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 licensie.

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 ni`et altij`d *even moo*i.

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.

Bal naar rechts

In deze les gaan we een bal naar rechts laten bewegen.

Het ziet er zo uit:

We leren in deze les wat een variabele is. Je kunt (bijna) niet programmeren zonder variabelen.

Wat weten we al?

Als je de vorige lessen hebt gedaan, weet je wat deze code doet:

```
void setup()
{
  size(600, 400);
}

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

Figure 1: Voorblad

Figure 2: Het logo van De Jonge Onderzoekers

Figure 3: Het logo van Codestarter

Figure 4: De licensie van dit boek

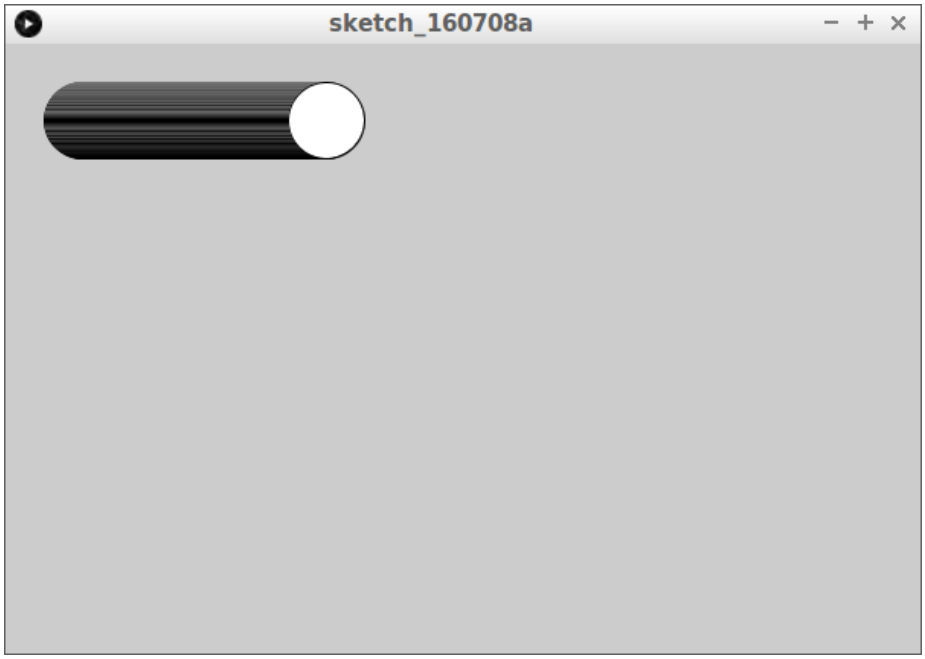


Figure 5: Bal naar rechts

Vragen

- Wat doet dit programma?
- Waar wordt deze cirkel getekend? Komt deze cirkel tegen de rand aan?

We gaan de code aanpassen:

```
float x = 50;

void setup()
{
  size(600, 400);
}

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

Vragen

- Wat doet dit programma?
- Wat zijn de verschillen?

De eerste nieuwe regel is:

```
float x = 50;
```

In mensentaal is dit: 'Lieve computer, onthoud het getal x. x heeft een beginwaarde van vijftig.'.

Een variabele is iets dat onthouden moet worden. Een kassa onthoudt bijvoorbeeld de hoeveelheid geld die alle boodschappen bij elkaar zijn. Variabelen die jij weet, zijn: je naam, je leeftijd, je geboortedatum, je adres, je telefoonnummer, je emailadres, en nog veel meer. Als iemand je je leeftijd vraagt, dan weet je welk getal je moet zeggen.

Het woord `x` is de naam van een variable. In dit geval van hoe ver de cirkel naar rechts staat. Het woord `float` betekent dat 'x' een getal is. Het symbool `=` betekent 'wordt vanaf nu'. Het getal `50` is de beginwaarde.

De tweede veranderde regel is:

```
ellipse(x,50,50,50);
```

In mensentaal is dit: 'Lieve computer, teken een ovaal die:

- `x` naar rechts is. De computer weet nog wel wat `x` is: vijftig!
- `50` omlaag is
- `50` pixels breed is
- `50` pixels hoog is

Vragen

- Wat als je `float x = 50;` weghaalt?
- Wat als je `float x = 50;` verandert naar `float rechts = 50;`?
- Wat als je `float x = 50;` verandert naar `float x = 100;`?
- Wat als je `ellipse(x,50,50,50);` weghaalt?
- Wat als je `ellipse(x,50,50,50);` verandert naar `ellipse(rechts,50,50,50);`?
- Wat als je `ellipse(x,50,50,50);` verandert naar `ellipse(x,x,50,50);`?
- Wat als je `ellipse(x,50,50,50);` verandert naar `ellipse(x,x,x,50);`?
- Wat als je `ellipse(x,50,50,50);` verandert naar `ellipse(x,x,x,x);`?
- Wat als je `x` vervangt door `rechts`?
- Wat als je `x` vervangt door `dinosaurus`?

Nu gaan we de cirkel laten bewegen:

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,50,50,50);
  x = x + 1;
}
```

Vragen

- Wat doet dit programma?
- Wat zijn de verschillen?

De nieuwe regel is:

```
x = x + 1;
```

In mensentaal is dit: 'Lieve computer, x is vanaf nu x plus een'. Of: 'Maak x een hoger'.

Vragen

- Als `x` vijftig is, wat is `x` dan na `x = x + 1;`?
- Als `x` 51 is, wat is `x` dan na `x = x + 1;`?
- Als `x` 52 is, wat is `x` dan na `x = x + 1;`?
- Als `x` 53 is, wat is `x` dan na `x = x + 1;`?
- Als `x` 54 is, wat is `x` dan na `x = x + 1;`?

Nu kunnen we snappen wat het programma doet. Hier staat het programma weer:

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,50,50,50);
  x = x + 1;
}
```

De eerste keer dat de computer `draw` gaat doen, dan vult deze voor `x` een `50` in. Daarna wordt x een hoger. Dan is `draw` klaar.

De tweede keer dat de computer `draw` gaat doen, dan vult deze voor `x` een `51` in. Daarna wordt x een hoger. Dan is `draw` klaar.

De derde keer dat de computer `draw` gaat doen, dan vult deze voor `x` een `52` in. Daarna wordt x een hoger. Dan is `draw` klaar.

Vragen

- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,50,50,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,x,50,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,50,x,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,50,50,x);`?

- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,x,50,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,x,x,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,50,x,x);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,50,50,x);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,x,x,50);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(50,x,x,x);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,50,x,x);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,x,50,x);`?
- Wat als je `ellipse(x,50,50,50);` vervangt door `ellipse(x,x,x,x);`?
- Wat als je `x = x + 1;` vervangt door `x = x + 2;`?
- Wat als je `x = x + 1;` vervangt door `x = x + 10;`?
- Wat als je `x = x + 1;` vervangt door `x = x + 0;`?
- Wat als je `x = x + 1;` vervangt door `x = x - 1;`?
- Wat als je `x = x + 1;` vervangt door `x = x - 0;`?
- Wat als je `x = x + 1;` vervangt door `x = x * 2;`?
- Wat als je `x = x + 1;` vervangt door `x = x * 1;`?
- Wat als je `x = x + 1;` vervangt door `x = x * 0;`?
- Wat als je `x = x + 1;` vervangt door `x = x / 2;`?
- Wat als je `x = x + 1;` vervangt door `x = x / -2;`?
- Wat als je `x = x + 1;` vervangt door `x = x / 1;`?
- Wat als je `x = x + 1;` vervangt door `x = x / 0;`?

