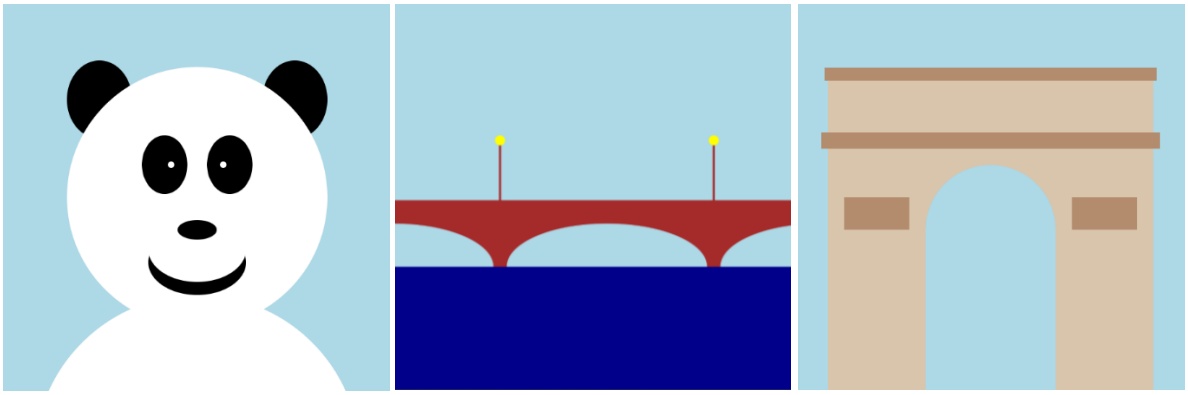
Intro
Hoi! Wij zijn Gemma, Max, Milène en Zoë van de opleiding ‘Communicatie en Multimedia-design’ in Maastricht. En je gaat met ons een wereldreis maken! Een wereldreis in code.

We maken tussenstops in Londen, New York, Rio de Janeiro, Hongkong en Parijs. Op elk van deze plekken ga je met behulp van code een tekening maken die past bij die plek. Hier zijn alvast wat voorbeelden die wij zelf gemaakt hebben:



Wij hebben er in elk geval al zin in, dus laten we maar snel vertrekken! Een hele nieuwe wereld van code wacht!

De basis van coderen
Wij gaan in deze workshop aan de slag met de codetaal JavaScript. Een codetaal lijkt eigenlijk heel erg op een ‘normale’ vreemde taal, zoals Frans of Spaans of Engels. Als je naar Frankrijk op vakantie gaat, zul je daar Frans moeten spreken zodat mensen je begrijpen. En wij moeten nu dus JavaScript tegen onze computer gaan ‘praten’ zodat onze computer ons begrijpt.

Maar JavaScript is nogal ingewikkeld, vooral als het je eerste codetaal is, dus is er een manier bedacht om JavaScript te gebruiken op een manier die redelijk begrijpbaar is. Die manier heet **p5.js**. En daar gaan wij mee werken.

Hieronder zie je hoe een p5.js-bestand eruitziet. Je ziet dat het bestaat uit cijfers, woorden en leestekens. Voor jou is deze taal nu nog moeilijk te lezen, maar de computer begrijpt het prima. Wat hieronder in code is geschreven, zijn instructies om een panda te tekenen – de panda die je net hebt gezien.

```
function setup() {
  createCanvas(600, 600);
}

function draw() {
  background("lightblue");

  //oren
  fill(0)
  ellipse(150,150,100,120)
  ellipse(450,150,100,120)

  //hoofd en lichaam
  noStroke()
  fill(255)
  ellipse(300,300,400,400)
  ellipse(300,700, 500,500)

  //ogen en mond
  fill(0)
  ellipse(250, 250, 70, 90)
  ellipse(350, 250, 70, 90)
  ellipse(300, 400, 150, 100)

  //mond vulling en pupillen
  fill(255)
  ellipse(300,380,150,100)
  ellipse(260,250, 10,10)
  ellipse(340,250,10,10)

  //neus
  fill(0)
  ellipse(300, 350, 60, 30)
}
```

