

Empowering Creative Thinking Through Programming

Les 9: Images, Rotatie en de return van parameters

Wat gaan we doen

Images voorbereiden

Images laden

Images afbeelden

Roteren

Parameters naar een class/instance

Images (grafisch materiaal)

- Processing slikt aantal file formats.
- Raster (jpg, gif, png (transparantie)
 - Vector (svg) (niet te moeilijk)
- images moeten in de data map in de projectmap staan

Hoe importeer ik een image?

Zelfde stappen als bij variabele/instance:

0: importeren in sketch map: Schets-> Bestand toevoegen

Code

1: ruimte maken van het juiste soort en naam geven

2: ruimte vullen met bestand (jpg/png/svg).

3: afbeelden (soort van rect() maar dan voor image)

Aparte opdrachten voor raster en vector

Maar de procedure is hetzelfde

Raster

voorbeeld_1

```
1 // voorbeeld image
2 // door Ton Markus En Vincent Booman, oktober 2019
3
4 PImage hoofd; //1
5
6 void setup() {
7   size(500,500);
8   hoofd = loadImage("hoofd.png"); //2
9 }
10
11 void draw() {
12   background(100);
13
14   image(hoofd, 100, 200); //3a
15   image(hoofd, 200, 100, 300, 50); //3b
16
17   x, y, width, height
18 }
```

Vector

voorbeeld_2

```
1 // voorbeeld shape
2 // door Ton Markus En Vincent Booman, oktober 2019
3
4 PShape body; //1
5
6 void setup() {
7   size(500,500);
8   body = loadShape("body.svg"); //2
9 }
10
11 void draw() {
12   background(100);
13
14   shape(body, 100, 200); //3a
15   shape(body, 200, 100, 300, 50); //3b
16 }
17
18 x, y, width, height
19
20 }
```

Overig:

- Veel hetzelfde als bij rect():

`rectMode() = imageMode() = shapeMode()`

Voor de dapperen: zelf shapes tekenen in Processing

Roteren => rotate()

- Is een goocheltrucje in Processing
- lang technisch verhaal (dat ik jullie wil besparen)
 - korte versie van de problemen:

A: roteren gebeurt standaard vanuit linkerbovenhoek

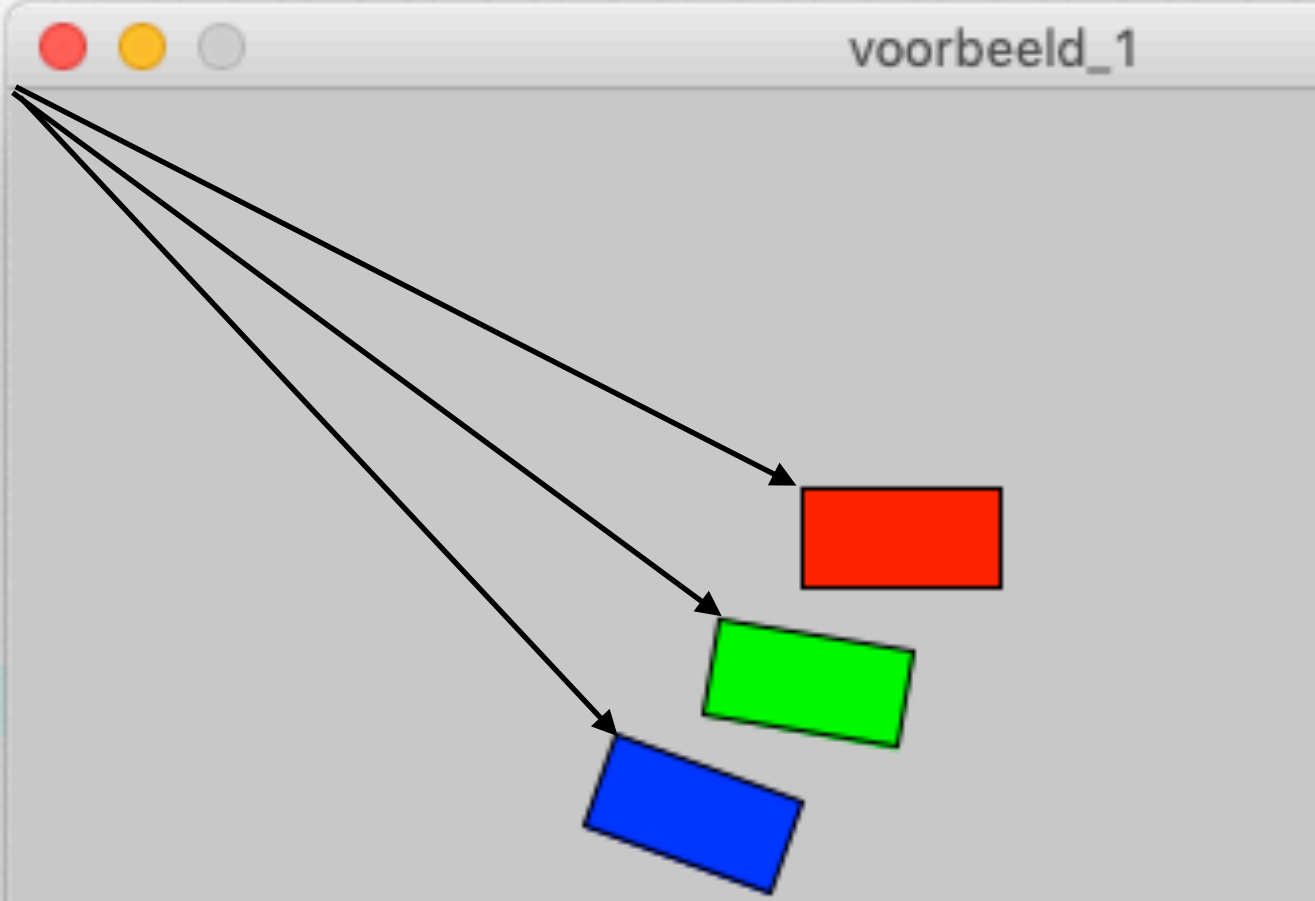
B: Rotatie opdracht wordt op alles uitgevoerd dat na rotatie-opdracht getekend wordt.

