul zeg je dat een lampje uit staat. Met het getal 255 zeg je dat een lampje aan staat. Met getallen tussen nul en 255 kun je het lampje ertussenin laten branden.

Met deze Processing code krijg je een rode achtergrond:

```
void setup()
{
  size(100,100);
}

void draw()
{
  background(255,0,0);
}
```



## Opdracht

- 1. Kopieer deze code in Processing en start de code
- 2. Wat is een RGB waarde?
- 3. Wat is de RGB waarde van groen? Maak in Processing een groene achtergrond
- 4. Wat is de RGB waarde van blauw? Maak in Processing een blauwe achtergrond
- 5. Wat is de RGB waarde van geel? Maak in Processing een gele achtergrond
- 6. Wat is de RGB waarde van cyaan/lichtblauw? Maak in Processing een cyane/lichtblauwe achtergrond
- 7. Wat is de RGB waarde van magenta/paars? Maak in Processing een magenta/paarse achtergrond
- 8. Wat is de RGB waarde van wit? Maak in Processing een witte achtergrond
- 9. Wat is de RGB waarde van zwart? Maak in Processing een zwart achtergrond
- 10. Wat is de RGB waarde van grijs? Maak in Processing een grijze achtergrond
- 11. Wat is de RGB waarde van donkerrood? Maak in Processing een donkerrode achtergrond
- 12. Wat is de RGB waarde van oranje? Maak in Processing een oranje achtergrond

## Oplossingen

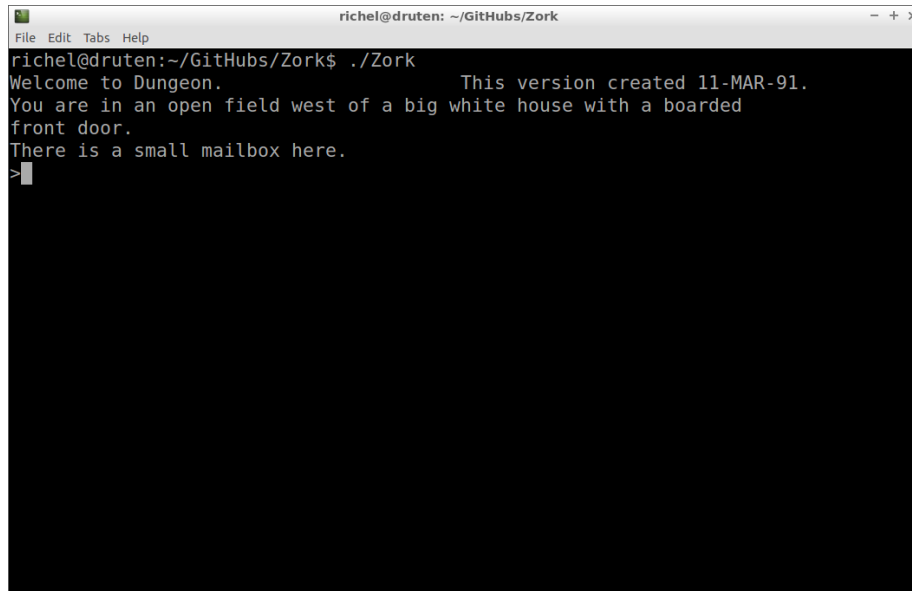
- 1. OK
- 2. De Rood-Groen-Blauw waarde. Dit zijn drie getallen van nul tot en met 255 die bepalen hoe hard de drie kleurenlampjes branden
- 3. `background(0,255,0)`
- 4. `background(0,0,255)`
- 5. `background(255,255,0)`
- 6. `background(0,255,255)`

- 7. `background(255,255,0)`
- 8. `background(255,255,255)`
- 9. `background(0,0,0)`
- 10. `background(128,128,128)`, maar andere getallen tussen 0 en 255 zijn ook goed. Als ze maar alledrie gelijk zijn
- 11. `background(128,0,0)`, maar het eerste getal mag ook een ander getal tussen de 0 en 255 zijn
- 12. `background(255,128,0)`, maar het tweede getal mag ook een ander getal in de buurt van 128 zijn

## text

Tekst wordt veel gebruikt, ook in games, voor bijvoorbeeld een score.

Hier zie je 'Zork, the underground empire', een van de beroemdste tekstavonturen ooit:



```

richel@druten: ~/GitHubs/Zork
File Edit Tabs Help
richel@druten:~/GitHubs/Zork$ ./Zork
Welcome to Dungeon.                This version created 11-MAR-91.
You are in an open field west of a big white house with a boarded
front door.
There is a small mailbox here.
>

```

Figure 42: Zork

In deze les gaan we leren

- hoe je tekst op het scherm zet

- hoe je berekeningen op het scherm zet
- hoe je tekst vergroot
- hoe je tekst een kleur geeft

Zo gaat het eruit zien:

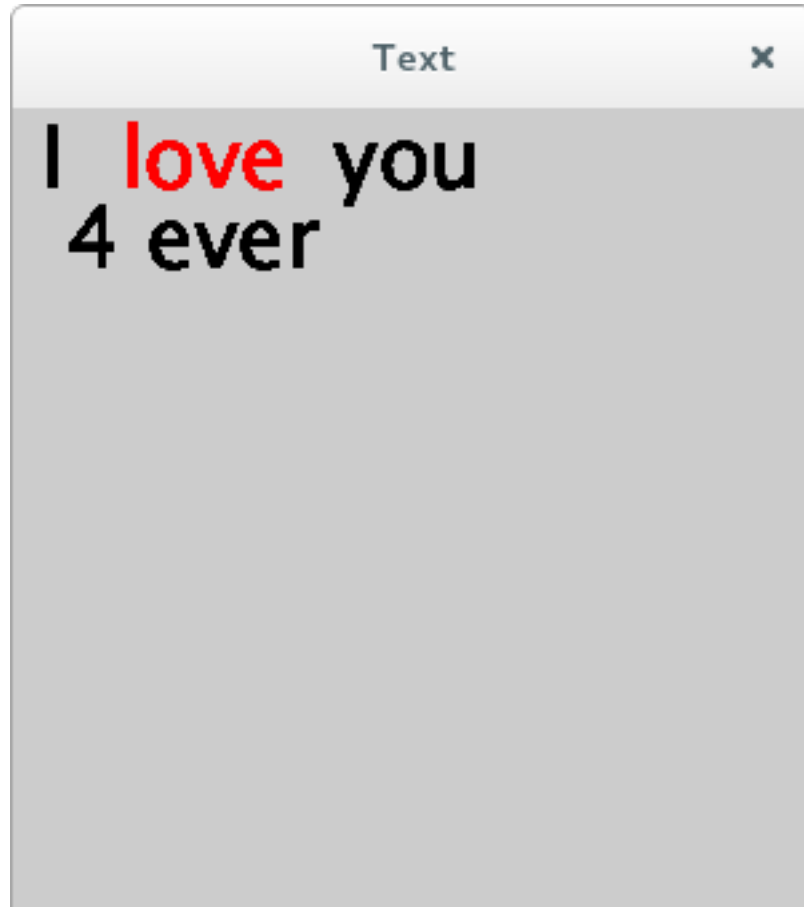


Figure 43: Text

Kun je nog geen puntjes tekenen? Ga dan [naar de les waarin je puntjes leert tekenen](#)

Kun je nog geen vlakken inkleuren? Ga dan [naar de les 'fill'](#)

## Tekst

Hier zie je de tekst 'Hallo' laat zetten op coördinaat (10,20):

```
text("Hallo", 10, 20);
```

Let op dat de tekst tussen dubbele apostroffen (") moet.

`text` kan ook rekenen!

Hier plus en min:

```
text(128 + 64, 10, 20);  
text(128 - 64, 10, 20);
```

Hier een keersom:

```
text(16 * 16, 10, 20);
```

Hier een deelsom:

```
text(256 / 16, 10, 20);
```

Tekstgrootte kun je aanpassen met

```
textSize(32);
```

Tekstkleur kun je aanpassen met `fill`:

```
fill(255, 0, 0);
```

## Opdracht

Zet de tekst `I love you 4 ever` op het scherm, waarbij:

- alle woorden zwart zijn, behalve `love`, die rood is
- de `4` is de uitkomst van een berekening, bijvoorbeeld `2 + 2` (maar hoe moeilijker de berekening, hoe stoerder)

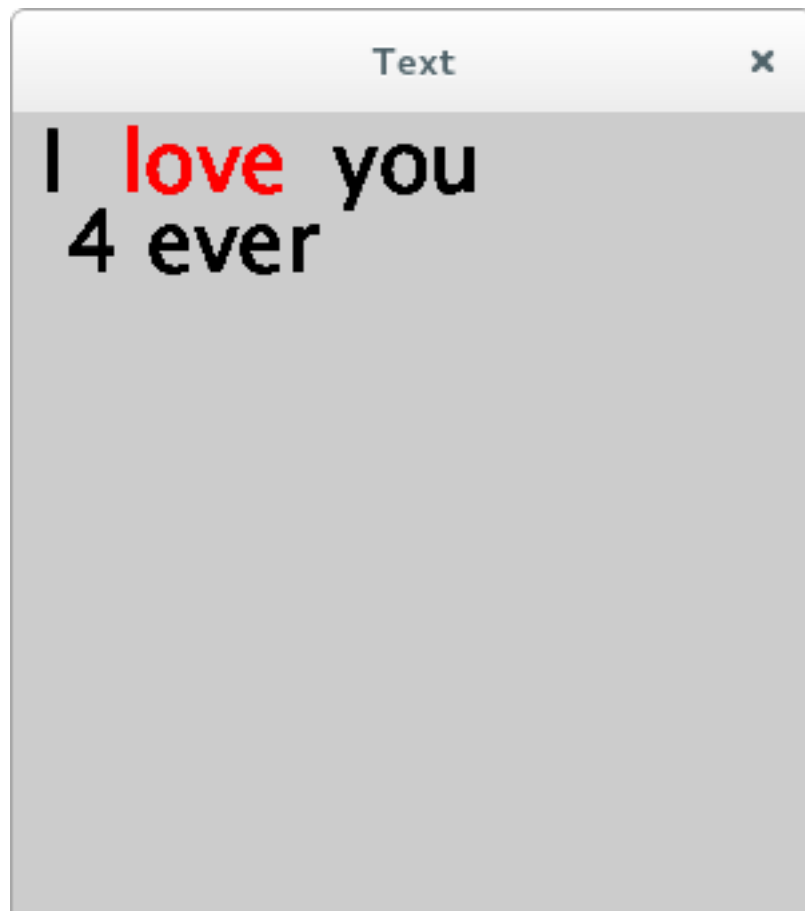


Figure 44: Text