Verder

Je zou nu kunnen doen:

- Bal die eeuwig naar rechts gaat

Bal die eeuwig naar rechts gaat

In deze les gaan we een bal eeuwig naar rechts laten gaan.

Het ziet er zo uit:

We leren in deze les wat `if`-statement is. Je kunt (bijna) niet programmeren zonder `if`-statements.

Wat weten we al?

Als je de vorige lessen hebt gedaan, weet je wat deze code doet:

Figure 6: Bal die eeuwig naar rechts gaat 1

Figure 7: Bal die eeuwig naar rechts gaat 2

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,50,50,50);
  x = x + 1;
}
```

Vragen

- Wat doet dit programma?
- In welke richting beweegt de ovaal
- Blijft de ovaal zichtbaar op het scherm?
- Kopieer de code en bekijk het programma. Klopt wat je dacht?

Een `if`-statement

We willen kunnen zeggen: 'Lieve computer, *als* de bal te ver naar rechts is, dan teleporteer je de bal naar rechts'. `if` is Engels voor 'als'.

Zo zou dit kunnen:

```
if (x > 200)
{
  x = 100;
}
```

Dit betekent:

- `if`: begin van een if statement. Een if-statement heeft dan twee gedeeltes:
- `()`: tussen de ronde haken staat een test; iets wat waar of niet waar is
- `{}`: tussen de accolades staat wat de computer moet doen als de test waar is
- `x > 200`: dit staat tussen de ronde haken. Dit is de test 'x is groter dan 200'. Het `>` tekentje betekent 'groter dan'
- `x = 100`: dit staat tussen de accolades. Als 'x is groter dan 200' waar is, dan krijgt `x` de waarde 100

Preciezer zeg je: 'Lieve computer, *als* x meer is dan 200, zet x dat op 100'. `if` is Engels voor 'als'.

Vragen

- Kopieer het `if`-statement tussen de accolades van de `draw` functie
- Wat doet het programma?

Als het kopieren niet is gelukt, gebruik dan deze code:

```
float x = 50;

void setup()
{
  size(600, 400);
}

void draw()
{
  ellipse(x,100,100,100);
```

```
  x = x + 1;
  if (x > 200)
  {
    x = 100;
  }
}
```

- Kun je ervoor zorgen dat de ovaal helemaal naar de linkerkant van het scherm springt?
- Kun je ervoor zorgen dat de ovaal helemaal naar rechst beweegt, voordat deze naar de linkerkant van het scherm springt?

Antwoord

Dit is een eeuwig naar rechts gaande bal:

```
float x = -50;

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + 1;
  if (x > 650)
  {
    x = -50;
  }
}
```

Het lijkt al een beetje op Lonelier Pong. Dit is geen toeval :-)

Verder

Je zou nu kunnen doen:

- Bal die horizontaal stuitert

Bal die horizontaal stuitert

In deze les gaan we een bal horizontaal laten stuiteren.

Het ziet er zo uit:

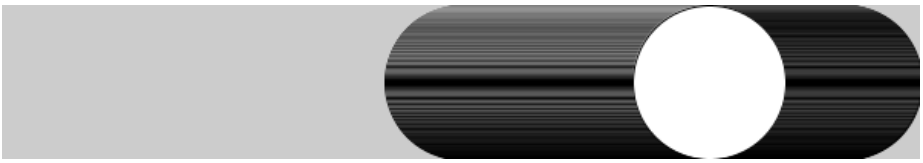


Figure 8: Bal die horizontaal stuitert (zie 'dojo/Images/BalDieHorizontaalStuitertGif')

We gaan in deze les twee variabelen en twee `if`-statements gebruiken.

Wat weten we al?

Dit is een eeuwig naar rechts gaande bal:

```
float x = 300;

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + 1;
  if (x > 650)
  {
    x = -50;
  }
}
```

Vragen

- Wat doet dit programma?
- In welke richting beweegt de ovaal?
- Blijft de ovaal zichtbaar op het scherm?
- Kopieer de code en bekijk het programma. Klopt wat je dacht?
- Verander het programma: laat de bal nu eeuwig naar links gaan

Twee variabelen

Nu onthoudt de computer een variabele: de x-coordinaat van de ovaal. Om een bal te laten stuiteren, moet er ook een richting onthouden worden: de bal gaat immers of naar links of naar rechts.

In deze code wordt de richting van de bal `dx` genoemd. Dit is een afkorting van 'delta x' en dat is weer wiskundetaal voor 'de verandering van x'.

Vragen

```
float x = 300;
float dx = 2;

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + dx;
  if (x > 650)
  {
    x = -50;
  }
}
```

- In welke richting beweegt de bal?
- Hoeveel pixels beweegt de bal per keer?
- Zet de waarde van `dx` op `1`. Wat zie je?
- Zet de waarde van `dx` op `0`. Wat zie je?
- Zet de waarde van `dx` op `-1`. Wat zie je?
- Bij sommige waarden van `dx` gaat de bal links het beeld uit. Maak een tweede if-statement, die ervoor zorgt dat de bal eeuwig links kan gaan. Test dit bij een `dx` van `2`, `0` en `-2`.
- Wat moet er met de `dx` gebeuren om de bal te laten stuiteren? Probeer dit!

Stuiteren

Als een bal met een snelheid van drie pixels naar rechst gaat en stuitert, dan gaat deze vanaf dan drie pixels naar links. Andersom is dat ook zo.

Hier is een manier om de bal te laten stuiteren:

```
float x = 300;
float dx = 2;

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + dx;
  if (x > 650)
  {
    dx = -2;
  }
  if (x < 50)
  {
    dx = 2;
  }
}
```

Vragen

- Kopieer en run de code. Stuitert de bal?
- Verander de snelheid van de bal naar een pixel per keer. Op hoeveel plekken moet je de code aanpassen?
- Verander de snelheid van de bal naar drie pixels per keer. Op hoeveel plekken moet je de code aanpassen?
- Verander de snelheid van de bal terug naar een pixel per keer. Op hoeveel plekken moet je de code aanpassen? Laat nu de bal in het begin naar links gaan. Op hoeveel plekken moet je de code aanpassen?

De richting omklappen

Er is een slimmere manier om `dx` te veranderen. We hebben gezien dat als `dx` gelijk was aan `2`, deze `-2` wordt bij bij een stuiter. We hebben gezien dat als `dx` gelijk was aan `-2`, deze `2` wordt bij bij een stuiter. Er komt een minnetje voor, of er komt een minnetje bij. Dit is gemakkelijk op dezelfde manier te doen:

```
dx = -dx;
```

Hiermee zeg je 'Lieve computer, de nieuwe waarde van `dx` is min de oude waarde'. Als de oude waarde van `dx 2` is, dan wordt deze nu `-2`. Als de oude waarde van `dx -2` is, dan wordt deze nu `--2` (jawel, min min twee) en dat mag je schrijven als `2` (omdat min keer min is plus).

Opdracht

- Gebruik nu de slimme manier om een bal te laten stuiteren.

Het lijkt al een beetje op Lonelier Pong. Dit is geen toeval :-)

Zwaartekracht

In deze les gaan we zwaartekracht programmeren.

Het ziet er zo uit:

We gaan in deze les twee variabelen en twee `if`-statements gebruiken.

Wat weten we al?

Dit is een eeuwig horizontaal stuiterende bal:

```
float x = 300;
float dx = 1; //Snelheid in de x richting

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + dx;
  if (x > 550 || x < 50)
  {
    dx = -dx;
  }
}
```

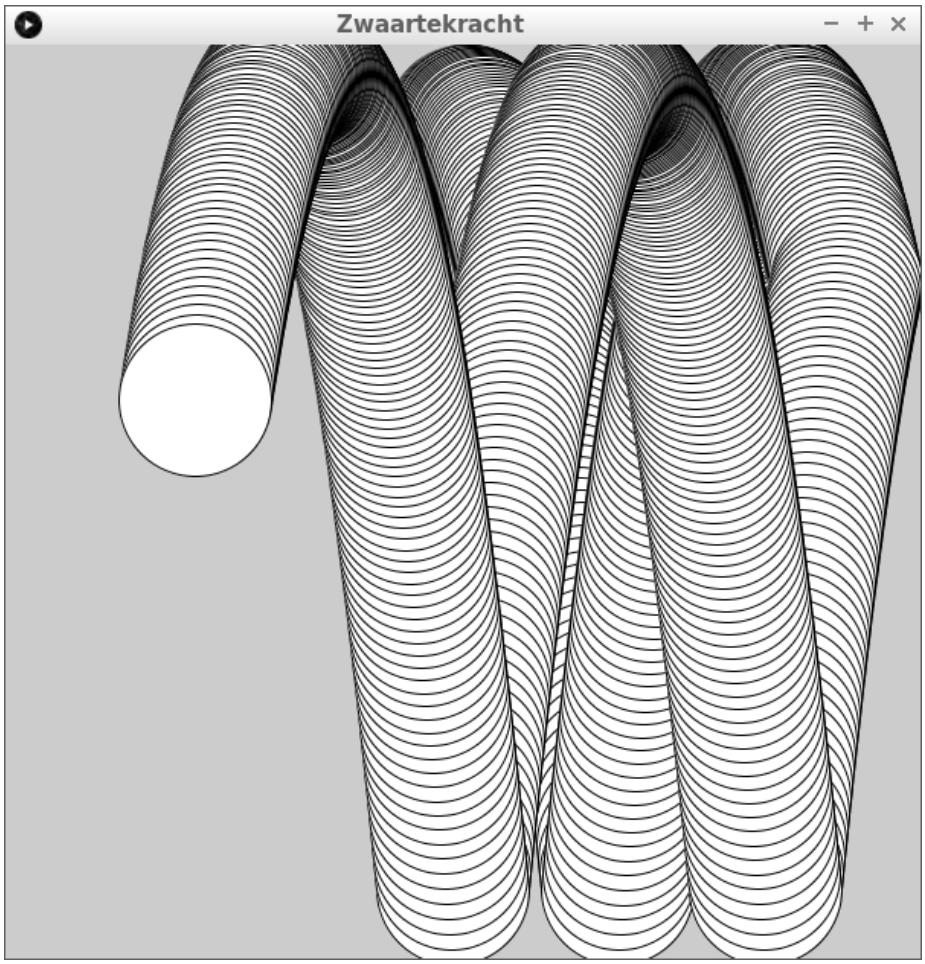


Figure 9: Zwaartekracht

Vragen

- Wat doet dit programma?
- In welke richting beweegt de ovaal?
- Blijft de ovaal zichtbaar op het scherm?

Opdrachten

- Laat de bal nu *ook* verticaal bewegen. Gebruik de variabele namen `y` en `dy`. Zorg dat de bal ook verticaal stuitert

Zwaartekracht

De zwaartekracht trekt aan voorwerpen. Iets dat omlaag valt, gaat hierdoor steeds sneller vallen. Iets dat omhoog gaat, gaat eerst steeds langzamer omhoog, en gaat daarna ook vallen. In de natuukunde gebruiken ze `g` (van 'gravity', dit is Engels voor zwaartekracht) voor de zwaartekracht.

Opdacht

In onze code hebben we nu een `dy`. Dit is de snelheid in de y richting. Elke beurt moet `dy` nu `g` groter worden. Een mooie waarde voor `g` is `0.1`.

Voeg toe dat de cirkel op een natuurlijke manier omlaag valt.

Twee variabelen

Nu onthoudt de computer een variabele: de x-coordinaat van de ovaal. Om een bal te laten stuiteren, moet er ook een richting onthouden worden: de bal gaat immers of naar links of naar rechts.

In deze code wordt de richting van de bal `dx` genoemd. Dit is een afkorting van 'delta x' en dat is weer wiskundetaal voor 'de verandering van x'.

Vragen

```
float x = 300;
float dx = 2;

void setup()
{
  size(600, 100);
```

```
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + dx;
  if (x > 650)
  {
    x = -50;
  }
}
```

- In welke richting beweegt de bal?
- Hoeveel pixels beweegt de bal per keer?
- Zet de waarde van `dx` op `1`. Wat zie je?
- Zet de waarde van `dx` op `0`. Wat zie je?
- Zet de waarde van `dx` op `-1`. Wat zie je?
- Bij sommige waarden van `dx` gaat de bal links het beeld uit. Maak een tweede if-statement, die ervoor zorgt dat de bal eeuwig links kan gaan. Test dit bij een `dx` van `2`, `0` en `-2`.
- Wat moet er met de `dx` gebeuren om de bal te laten stuiteren? Probeer dit!

Stuiteren

Als een bal met een snelheid van drie pixels naar rechst gaat en stuitert, dan gaat deze vanaf dan drie pixels naar links. Andersom is dat ook zo.

Hier is een manier om de bal te laten stuiteren:

```
float x = 300;
float dx = 2;

void setup()
{
  size(600, 100);
}

void draw()
{
  ellipse(x,50,100,100);
  x = x + dx;
  if (x > 650)
  {
```

```
    dx = -2;
  }
  if (x < 50)
  {
    dx = 2;
  }
}
```

Vragen

- Kopieer en run de code. Stuitert de bal?
- Verander de snelheid van de bal naar een pixel per keer. Op hoeveel plekken moet je de code aanpassen?
- Verander de snelheid van de bal naar drie pixels per keer. Op hoeveel plekken moet je de code aanpassen?
- Verander de snelheid van de bal terug naar een pixel per keer. Op hoeveel plekken moet je de code aanpassen? Laat nu de bal in het begin naar links gaan. Op hoeveel plekken moet je de code aanpassen?

De richting omklappen

Er is een slimmere manier om `dx` te veranderen. We hebben gezien dat als `dx` gelijk was aan `2`, deze `-2` wordt bij bij een stuiter. We hebben gezien dat als `dx` gelijk was aan `-2`, deze `2` wordt bij bij een stuiter. Er komt een minnetje voor, of er komt een minnetje bij. Dit is gemakkelijk op dezelfde manier te doen:

```
dx = -dx;
```

Hiermee zeg je 'Lieve computer, de nieuwe waarde van `dx` is min de oude waarde'. Als de oude waarde van `dx` `2` is, dan wordt deze nu `-2`. Als de oude waarde van `dx` `-2` is, dan wordt deze nu `--2` (jawel, min min twee) en dat mag je schrijven als `2` (omdat min keer min is plus).

Opdracht

- Gebruik nu de slimme manier om een bal te laten stuiteren.

Het lijkt al een beetje op Lonelier Pong. Dit is geen toeval :-)

`width` en `height`

`width` en `height` zijn ingebouwde variabelen in Processing, die handig zijn om te gebruiken zodat je programma nog werkt als je de grootte van je scherm aanpast.

Stel dat je een programma maakt wat een ovaal tekent die het scherm opvult, deze zou er zo uit kunnen zien:

```
void setup() {
  size(256, 256);
  ellipse(128, 128, 256, 256);
}
```

Dit programma tekent dit:

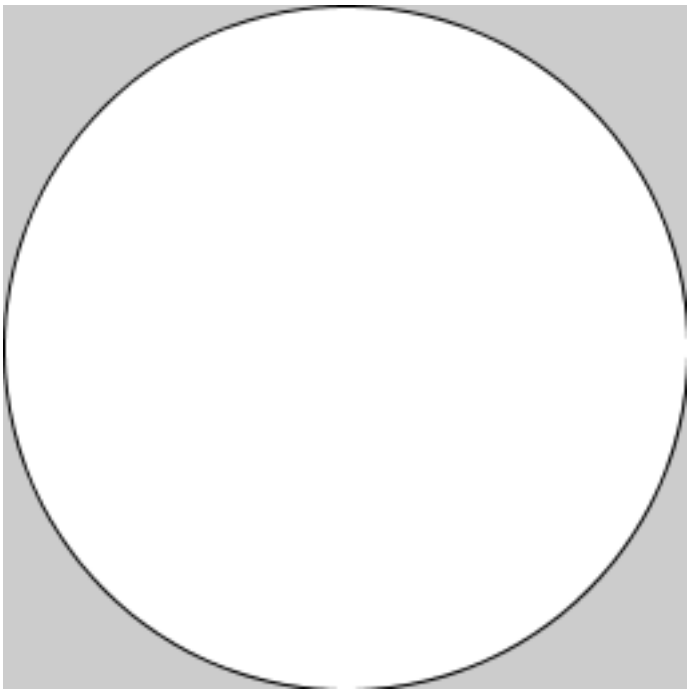


Figure 10: circle 256

Maar dit programma werkt alleen voor een scherm wat 256 bij 256 pixels is. Dat is natuurlijk onhandig, want elke keer als je een nieuwe grootte kiest moet je een heleboel code opnieuw typen!

Als we de breedte en hoogte van het scherm weten, weten we ook welke getallen in `ellipse` moeten. De x cordinaat van de ellipse is namelijk de helft van de

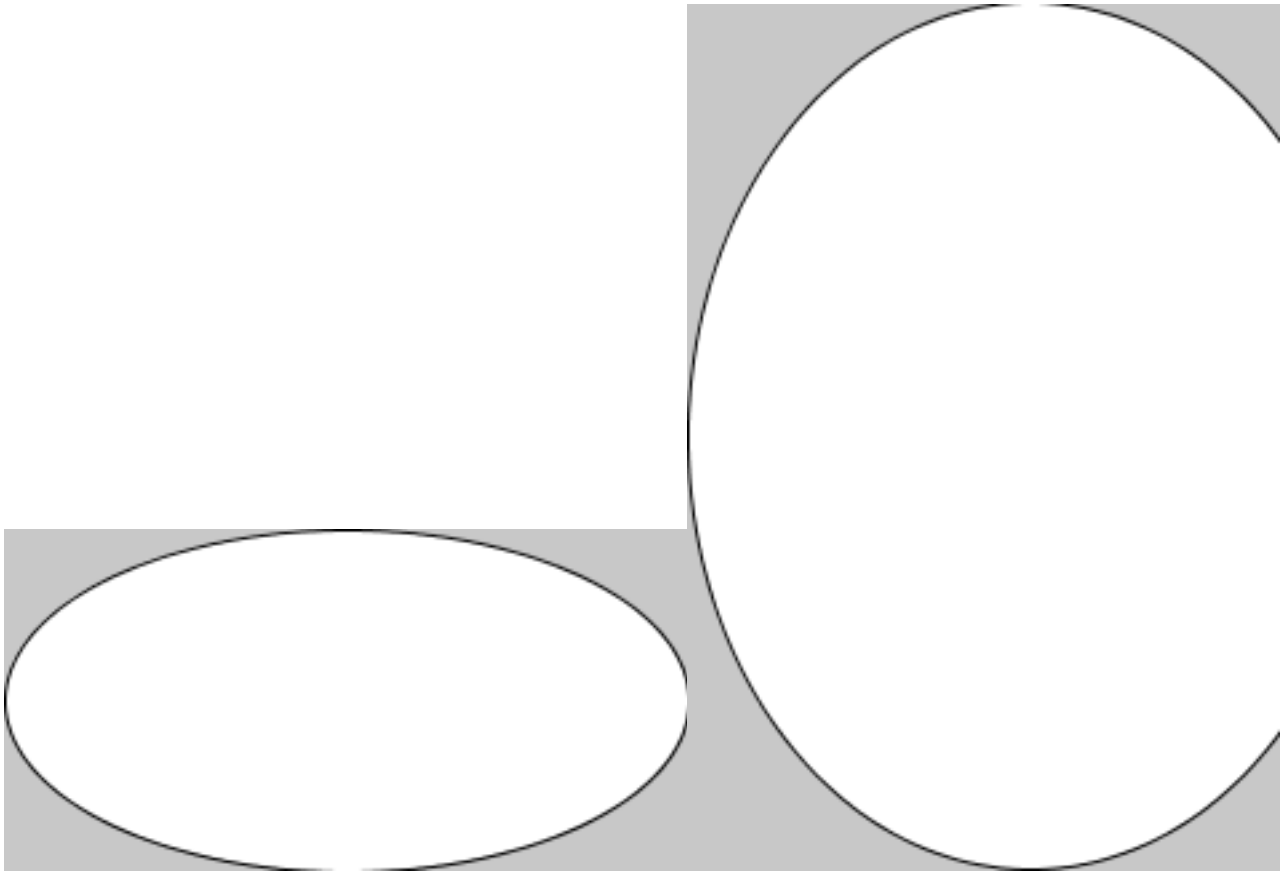
breedte, de y cordinaat de helft van de hoogte. En de breedte en hoogte van de `ellipse` zijn hetzelfde als die van het scherm.

Gelukkig weet Processing de breedte en hoogte van het scherm. De breedte van het scherm heet in Processing `width` en de hoogte heet `height`. Deze getallen worden bepaald zodra je size gebruikt om de grootte van je scherm te defineren.

Het programma wat een ovaal tekent die het scherm opvult, ziet er dan zo uit:

```
void setup() {
  size(256, 256);
  ellipse(width/2, height/2, width, height);
}
```

Maar nu past de ovaal nog steeds als we de getallin in `size` veranderen!



Arrays1

Met arrays kun je de computer veel waardes laten onthouden: de coordinaten van kogels, meteorieten, vijanden.

In deze les gaan we leren

- waarom je arrays nodig hebt
- wat arrays zijn
- hoe je een array met een element gebruikt

Zo gaat het eruit zien:

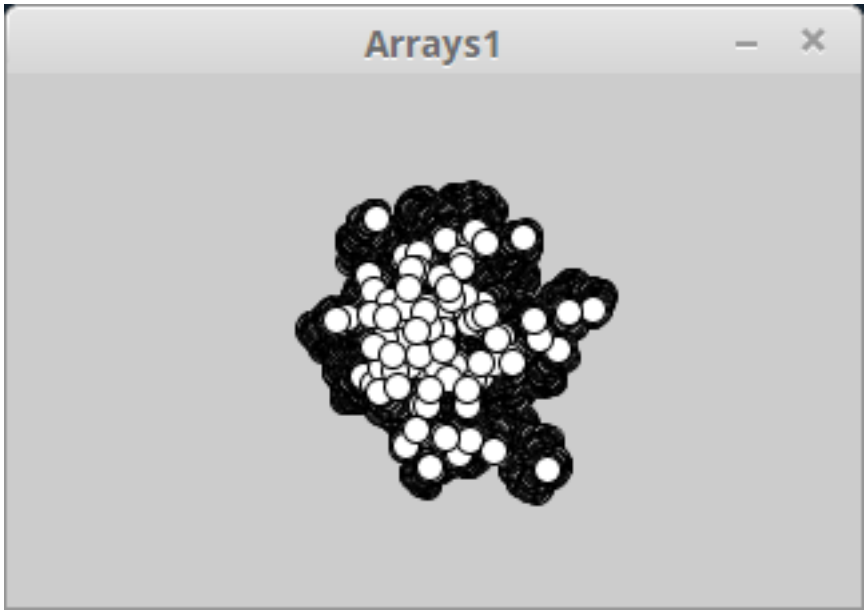


Figure 11: Arrays1

Rooksimulator met een deeltje

Je bent bezig een simulatie te maken: je wilt allemaal rookdeeltjes laten bewegen op het scherm.

Dit is je code:

```
float x = 160;
float y = 100;
```

```
void setup()
{
  size(320, 200);
}

void draw()
{
  x += random(-1,1);
  y += random(-1,1);
  ellipse(x, y, 10, 10);
}
```

Dit is wat de code betekent:

- `float x = 160`: 'Lieve computer, onthoudt een gebroken getal met de naam `x`, met als beginwaarde 160'. Dit wordt de x coordinaat van het eerste rookdeeltje
- `float y = 100`: 'Lieve computer, onthoudt een gebroken getal met de naam `y`, met als beginwaarde 100'. Dit wordt de y coordinaat van het eerste rookdeeltje
- `void setup() {}`: de klaarzet functie. Bij het opstarten wordt de code tussen de accolates een keer uitgevoerd
- `size(320, 200)`: maak een venster van 320 pixels breed en 200 pixels hoog
- `void draw() {}`: de teken functie. De code tussen de accolates wordt oneindig vaak uitgevoerd
- `x += random(-1,1)`: verander de waarde van `x` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig horizontaal bewegen
- `y += random(-1,1)`: verander de waarde van `y` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig verticaal bewegen
- `ellipse(x, y, 10, 10)`: teken een ovaal met als middelpunt (`x`, `y`) met breedte 10 en hoogte 10. Teken het eerste rookdeeltje

Vragen

- Run deze code
- Zorg dat er een tweede rookdeeltje bijkomt
- Hoeveel regels code kost het ongeveer om honderd rookdeeltjes te programmeren?

Rooksimulator met twee deeltjes

Je bent bezig een simulatie te maken: je wilt allemaal rookdeeltjes laten bewegen op het scherm.

Dit is je code:

```
float x1 = 160;
float y1 = 100;
float x2 = 160;
float y2 = 100;

void setup()
{
  size(320, 200);
}

void draw()
{
  x1 += random(-1,1);
  y1 += random(-1,1);
  ellipse(x1, y1, 10, 10);
  x2 += random(-1,1);
  y2 += random(-1,1);
  ellipse(x2, y2, 10, 10);
}
```

Dit is wat de code betekent:

- `float x1 = 160`: 'Lieve computer, onthoudt een gebroken getal met de naam `x1`, met als beginwaarde 160'. Dit wordt de x coordinaat van het eerste rookdeeltje
- `float y1 = 100`: 'Lieve computer, onthoudt een gebroken getal met de naam `y1`, met als beginwaarde 100'. Dit wordt de y coordinaat van het eerste rookdeeltje
- `float x2 = 160`: 'Lieve computer, onthoudt een gebroken getal met de naam `x2`, met als beginwaarde 160'. Dit wordt de x coordinaat van het tweede rookdeeltje
- `float y2 = 100`: 'Lieve computer, onthoudt een gebroken getal met de naam `y2`, met als beginwaarde 100'. Dit wordt de y coordinaat van het tweede rookdeeltje
- `void setup() {}`: de klaarzet functie. Bij het opstarten wordt de code tussen de accolates een keer uitgevoerd
- `size(320, 200)`: maak een venster van 320 pixels breed en 200 pixels hoog

- `void draw() {}`: de teken functie. De code tussen de accolates wordt oneindig vaak uitgevoerd
- `x1 += random(-1,1)`: verander de waarde van `x1` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig horizontaal bewegen
- `y1 += random(-1,1)`: verander de waarde van `y1` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig verticaal bewegen
- `ellipse(x1, y1, 10, 10)`: teken een ovaal met als middelpunt (`x1`, `y1`) met breedte 10 en hoogte 10. Teken het eerste rookdeeltje
- `x2 += random(-1,1)`: verander de waarde van `x2` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 2 willekeurig horizontaal bewegen
- `y2 += random(-1,1)`: verander de waarde van `y2` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 2 willekeurig verticaal bewegen
- `ellipse(x2, y2, 10, 10)`: teken een ovaal met als middelpunt (`x2`, `y2`) met breedte 10 en hoogte 10. Teken het tweede rookdeeltje

Vragen

- Run deze code
- Zorg dat er een derde rookdeeltje bijkomt
- Hoeveel regels code kost het ongeveer om honderd rookdeeltjes te programmeren?

Waarom arrays?

Dit is de code met drie rookdeeltjes:

```
float x1 = 160;
float y1 = 100;
float x2 = 160;
float y2 = 100;
float x3 = 160;
float y3 = 100;

void setup()
{
  size(320, 200);
  x1 = 160;
  y1 = 100;
  x2 = 160;
  y2 = 100;
  x3 = 160;
  y3 = 100;
}
```

```
void draw()
{
  x1 += random(-1,1);
  y1 += random(-1,1);
  ellipse(x1, y1, 10, 10);
  x2 += random(-1,1);
  y2 += random(-1,1);
  ellipse(x2, y2, 10, 10);
  x3 += random(-1,1);
  y3 += random(-1,1);
  ellipse(x3, y3, 10, 10);
}
```

Het valt op dat er veel herhaling in zit. Dit komt omdat we de computer steeds een getal tegelijk laten onthouden: `float x1 = 160` betekent 'Lieve computer, onthoudt een gebroken getal met de naam `x1`, met als beginwaarde 160'. Wat we willen kunnen zeggen is 'Lieve computer, onthoud keiveel gebroken getallen'. Dit is precies wat een array kan doen.

Wat is een array?

Een array kun je zien als een kast met laatjes. In deze les beginnen we met een kast met een laatje:

Elk laatje heeft een nummer en in elk laatje kan een getal.

Hier zie je het nummer van het laatje, en het getal wat erin zit:

Het laatje heeft nummer *nul* (links) en in het laatje zit het getal tweeenveertig.

Het valt op dat het laatje nummer *nul* heeft. Je zegt: 'Het eerste laatje heeft index nul'. Als je normaal telt, begin je bij een. Bij indices (het meervoud van index) begin je te tellen bij nul. De kast heeft een laatje, met index nul.

Vragen

- Wat is een array?
- Wat is een index?
- Wat is de laagste index?

Werken met een array met een laatje

Stel we willen een array maken van gebroken getallen (`floats`) met de naam `geheime_getallen`, dan moeten we boven de `setup` het volgende typen:

Figure 12: Kast met laatje

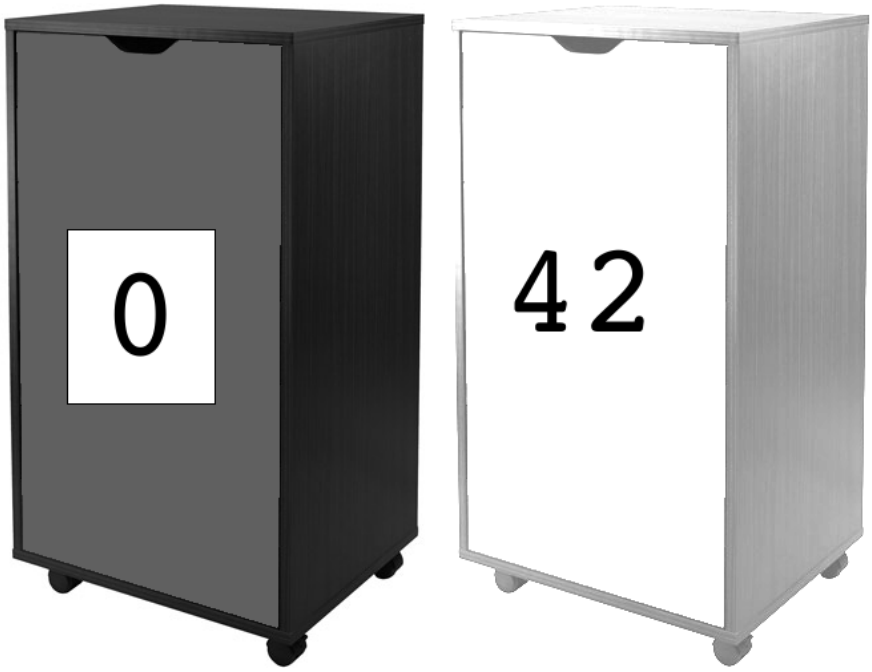


Figure 13: Kast met genummerde laatjes