(dat 'doorwerken' geldt voor veel opdrachten, denk ook aan rectMode() die door blijft werken totdat er een nieuwe rectMode wordt opgegeven)

A: rotatie vanuit linkerbovenhoek venster

Roteer PI/18 is ongeveer 10° :

```
1 // voorbeeld 1: rotatie vanuit 0,0
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   rect(200,100,50,25);
7
8   rotate(PI/18.0);
9   fill(0,255,0);
10  rect(200,100,50,25);
11
12  rotate(PI/18.0);
13  fill(0,0,255);
14  rect(200,100,50,25);
15 }
16
17
```



Rotatie niet vanuit 200,100 maar vanuit Linkerbovenhoek (0,0)

Waarom?

- Teken een rechthoek: `rect(200,100,50,25);`
- Opdracht is “begin bij een ‘beginpositie’ van 0,0 (linkerbovenhoek) en teken daarvandaan op positie 200,100 een rechthoek.
- Maar bij de rotatie kan je niet opgeven dat ie op een andere plek dan de beginpositie van 0,0 de rotatie moet uitvoeren
- Dus `rotate(200,100,PI/18.0)` gaat (helaas) niet werken....

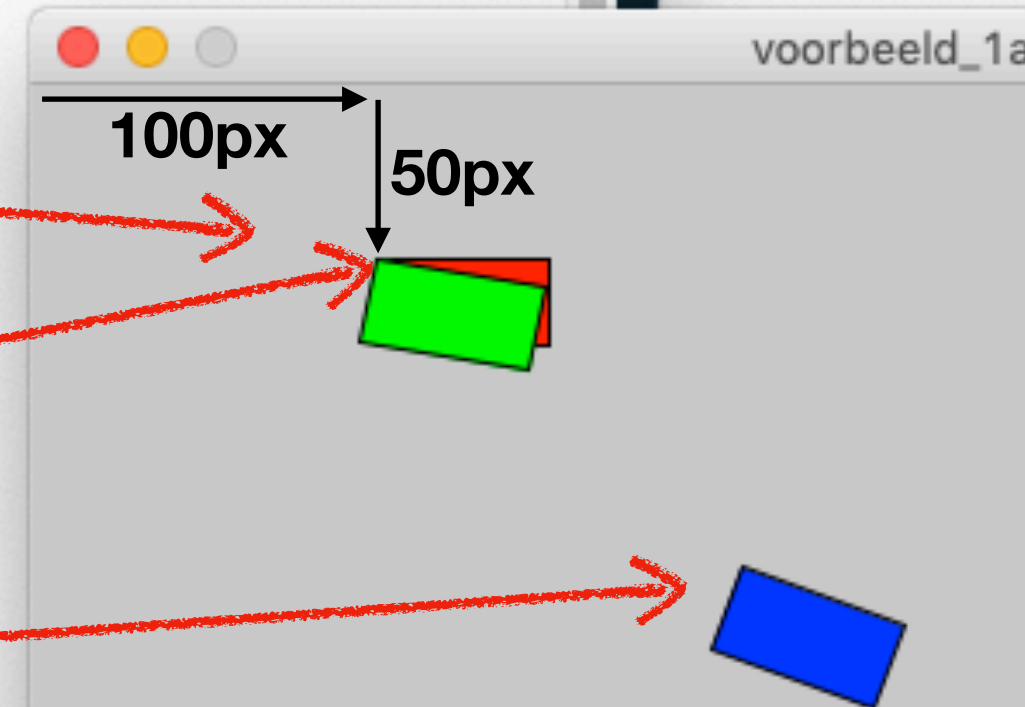
Hoe lossen we dit op?