In p5.js is volgorde heel belangrijk. De computer 'leest' de code van boven naar beneden; wat er bovenaan je code staat, wordt als eerste getekend. Dat klinkt misschien lastig, maar er zijn veel dingen waarbij volgorde belangrijk is. Als je op reis gaat, bijvoorbeeld, is volgorde ook belangrijk:

- Kiezen waar je heen gaat reizen
- Vliegtickets bestellen
- Koffer inpakken
- Naar het vliegveld rijden
- In het vliegtuig stappen
- Uitstappen op je bestemming
- Aan het strand gaan liggen

Je kunt dit lijstje niet van beneden naar boven lezen. Je kunt niet aan het strand gaan liggen als je nog niet bij het strand bent, en je kunt niet in het vliegtuig stappen als je nog geen vliegtickets hebt (niet legaal, in elk geval), en het wordt moeilijk om vliegtickets te bestellen als je nog niet weet waar je heen gaat.

P5.js werkt ook zo, alleen ziet een stappenplan er dan bijvoorbeeld zo uit:

- Maak een canvas (een canvas kun je zien als een vel papier waarop je straks je tekeningen gaat maken)
- Geef aan hoe groot het canvas moet zijn
- Kies een kleur voor het canvas
- Kies een kleur die je voor je vormen gaat gebruiken
- Teken een vorm
- Geef aan waar je je vorm gaat tekenen en hoe groot je vorm gaat zijn

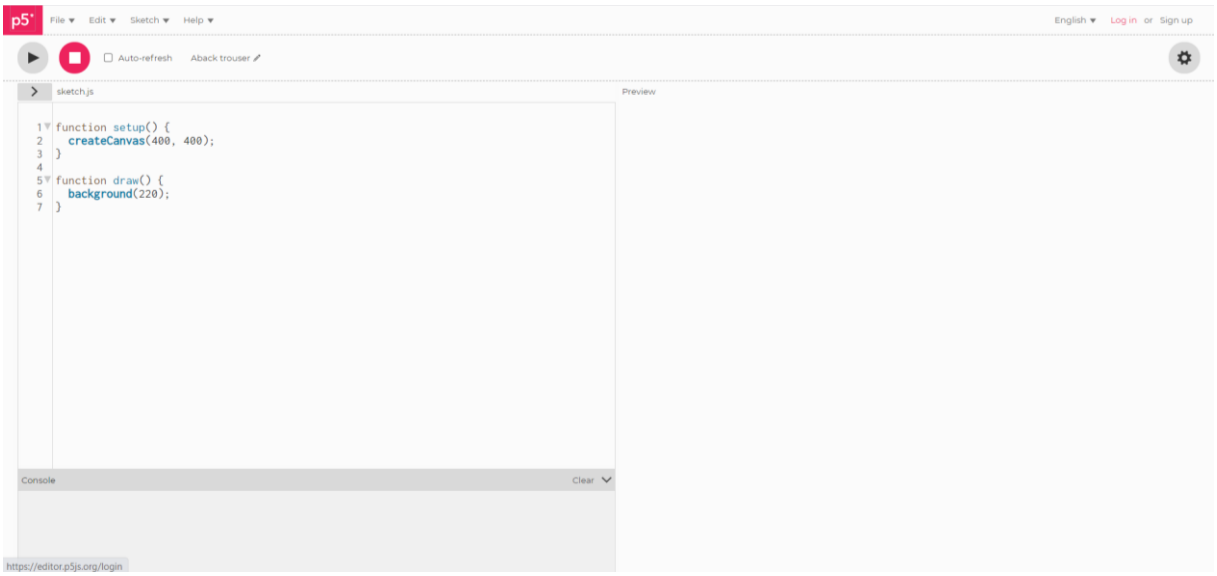
Ook dit lijstje kun je niet van beneden naar boven lezen. Je kunt de positie en grootte van je vorm niet aangeven als je vorm nog niet bestaat, en je kunt je vorm überhaupt niet tekenen als je geen canvas hebt om het op te tekenen.