```
float[] geheime_getallen;
```

Hiermee zeg je: 'Lieve computer, onthoud keiveel gebroken getallen met de naam `geheime_getallen`'.

Er is nog niet gezegd *hoeveel* gebroken getallen dat zijn. Vaak wordt de `setup` functie gebruikt om te zeggen hoeveel getallen er onthouden moeten worden:

```
geheime_getallen = new float[1];
```

Hiermee zeg je: 'Lieve computer, maak `geheime_getallen` groot genoeg om een gebroken getal (`floats`) te onthouden'.

Om de kast met de laatjes na te maken, kun je de volgende code gebruiken:

```
geheime_getallen[0] =  42;
```

Hiermee zeg je, in de derde regel: 'Lieve computer, stop in laatje met index nul het getal tweeenveertig'. Deze code zou prima in de `setup` functie gedaan kunnen worden.

Je zou ook de waarde in de laatjes kunnen lezen:

```
float x = geheime_getallen[0];
ellipse(x,200,300,400);
```

Hiermee zeg je: 'Lieve computer, kijk wat er in laatje met index nul zit en onthoud dat als `x`. Teken dan een ovaal die `x` pixels naar rechts is, 200 pixels omlaag is, 300 pixels breed is, en 400 pixels hoog is.'.

Alles bij elkaar krijg je dit programma:

```
float[] geheime_getallen;

void setup()
{
  size(400,400);
  geheime_getallen = new float[1];
  geheime_getallen[0] = 42;
}

void draw()
{
  float x = geheime_getallen[0];
  ellipse(x,200,300,400);
}
```

Dit programma ziet er niet erg mooi uit. Het is bedoeld om je te laten hoe je arrays maakt, vult en leest.

Vragen

- Welke foutmelding krijg je als je `float[] geheime_getallen;` in de `setup` functie zet?
- Welke foutmelding krijg je als je `float geheime_getallen;` (dus zonder blokhaken) gebruikt?
- Je wilt een array maken van gebroken getallen met de naam `snelheden`. Hoe zeg je dat in code?

Verder

Je zou nu kunnen doen:

- Arrays 2

Arrays2

Met arrays kun je de computer veel waardes laten onthouden: de coordinaten van kogels, meteorieten, vijanden. Met een for loop kun je door een array heen gaan.

In deze les gaan we leren

- waarom je arrays nodig hebt
- wat arrays zijn
- hoe je arrays met een for loop gebruikt

Zo gaat het eruit zien:

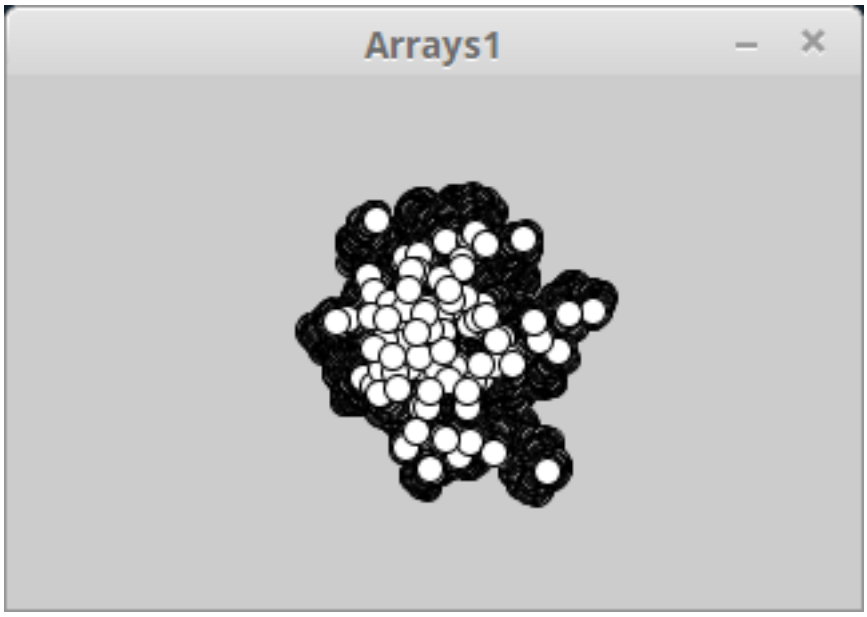


Figure 14: Arrays2

Rooksimulator

Je bent bezig een simulatie te maken: je wilt allemaal rookdeeltjes laten bewegen op het scherm.

Dit is je code:

```
float x1 = 160;
```

```
float y1 = 100;
float x2 = 160;
float y2 = 100;

void setup()
{
  size(320, 200);
}

void draw()
{
  x1 += random(-1,1);
  y1 += random(-1,1);
  ellipse(x1, y1, 10, 10);
  x2 += random(-1,1);
  y2 += random(-1,1);
  ellipse(x2, y2, 10, 10);
}
```

Dit is wat de code betekent:

- `float x1 = 160`: 'Lieve computer, onthoudt een gebroken getal met de naam `x1`, met als beginwaarde 160'. Dit wordt de x coordinaat van het eerste rookdeeltje
- `float y1 = 100`: 'Lieve computer, onthoudt een gebroken getal met de naam `y1`, met als beginwaarde 100'. Dit wordt de y coordinaat van het eerste rookdeeltje
- `float x2 = 160`: 'Lieve computer, onthoudt een gebroken getal met de naam `x2`, met als beginwaarde 160'. Dit wordt de x coordinaat van het tweede rookdeeltje
- `float y2 = 100`: 'Lieve computer, onthoudt een gebroken getal met de naam `y2`, met als beginwaarde 100'. Dit wordt de y coordinaat van het tweede rookdeeltje
- `void setup() {}`: de klaarzet functie. Bij het opstarten wordt de code tussen de accolates een keer uitgevoerd
- `size(320, 200)`: maak een venster van 320 pixels breed en 200 pixels hoog
- `void draw() {}`: de teken functie. De code tussen de accolates wordt oneindig vaak uitgevoerd
- `x1 += random(-1,1)`: verander de waarde van `x1` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig horizontaal bewegen
- `y1 += random(-1,1)`: verander de waarde van `y1` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 1 willekeurig verticaal bewegen
- `ellipse(x1, y1, 10, 10)`: teken een ovaal met als middelpunt (`x1`, `y1`) met breedte 10 en hoogte 10. Teken het eerste rookdeeltje

- `x2 += random(-1,1)`: verander de waarde van `x2` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 2 willekeurig horizontaal bewegen
- `y2 += random(-1,1)`: verander de waarde van `y2` met een willekeurige waarden van -1 tot 1. Dit laat rookdeeltje 2 willekeurig verticaal bewegen
- `ellipse(x2, y2, 10, 10)`: teken een ovaal met als middelpunt (`x2`, `y2`) met breedte 10 en hoogte 10. Teken het tweede rookdeeltje

Vragen

- Run deze code
- Zorg dat er een derde rookdeeltje bijkomt
- Hoeveel regels code kost het ongeveer om honderd rookdeeltjes te programmeren?

Waarom arrays?

Dit is de code met drie rookdeeltjes:

```
float x1 = 160;
float y1 = 100;
float x2 = 160;
float y2 = 100;
float x3 = 160;
float y3 = 100;

void setup()
{
  size(320, 200);
  x1 = 160;
  y1 = 100;
  x2 = 160;
  y2 = 100;
  x3 = 160;
  y3 = 100;
}

void draw()
{
  x1 += random(-1,1);
  y1 += random(-1,1);
  ellipse(x1, y1, 10, 10);
  x2 += random(-1,1);
  y2 += random(-1,1);
```

```
  ellipse(x2, y2, 10, 10);
  x3 += random(-1,1);
  y3 += random(-1,1);
  ellipse(x3, y3, 10, 10);
}
```

Het valt op dat er veel herhaling in zit. Dit komt omdat we de computer steeds een getal tegelijk laten onthouden: `float x1 = 160` betekent 'Lieve computer, onthoudt een gebroken getal met de naam `x1`, met als beginwaarde 160'. Wat we willen kunnen zeggen is 'Lieve computer, onthoud keiveel gebroken getallen'. Dit is precies wat een array kan doen.

Wat is een array?

Een array kun je zien als een kast met laatjes:

Elk laatje heeft een nummer en in elk laatje kan een getal.

Hier zie je de nummers van de laatjes:

Het valt op dat het eerste laatje nummer *nul* heeft. Je zegt: 'Het eerste laatje heeft index nul'. Als je normaal telt, begin je bij een. Bij indices (het meervoud van index) begin je te tellen bij nul. De kast heeft zeven laatjes, met indices nul tot en met zes.

Vragen

- Wat is een array?
- Wat is een index?
- Een array kan drie getallen bevatten. Wat zijn de indices?
- Een array heeft als hoogste index `13`. Wat is de grootte van de array?
- Een array kan 314 getallen bevatten. Wat is de hoogste index?
- Een array heeft als hoogste index `0`. Wat is de grootte van de array?

Werken met arrays

Stel we willen een array maken van gebroken getallen (`floats`) met de naam `geheime_getallen`, dan moeten we boven de `setup` het volgende typen:

```
float[] geheime_getallen;
```

Hiermee zeg je: 'Lieve computer, onthoud keiveel gebroken getallen met de naam `geheime_getallen`'.

Er is nog niet gezegd *hoeveel* gebroken getallen dat zijn. Vaak wordt de `setup` functie gebruikt om te zeggen hoeveel getallen er onthouden moeten worden:

Figure 15: Kast met laatjes

Figure 16: Kast met genummerde laatjes

```
geheime_getallen = new float[7];
```

Hiermee zeg je: 'Lieve computer, maak `geheime_getallen` groot genoeg om zeven gebroken getallen (`float`s) te onthouden'.

Om de kast met de laatjes na te maken, kun je de volgende code gebruiken:

```
geheime_getallen[0] =  0;
geheime_getallen[1] =  1;
geheime_getallen[2] =  4;
geheime_getallen[3] =  9;
geheime_getallen[4] = 16;
geheime_getallen[5] = 25;
geheime_getallen[6] = 36;
```

Hiermee zeg je, in de derde regel: 'Lieve computer, stop in laatje met index twee het getal vier'. Deze code zou prima in de `setup` functie gedaan kunnen worden.

Dit kan ook slimmer. Hieronder zie je code die *precies* hetzelfde doet:

```
for (int i=0; i!=7; ++i)
{
  geheime_getallen[i] = i * i;
}
```

Hiermee zeg je: 'Lieve computer, laat een for loop lopen van nul tot zeven met `i` als teller. Stop in het `i`-de laatje het getal `i` keer `i`'.

Je zou ook de waarden in de laatjes kunnen lezen:

```
for (int i=0; i!=7; ++i)
{
  float x = geheime_getallen[i];
  ellipse(x,200,300,400);
}
```

Hiermee zeg je: 'Lieve computer, laat een for loop lopen van nul tot zeven met `i` als teller. Kijk wat er in het `i`-de laatje zit en onthoud dat als `x`. Teken dan een ovaal die `x` pixels naar rechts is, 200 pixels omlaag is, 300 pixels breed is, en 400 pixels hoog is.'.

Alles bij elkaar krijg je dit programma:

```
float[] geheime_getallen;

void setup()
```

```
{
  size(400,400);
  geheime_getallen = new float[7];
  for (int i=0; i!=7; ++i)
  {
    geheime_getallen[i] = i * i;
  }
}

void draw()
{
  for (int i=0; i!=7; ++i)
  {
    float x = geheime_getallen[i];
    ellipse(x,200,300,400);
  }
}
```

Dit programma ziet er niet erg mooi uit. Het is bedoeld om je te laten hoe je arrays maakt, vult en leest.

Vragen

- Welke foutmelding krijg je als je `float[] geheime_getallen;` in de `setup` functie zet?
- Welke foutmelding krijg je als je `float geheime_getallen;` (dus zonder blokhaken) gebruikt?
- Je wilt een array maken van gebroken getallen met de naam `snelheden`. Hoe zeg je dat in code?
- Je hebt een array van gebroken getallen met de naam `schades`. Je wilt dat deze 345 groot wordt. Hoe zeg je dat in code?
- Je hebt een array van gebroken getallen met de naam `roodwaarden` die 987 groot is. Je wilt alle `roodwaarden` op `128` zetten. Hoe doe je dat?
- Je hebt een array van gebroken getallen met de naam `breedtes` die 345 groot is. Je wilt 345 ovalen tekenen, op `123` pixels naar rechts, `234` pixels omlaag, met een breedte van `breedtes` en een hoogte van `345`. Hoe doe je dat?

Code met drie rookdeeltjes

Kijk naar de code met de drie rookdeeltjes:

```
float x1 = 160;
```

```
float y1 = 100;
float x2 = 160;
float y2 = 100;
float x3 = 160;
float y3 = 100;

void setup()
{
  size(320, 200);
  x1 = 160;
  y1 = 100;
  x2 = 160;
  y2 = 100;
  x3 = 160;
  y3 = 100;
}

void draw()
{
  x1 += random(-1,1);
  y1 += random(-1,1);
  ellipse(x1, y1, 10, 10);
  x2 += random(-1,1);
  y2 += random(-1,1);
  ellipse(x2, y2, 10, 10);
  x3 += random(-1,1);
  y3 += random(-1,1);
  ellipse(x3, y3, 10, 10);
}
```

Opdracht

- Schrijf deze code om dat deze gebruik maakt van arrays
- Verander de code zo dat er honderd rookdeeltjes kunnen zijn

Oplossing

Hier de oplossing:

```
final int aantal_punten = 100;
float[] xs;
float[] ys;

void setup()
```

```
{
  size(320, 200);
  xs = new float[aantal_punten];
  ys = new float[aantal_punten];
  for (int i=0; i!=aantal_punten; ++i)
  {
    xs[i] = 160;
    ys[i] = 100;
  }
}

void draw()
{
  for (int i=0; i!=aantal_punten; ++i)
  {
    xs[i] += random(-1,1);
    ys[i] += random(-1,1);
    ellipse(xs[i], ys[i], 10, 10);
  }
}
```

Dit doet de code:

- `final int aantal_punten = 100`: 'Lieve computer, onthoud een heel getal (`int`), die ik `aantal_punten` noem, met beginwaarde 100, die ik nooit zal veranderen (`final`)
- `float[] xs`: 'Lieve computer, onthoud een heleboel gebroken getallen (`float[]`), die ik `xs` noem'. Dit zijn de x-coordinaten (hoeveel pixels naar rechts) van de rookdeeltjes.
- `float[] ys`: 'Lieve computer, onthoud een heleboel gebroken getallen (`float[]`), die ik `ys` noem'. Dit zijn de y-coordinaten (hoeveel pixels omlaag) van de rookdeeltjes.
- `size(320, 200)`: maak het scherm 320 pixels breed en 200 pixels hoog
- `xs = new float[aantal_punten]`: 'Lieve computer, maak `xs` groot genoeg om `aantal_punten` (dus 100) gebroken getallen (`float`s) te onthouden'
- `ys = new float[aantal_punten]`: 'Lieve computer, maak `ys` groot genoeg om `aantal_punten` (dus 100) gebroken getallen (`float`s) te onthouden'

```
for (int i=0; i!=aantal_punten; ++i)
{
  xs[i] = 160;
  ys[i] = 100;
}
```

- Laat een teller lopen van nul to `aantal_punten` (dus 100) in stapjes van een en noem het tellertje `i`. Zet de `i`-de `xs` op `160`. Zet de `i`-de `ys` op `100`.

```
for (int i=0; i!=aantal_punten; ++i)
{
  xs[i] += random(-1,1);
  ys[i] += random(-1,1);
  ellipse(xs[i], ys[i], 10, 10);
}
```

- Laat een teller lopen van nul to `aantal_punten` (dus 100) in stapjes van een en noem het tellertje `i`. Verander de `i`-de `xs` willekeurig met -1, 0 of +1. Verander de `i`-de `ys` willekeurig met -1, 0 of +1. Teken een ellipse met de `i`-de `xs` naar rechts, de `i`-de `ys` omlaag, 10 pixels breed en 10 pixels hoog

PImage les 1

In deze les gaan we met een plaatje werken.

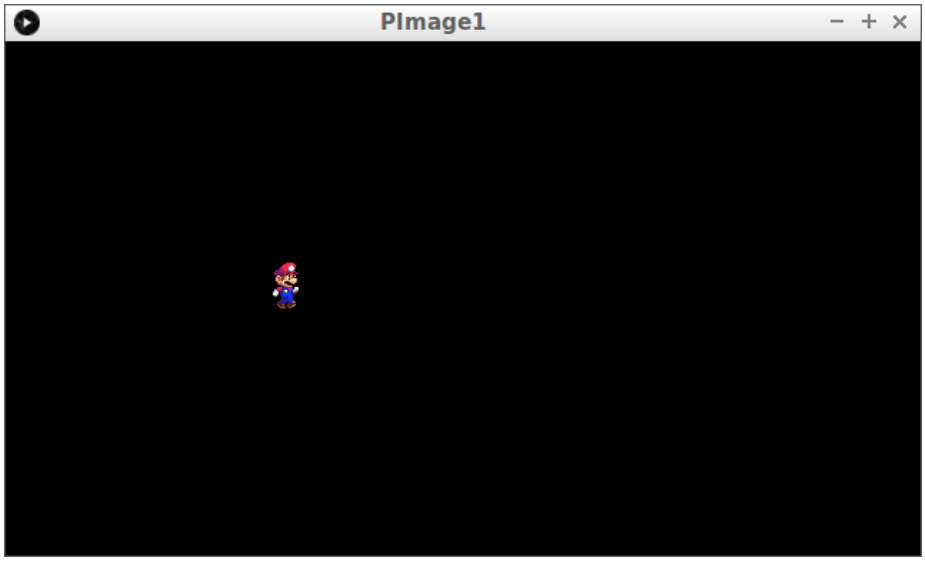


Figure 17: PImage1.png

Een plaatje vinden

Op de GitHub van de cursus hebben we een aantal goede plaatjes verzameld.

Er zijn twee manieren de plaatjes te downloaden:

- Ga naar https://github.com/richelbilderbeek/Dojo/tree/master/Sprites. Klik dan op de naam van een plaatje. Nu kun je het plaatje zien. Klik met de rechtermuisknop op het plaatje en kies 'Afbeelding opslaan als'/'Save image as'
- Je kunt alle plaatjes in een keer downloaden. Download de cursus als zip hier: https://github.com/richelbilderbeek/Dojo/archive/master.zip. Kies 'Alles uitpakken'/'Extract all' om het zipje uit te pakken. In de folder 'Sprites' staan de plaatjes

In dit voorbeeld gebruik ik dit plaatje van Mario:

Figure 18: mario.png

Code

De code om Mario op de muis cursor te laten zien is simpel:

```
PImage plaatje;

void setup() {
  size(640, 360);
  plaatje = loadImage("mario.png");
}

void draw() {
  background(0);
  image(plaatje, mouseX, mouseY);
}
```

- `PImage plaatje`: onthoud een `PImage` met de naam `plaatje`. Let op: `PImage` begint met twee hoofdletters
- `void setup() {}`: de setup functie, de computer voert een keer alles tussen de accolades uit
- `size(640, 360)`: maak een scherm van 640 pixels breed en 360 pixels hoog
- `plaatje = loadImage("mario.png")`. Laat `plaatje` de afbeelding van Mario krijgen, door het bestand `mario.png` te laden

- `void draw() {}`: de draw functie, de computer voert steeds alles tussen de accolades uit
- `background(0)`: maak de achtergrond zwart
- `image(plaatje, mouseX, mouseY)`: teken (de linkerbovenhoek van) het plaatje `plaatje` op de plek waar de muiscursus is.

Dit werkt niet meteen, omdat de bestanden op de juiste plek moeten staan.

Bestanden op de juiste plek

Hier zie je een plaatje waarop staat waar de bestanden moeten staan:

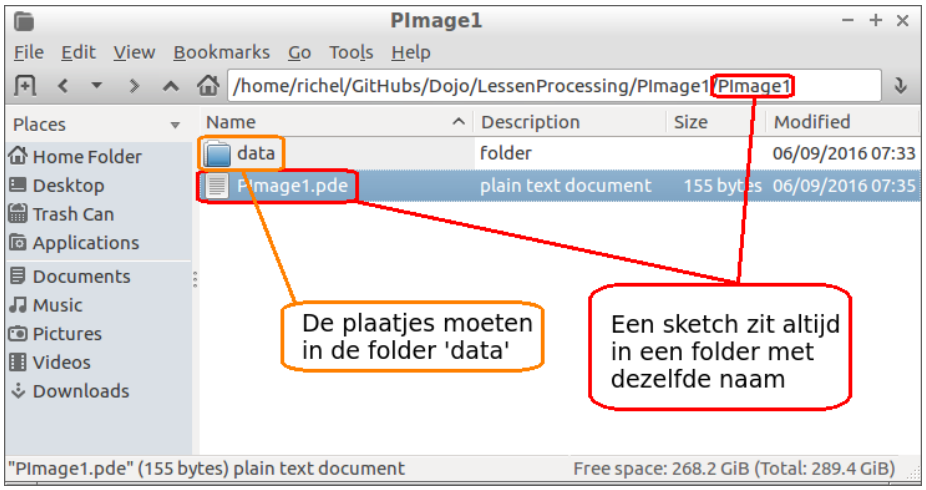


Figure 19: Folderstructuur.png

- De sketch heet `PImage1.pde`. Daarom staat deze in de map `PImage1`
- De sketch heeft een folder `data`. Hierin staat het plaatje, `mario.png`

Opdracht

- Krijg bovenstaand voorbeeld werkend
- Maak een eigen programma met een ander plaatje