- Met een opdracht die de beginpositie op een andere plek dan de linkerbovenhoek van het venster neerzet.
 - Die opdracht is `translate()`;
- `translate(150,200)` wil zeggen: Ga alle opdrachten die hierna komen niet vanuit de linkerbovenhoek van het venster maar vanuit 150,200 uitvoeren
- Let op: dit geldt voor alle opdrachten, dus ook `rect()`

Rotatie vanuit nieuw beginpunt

voorbeeld_1a

```
1 // voorbeeld 1a: translatie met rotatie
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   translate(100,50); // beginpunt verleggen
7   rect(0,0,50,25); // rect vanuit nieuw beginpunt
8
9   rotate(PI/18.0); // rotate vanuit nieuw beginpunt
10  fill(0,255,0);
11  rect(0,0,50,25); // rect vanuit nieuw beginpunt
12
13  translate(120,70); // beginpunt nogmaals verleggen
14  rotate(PI/18.0); // rotate vanuit nieuw nieuw beginpunt
15  fill(0,0,255);
16  rect(0,0,50,25); // rect vanuit nieuw nieuw beginpunt
17 }
```



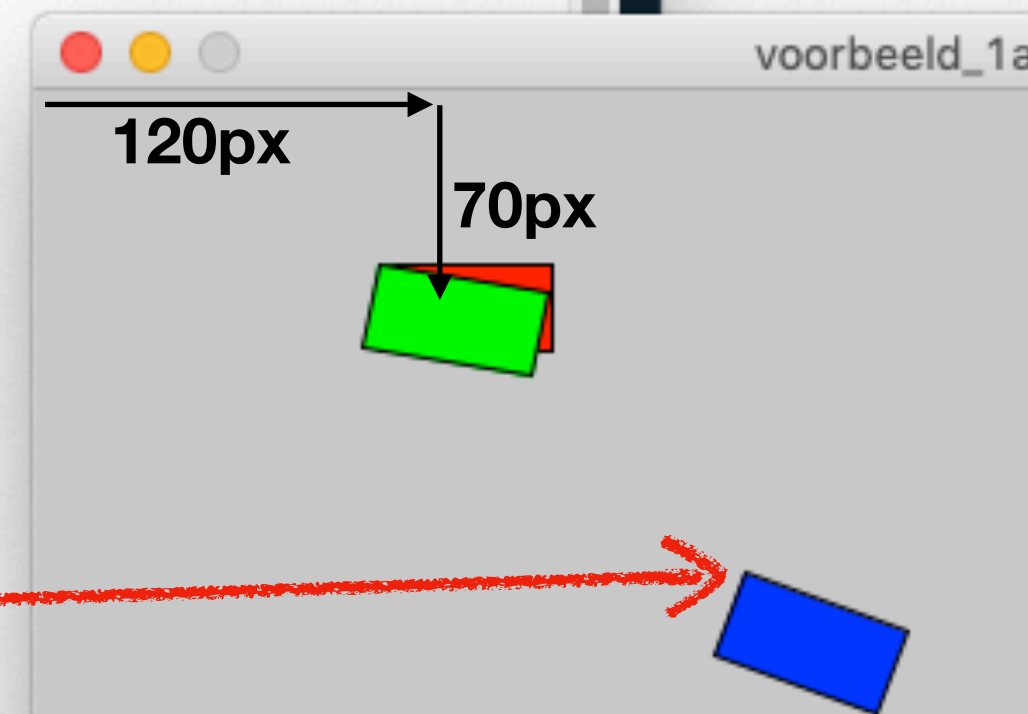
Let op dat de `rect()` opdrachten ook allemaal met 0,0 beginnen omdat ze niet meer met de linkerbovenhoek van het venster als beginpunt worden getekend, maar vanuit het 'nieuwe' beginpunt na de `translate()` opdracht.