Nu dat je dat weet, kunnen we aan de slag om zelf code te gaan schrijven!

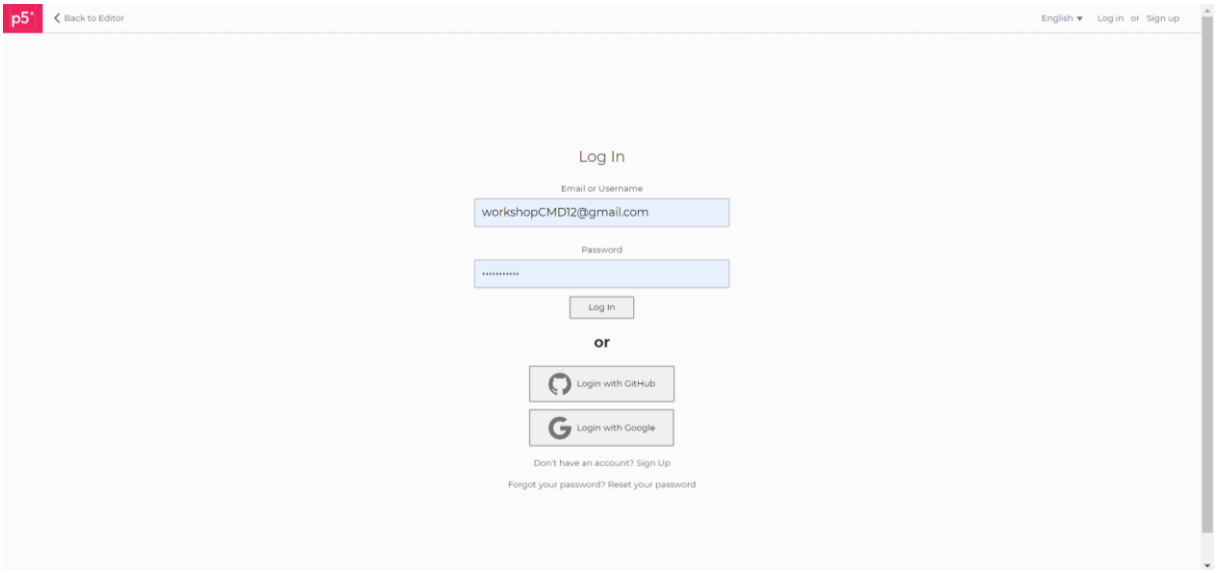
Een eerste p5.js-bestand

Voor het aanmaken van een p5.js-bestand ga je naar **https://editor.p5js.org/** . Daar staat een bestand al voor je klaar, maar voordat we er iets mee gaan doen, willen we dat je inlogt – zo kun je je

