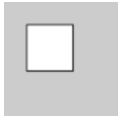


Basis van een Processing-script

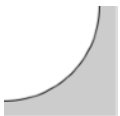
```
void setup() {    // één keer doen
}
```

```
void draw() {    // steeds opnieuw doen
}
```

Vormen



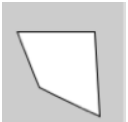
```
rect(20, 20, 40, 40);
// rechthoek
```



```
ellipse(0, 0, 80, 40);
// ronde vorm van 80 pixels
// breed en 40 hoog
```



```
line(20, 20, 80, 50);
// rechte lijn met beginpunt
// 20,20 en eindpunt 80,50
```



```
quad(152, 124, 344, 80, 276,
252, 120, 304);
// vierhoek
```

```
arc(50, 50, 80, 80, PI);    // open cirkel
curve();                    // gebogen lijnen
bezier();                   // nog meer gebogen lijnen
beginShape(), vertex() en endShape()
// complexe vormen
```

Kleur

```
background(255);    // achtergrondkleur
fill(100);           // inkleuren
noFill();            // niet inkleuren
```

```
stroke(30);          // kleur van randen
strokeWeight(5);     // dikte van randen
noStroke();          // geen randen
```

Voorbeelden:

```
fill(200);    // één getal is een grijswaarde
              // tussen zwart (0) en wit (255)
fill(255, 20); // twee getallen: grijswaarde
              // en doorzichtigheid
fill(255, 40, 200); // drie getallen: rood,
              // groen en blauw mengen
fill(255, 100, 100, 30); // rood, groen,
              // blauw en doorzichtigheid
```

Grootte van het Processing-scherm

```
size(200, 200); // grootte scherm opgeven
fullScreen();   // scherm beeldvullend maken
```

```
width // breedte van scherm als variabele
height // hoogte van scherm als variabele
```

Het midden van het scherm vind je door width en height door tweeën te delen:

```
rect(width/2, height/2, 40, 40);
```

Muis en toetsenbord

```
mouseX // horizontale positie van je muis
mouseY // verticale positie van je muis
```

Deze woorden kun je gebruiken in een if-statement of als functie:

```
mousePressed // 'true' als je op muis klikt
mouseDragged // 'true' als je muis over
              // het scherm sleept
mouseMoved   // als de muis beweegt
              // zonder dat de knop is ingedrukt
keyPressed   // 'true' als je typt
key          // variabele met de laatst getypte toets
```

Variabelen

```
int    // hele getallen, positief of negatief
float  // decimale getallen
boolean // waar/niet waar, 'true' of 'false'
string // een tekst
char   // een enkele letter
color  // een kleur
```

```
int getal = 10;
float variabele1 = 3.01;
```

Vragen stellen

```
if (voorwaarde) { .. }    // if-statement
```

```
if (aantal > 10) { // zijn het er meer dan 10?
  // zo ja, doe dan hier iets
}
```

```
<    // kleiner dan
>    // groter dan
==   // gelijk aan
!=   // niet gelijk aan
```

Iets meerdere keren doen

```
while (voorwaarde) { .. }    // while loop:
    blijf iets doen zolang de
    voorwaarde waar is
```

```
for (int i=0; i < 10; i++) { .. }    // for-loop
    (in dit voorbeeld): begin bij 0, ga
    door zolang we nog niet bij 10 zijn,
    en tel er steeds 1 bij op.
```

Steeds anders

```
random()    // kies een willekeurig getal
noise()     // willekeurig, maar wel steeds
            dichtbij het getal ervoor
frameCount // aantal beeldjes in de
            animatie tot nu toe
```

Het canvas verschuiven of draaien

```
float r = 0; // variabele
```

```
void draw() {
    background(51);
    translate(50, 50);
    rotate(r); // draaien
    rect(0,0,60,20);
```

```
// klein beetje verder draaien
r += 0.02;
}
```

```
rotate(0.1); // draai het canvas rond 0,0
```

```
translate(300, 300); // verplaats het
    beginpunt van het canvas (dus 0,0)
    van linksbovenin naar ergens
    anders (bijvoorbeeld het midden).
```

```
pushMatrix(); // onthou alle rotates en
    translates die hierna komen
popMatrix();  // maak alle rotates en
    translates ongedaan voor
    alles wat hierna komt.
```

Nummers passend maken met map()

Waarde (bijvoorbeeld de muispositie)
 ‘vertalen’ naar een ander bereik:

```
float h = map(mouseX, 0, width, 40, 300);
```

Afstand tot een bepaald punt met dist()

```
void draw() {
    // kleur de achtergrond lichter naarmate
    // je muis verder weg is van het midden.
    float d = dist(50, 50, mouseX, mouseY);
    background(d*2, d*2.5, d*0.8);
}
```

Tekst invoegen

```
tekst komt tekstSize(20);
hier        text("tekst komt", 6, 20);
            tekstSize(20);
            text("hier", 6, 70);
```

Plaatjes inladen

```
PImage foto; // variabele
```

```
void setup() {
    size(400,400);
    // laad een plaatje in dat in dezelfde map
    // staat als dit script
    foto = loadImage("foto.jpg");
}
```

```
void draw() {
    // laad het plaatje zien
    image(foto, 0, 0);
}
```

Golfbewegingen met sinus

Gebruik `sin()` en `cos()` om golvende veranderingen te krijgen:

```
float hoek = 0; // variabele
void draw() {
    background(0);
    // sinus gebruiken als grootte
    circle(50, 50, sin(hoek)*100);
    // hoek een klein beetje veranderen
    hoek += 0.02;
}
```

Meer informatie en inspiratie

Alle functies: processing.org/reference
 Mooie voorbeelden:

- processing.org/examples
- openprocessing.org
- reddit.com/r/processing
- generative-gestaltung.de/2/