Maar waarom staat dat blauwe blokje zo ver weg bij de rest?

Want de 2e translate scheelt maar 20 pixels met de 1e

voorbeeld_1a

```
1 // voorbeeld 1a: translatie met rotatie
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   translate(100,50); // beginpunt verleggen
7   rect(0,0,50,25); // rect vanuit nieuw beginpunt
8
9   rotate(PI/18.0); // rotate vanuit nieuw beginpunt
10  fill(0,255,0);
11  rect(0,0,50,25); // rect vanuit nieuw beginpunt
12
13  translate(120,70); // beginpunt nogmaals verleggen
14  rotate(PI/18.0); // rotate vanuit nieuw nieuw beginpunt
15  fill(0,0,255);
16  rect(0,0,50,25); // rect vanuit nieuw nieuw beginpunt
17 }
```

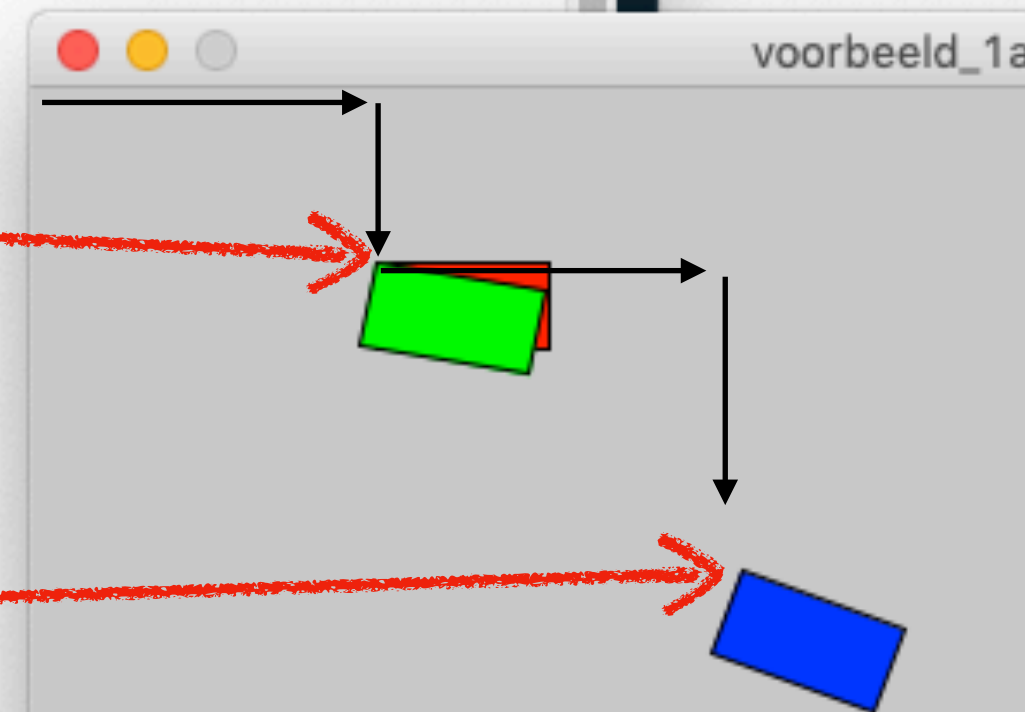


B: opdrachten worden opgeteld

De 2 translate() en de 2 rotate() opdrachten worden opgeteld

voorbeeld_1a

```
1 // voorbeeld 1a: translatie met rotatie
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   translate(100,50); // beginpunt verleggen
7   rect(0,0,50,25); // rect vanuit nieuw beginpunt
8
9   rotate(PI/18.0); // rotate vanuit nieuw beginpunt
10  fill(0,255,0);
11  rect(0,0,50,25); // rect vanuit nieuw beginpunt
12
13  translate(120,70); // beginpunt nogmaals verleggen
14  rotate(PI/18.0); // rotate vanuit nieuw nieuw beginpunt
15  fill(0,0,255);
16  rect(0,0,50,25); // rect vanuit nieuw nieuw beginpunt
17 }
```