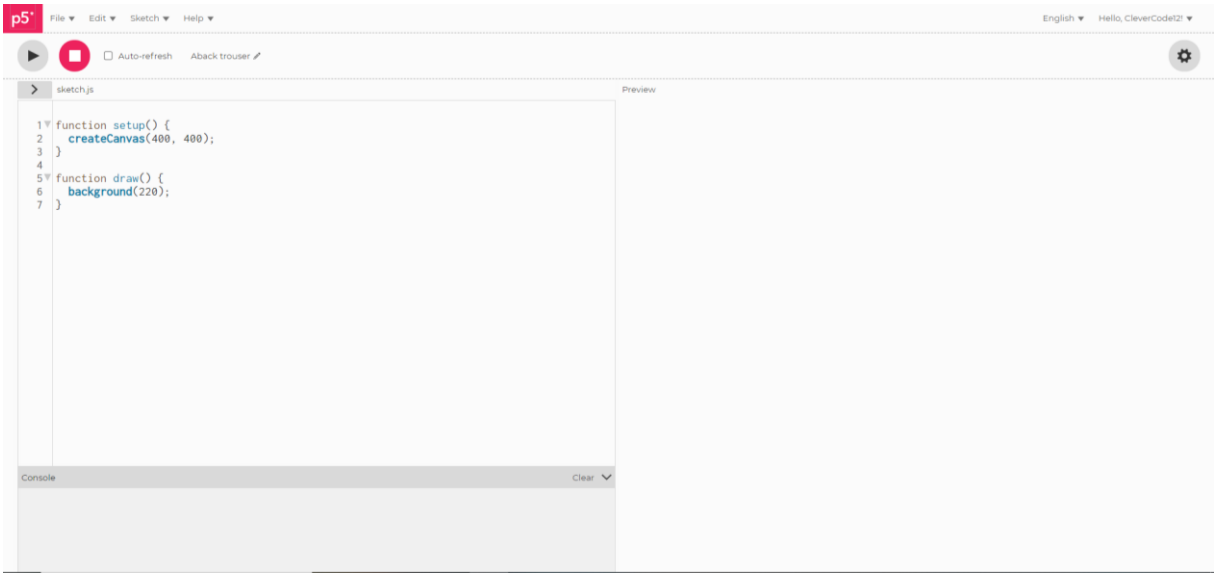
werk straks opslaan. We hebben een account aangemaakt waarin de hele klas gaat werken. Je vindt de ‘log in’-knop rechtsboven in het scherm.



Je logt in met de e-mail **workshopCMD12@gmail.com** en het wachtwoord **workshopcmd** .



Als je ingelogd bent, staat het bestand automatisch voor je klaar. Als het goed is, ziet dat bestand er zo uit:



Zoals je ziet, is de code verdeeld in twee zogenoemde ‘functions’: **setup** en **draw**. Alle code in setup wordt één keer uitgevoerd. Alle code in draw wordt zestig keer per seconde uitgevoerd. Het is voornamelijk belangrijk om draw te gebruiken als je iets maakt wat beweegt, maar het is geen probleem als je het gebruikt voor stilstaande vormen. Aangezien wij vooral tekeningen gaan maken die stilstaan, maakt het eigenlijk niet uit of je je code in setup of draw zet, zolang je je code maar in één van de twee zet. In de uitleg zetten we de code in de draw, maar stiekem is het ook prima als je al je code in de setup zet.

! De setup en draw beginnen bij ‘{‘ en eindigen bij ‘}’. Die haakjes heten accolades. Als je iets in de setup of de draw wilt typen, typ je het dus ergens *na* de ‘{‘ en *voor* de ‘}’. Alles wat buiten de accolades getypt wordt, werkt niet.

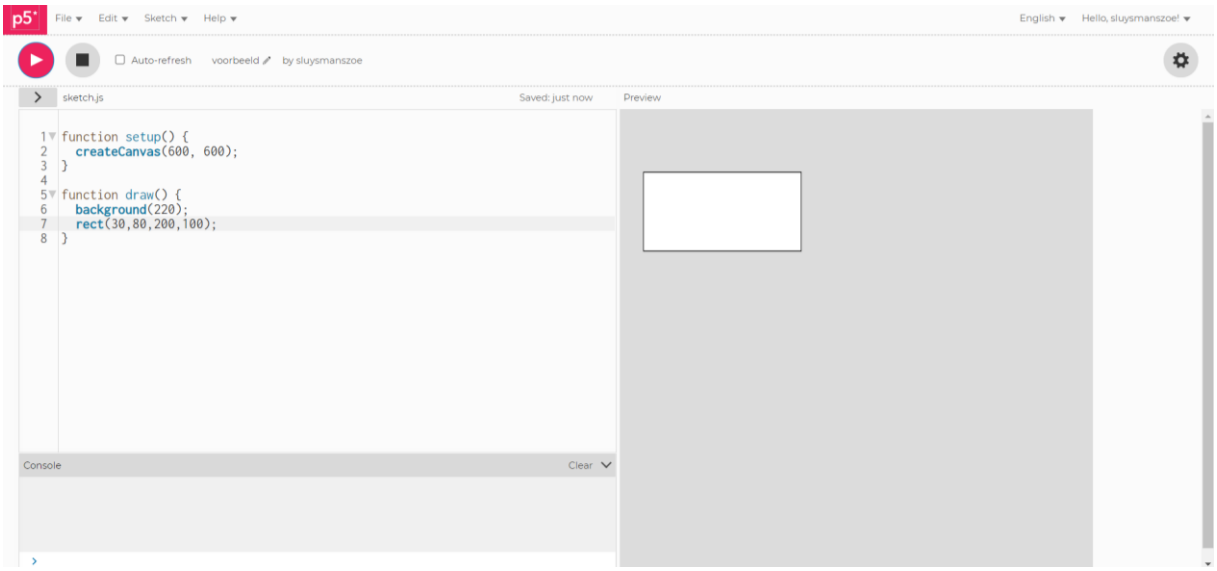
Links van de regels code zie je de regelnummers staan. Op regel 2 staat **createCanvas**. Tussen de haakjes staat ‘400, 400’. Dit is de grootte van het canvas, opgeschreven in pixels. Een pixel is een gekleurd puntje op je beeldscherm – alle puntjes bij elkaar vormen het beeld. Als je iets maakt in code, meet je dingen meestal in pixels.