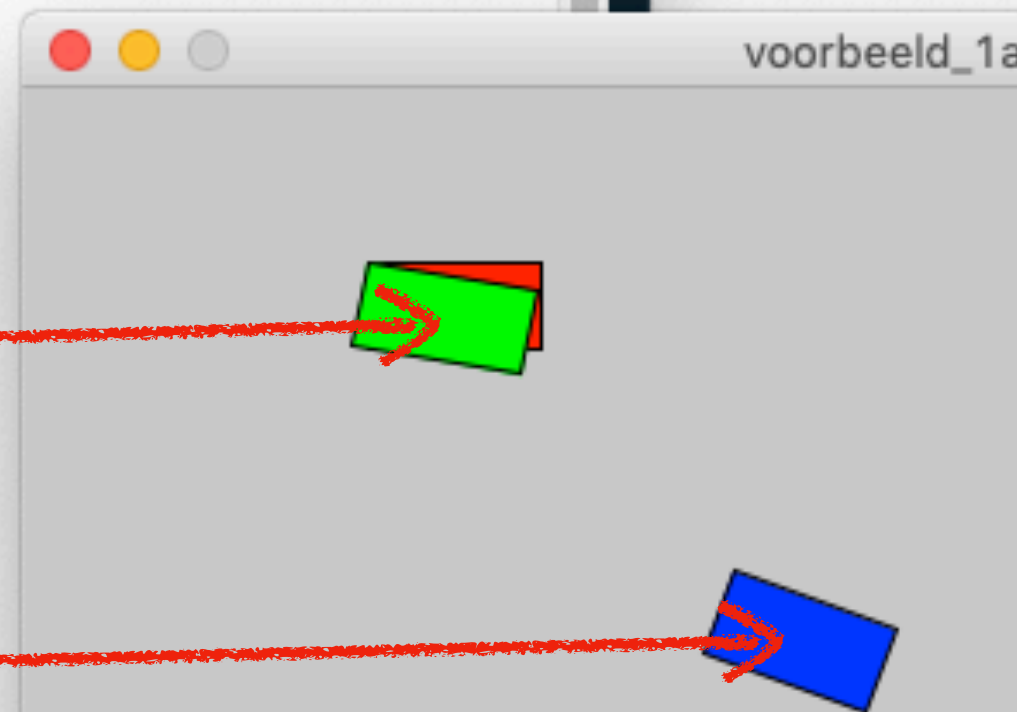


B: opdrachten worden opgeteld

De 2 translate() en de 2 rotate() opdrachten worden opgeteld

voorbeeld_1a

```
1 // voorbeeld 1a: translatie met rotatie
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   translate(100,50); // beginpunt verleggen
7   rect(0,0,50,25); // rect vanuit nieuw beginpunt
8
9   rotate(PI/18.0); // rotate vanuit nieuw beginpunt
10  fill(0,255,0);
11  rect(0,0,50,25); // rect vanuit nieuw beginpunt
12
13  translate(120,70); // beginpunt nogmaals verleggen
14  rotate(PI/18.0); // rotate vanuit nieuw nieuw beginpunt
15  fill(0,0,255);
16  rect(0,0,50,25); // rect vanuit nieuw nieuw beginpunt
17 }
```



Regel 9 en regel 14 geven dezelfde rotate() opdracht, maar het blauwe blokje is verder geroteerd dan het groene blokje.

Waarom?

- Computers zijn dom
- Of eigenlijk; computers zijn braaf. Ze onthouden alle opdrachten die je ze geeft en voeren die netjes achter elkaar uit...