Het canvas is automatisch 400 pixels breed en 400 pixels hoog. Voor onze tekeningen werken we met een canvas van 600 bij 600 pixels. Kun je de code zo aanpassen dat het canvas 600 pixels breed en 600 pixels hoog is?

! Door op de play-knop linksboven op het scherm te klikken, voer je je code uit.

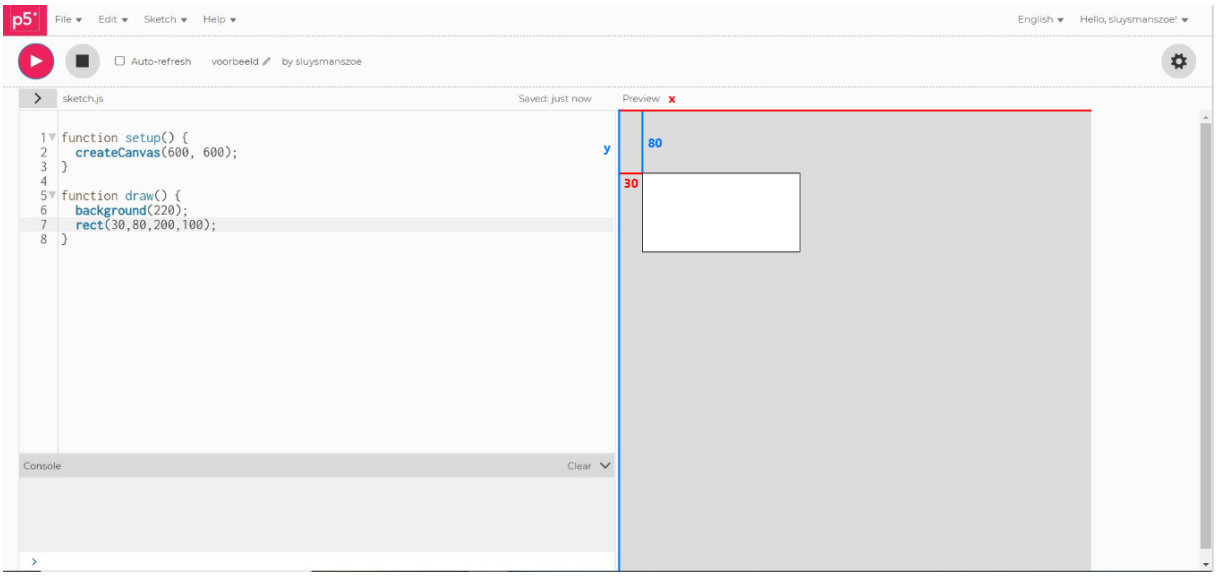
Coördinaten

Gefeliciteerd – je canvas is klaar voor gebruik! Nu kunnen we onze eerste vorm gaan tekenen. Voor onze eerste vorm gaan we een rechthoek (‘rectangle’ in het Engels, of **rect** in code) tekenen.