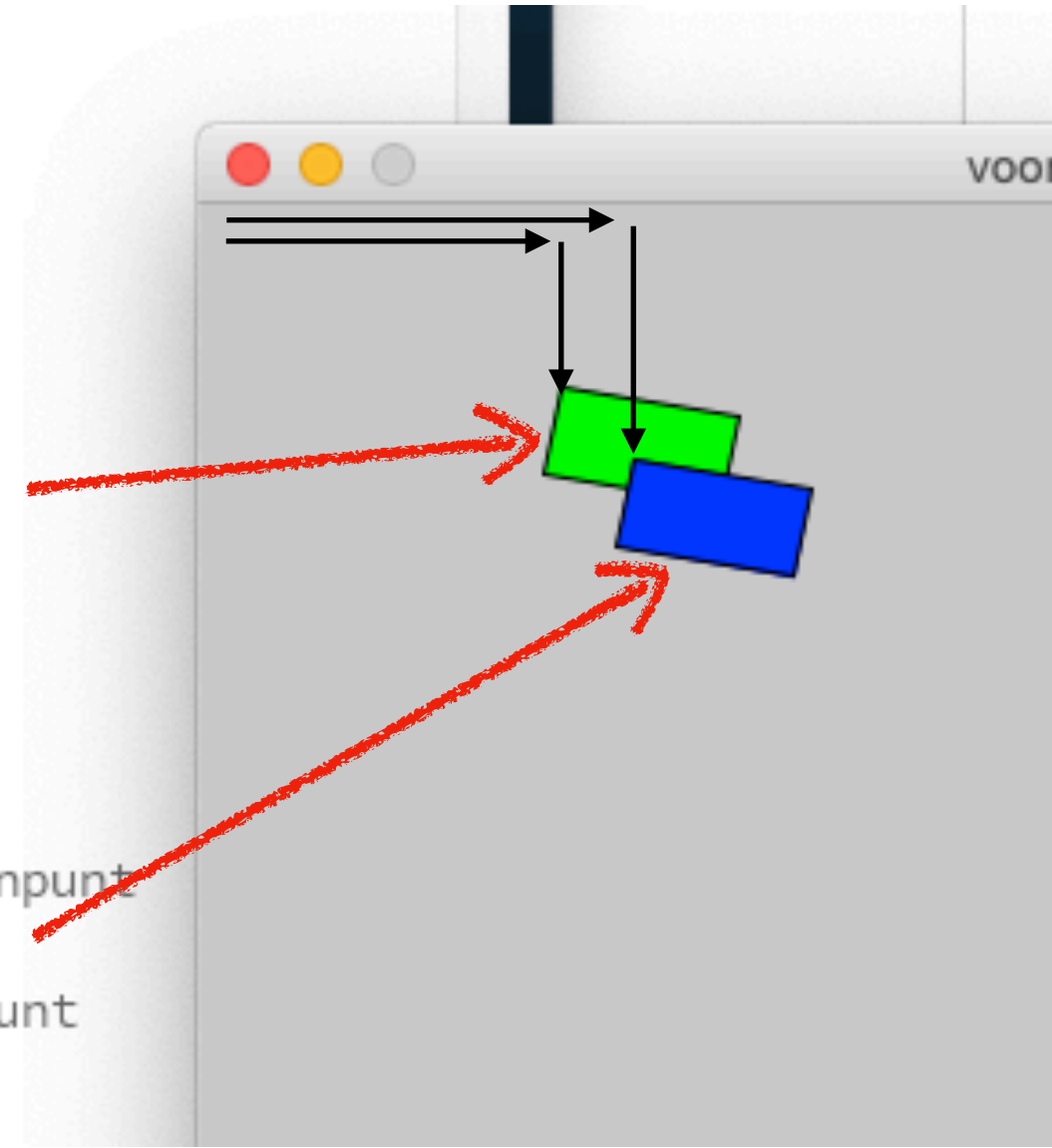
Hoe lossen we dit op?

- Door expliciet tegen de computer te zeggen dat de `translate()` en `rotate()` opdrachten alleen gelden voor een klein stukje code, niet alles dat er na komt ook nog.
 - Soort van `{ }` brackets bij een if-statement
 - Maar in plaats van brackets gebruiken we
 `pushMatrix()` als `{`
 en
 `popMatrix()` als `}`.

Stukje voor stukje

Ieder stukje code wordt 'apart' uitgevoerd

```
2
3 void setup() {
4   size(500,500);
5   fill(255,0,0);
6   pushMatrix();
7   translate(100,50); // beginpunt verleggen
8   rotate(PI/18.0); // rotate vanuit nieuw beginpunt
9   fill(0,255,0);
10  rect(0,0,50,25); // rect vanuit nieuw beginpunt
11  popMatrix();
12
13  pushMatrix();
14  translate(120,70); // beginpunt nogmaals verleggen
15  rotate(PI/18.0); // rotate vanuit nieuw nieuw beginpunt
16  fill(0,0,255);
17  rect(0,0,50,25); // rect vanuit nieuw nieuw beginpunt
18  popMatrix();
19 }
```



Regel 7-10 gelden alleen voor groene blokje, regel 14-17 gelden alleen voor blauwe blokje.