Zoals je ziet, staan er bij ‘rect’ vier getallen tussen de haakjes. De eerste twee getallen bepalen de locatie van de rechthoek. Het eerste getal bepaalt hoeveel pixels de rechthoek van de *linkerkant* van het canvas af is. Dit noemen we ‘**x**’. Deze rechthoek staat dus 30 pixels van de linkerkant van het canvas af.

Het tweede getal bepaalt hoeveel pixels de rechthoek van de *bovenkant* van het canvas af is. Dit noemen we ‘**y**’. Deze rechthoek staat 80 pixels van de bovenkant van het canvas af.

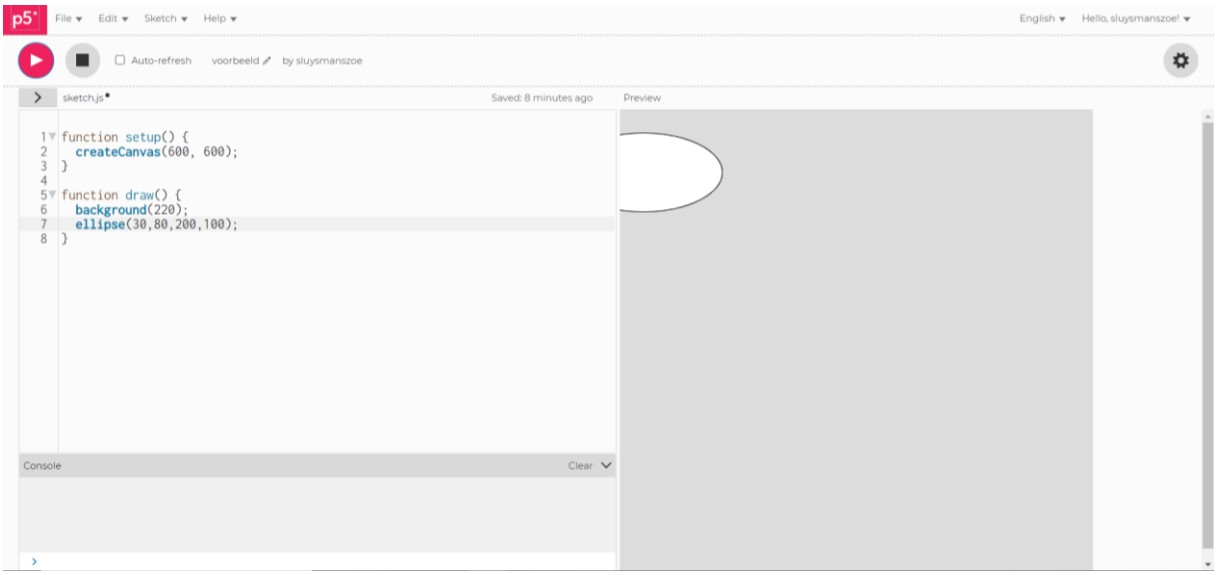


De laatste twee getallen bepalen hoe groot de rechthoek is: deze rechthoek is 200 pixels breed en 100 pixels hoog. Kort samengevat werkt een ‘rect’ dus zo: rect(x, y, breedte, hoogte).