The return of parameters

- Parameters hebben we al een keer behandeld: bij het aanroepen stuur je waardes mee waarmee de functie aan het werk gaat.
- Principe werkt ook als de functie in een class staat.
 - Stappen:
 - bij aanmaken instance waardes meesturen. Net als bij een gewone functie staan die waardes tussen de haakjes achter de naam.
 - Parameter waardes in de class afvangen in de constructor functie.
 - Parameter waardes omzetten in globale variabelen.

The return of parameters

Oude situatie: waarden worden bepaald binnen class

```
voorbeeld_4  Boeing747 ▼
// voorbeeld loops, ArrayList en instances
// door Ton Markus En Vincent Booman, oktober 2019

ArrayList<boeing747> WatVliegtErNu; // geheugenruimte inruil

void setup() {
  WatVliegtErNu = new ArrayList<boeing747>(); // ruimte vu
  for (int t=0; t<5; t=t+1) { // loop
    WatVliegtErNu.add(new boeing747()); // voeg waarde toe
  }
  1: maak instance aan
}

void draw() {
  for (int i=0; i<WatVliegtErNu.size(); i=i+1) { // loop
    boeing747 huidigeVliegtuig = WatVliegtErNu.get(i);
    huidigeVliegtuig.veranderHoogte();
  }
}
```

```
voorbeeld_4  Boeing747 ▼
1 class boeing747 {
2   int mijnHoogte = 0;
3   int stijgSnelheid;
4
5   boeing747() { // de constructor functie
6     stijgSnelheid = int(random(10));
7   }
8
9   void veranderHoogte() {
10    mijnHoogte = mijnHoogte + stijgSnelheid;
11    println("Mijn hoogte is: " + mijnHoogte);
12  }
13
14 }
15
16
17
```

2: bepaal stijghoogte

The return of parameters

Nieuwe situatie: waarden worden bepaald bij aanmaken instance (centraal/overzicht)

```
voorbeeld_1  Boeing747 ▼
1 // voorbeeld parameters naar instances
2 // door Ton Markus En Vincent Booman, november 2019
3
4 ArrayList<boeing747> WatVliegtErNu; // geheugenruimte inruime
5
6 void setup() {
7   WatVliegtErNu = new ArrayList<boeing747>(); // ruimte vul
8   for (int t=0; t<5; t=t+1) { // loop
9     WatVliegtErNu.add(new boeing747(int(random(10)))); // v
10  }
11 }
12
13 void draw() {
14   int hoogGenoeg = 0;
15   for (int i=0; i<WatVliegtErNu.size(); i=i+1) { // loop
16     boeing747 huidigeVliegtuig = WatVliegtErNu.get(i);
17     huidigeVliegtuig.veranderHoogte();
18   }
19 }
```

1: maak instance aan inclusief stijghoogte

```
voorbeeld_1  Boeing747 ▼
1 class boeing747 {
2   int mijnHoogte = 0;
3   int stijgSnelheid;
4
5   boeing747(int snelheid_parameter) { // de const
6     stijgSnelheid = snelheid_parameter;
7   }
8
9   void veranderHoogte() {
10     mijnHoogte = mijnHoogte + stijgSnelheid;
11     println("Mijn hoogte is: " + mijnHoogte);
12   }
13
14 }
15 }
```

2: zet parameter om in globale variabele

Waarom?

Team spirit...

Op deze manier kunnen instellingen voor het spel op een centrale lokatie overzichtelijk aangemaakt/aangepast worden.

Want je bent niet alleen op de wereld. Anderen moeten op een eenvoudig manier met jouw code verder kunnen.

“cowboy coding culture”

<https://daringfireball.net/linked/2019/11/01/van-rossum-clever-code>

Lesopdracht

Laat een rechthoek roterend rond het midden van de rechthoek door het venster bewegen.

Hint 1: De positie van het blokje wordt niet meer bepaald door de cijfers in de `rect()` opdracht, maar door de cijfers in de `translate()`

Hint 2: `rectMode()`