Vormen

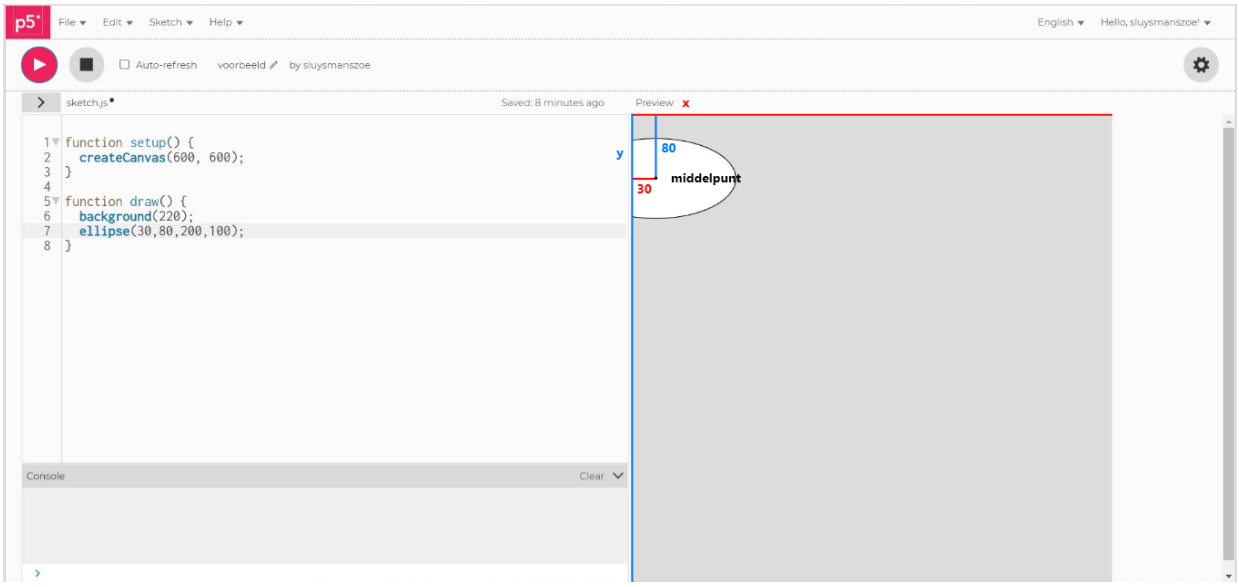
Rechthoeken zijn niet de enige vormen die je kunt tekenen. In onze workshop komen ook lijnen (**line**) en cirkels (**ellipse**) aan bod.

Een cirkel werkt hetzelfde als een rechthoek op het gebied van getallen: de getallen van een cirkel zijn ook ellipse(x, y, breedte, hoogte). Maar er is wel een belangrijk verschil.

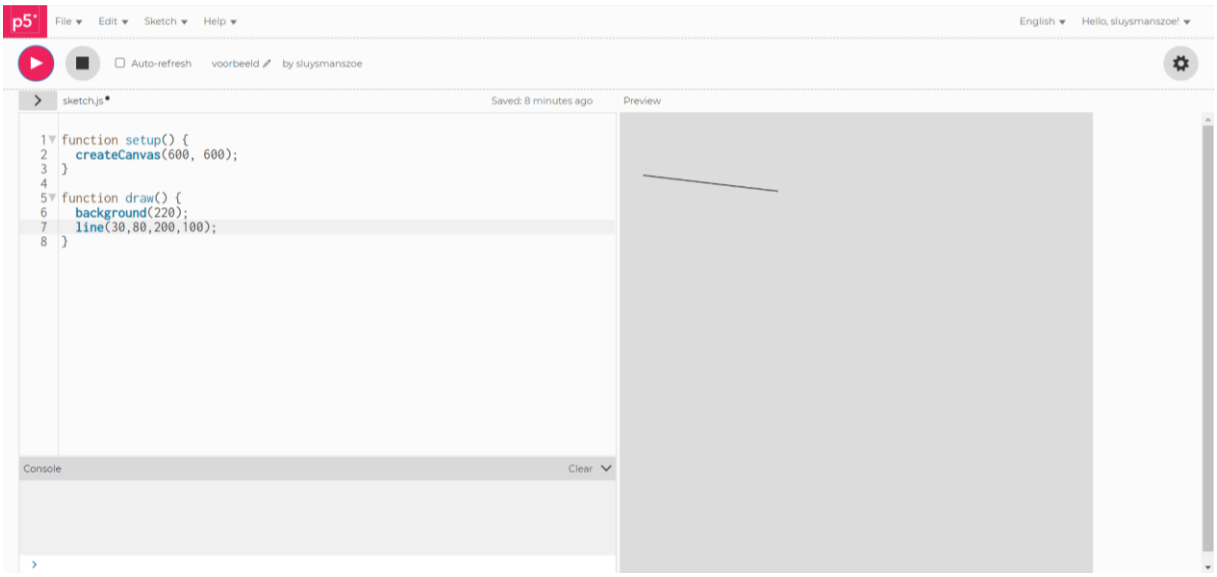


Deze cirkel heeft exact dezelfde getallen als de rechthoek, maar hij staat op een andere plek. De eerste twee getallen van ‘ellipse’ geven namelijk aan waar het *middelpunt* van de cirkel zich bevindt. Terwijl de eerste twee getallen van ‘rect’ aangeven waar de *linkerbovenhoek* van de rechthoek zich bevindt.

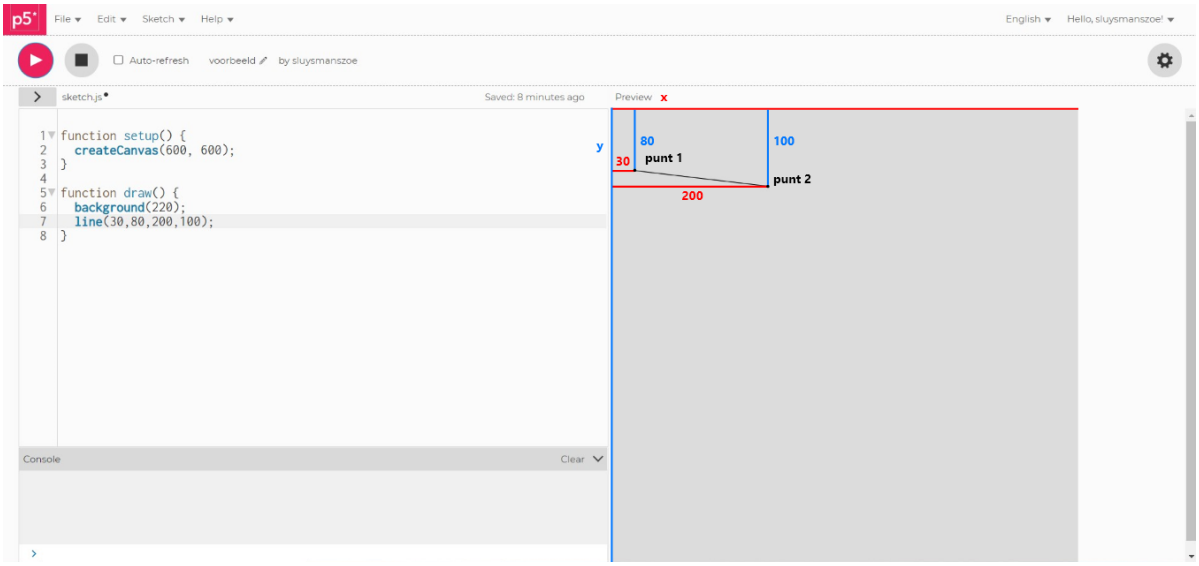
Het middelpunt van deze cirkel is dus 30 pixels van de linkerzijkant en 80 pixels van de bovenkant van het canvas af. Dit is belangrijk om te weten als je straks met cirkels gaat werken.



Een ‘line’ werkt anders: deze bestaat eigenlijk uit twee verschillende punten die door een lijn zijn verbonden. Jij tekent dus de twee uiteinden van de lijn; de computer tekent de lijn zelf.



De eerste twee getallen bepalen waar het *eerste punt* zich bevindt; in dit geval dus op 30 pixels afstand van de linkerkant van het canvas en op 80 pixels afstand van de bovenkant van het canvas. En de laatste twee getallen bepalen waar het *tweede punt* zich bevindt. In dit geval dus op 200 pixels afstand van de linkerkant van het canvas en op 100 pixels afstand van de bovenkant.



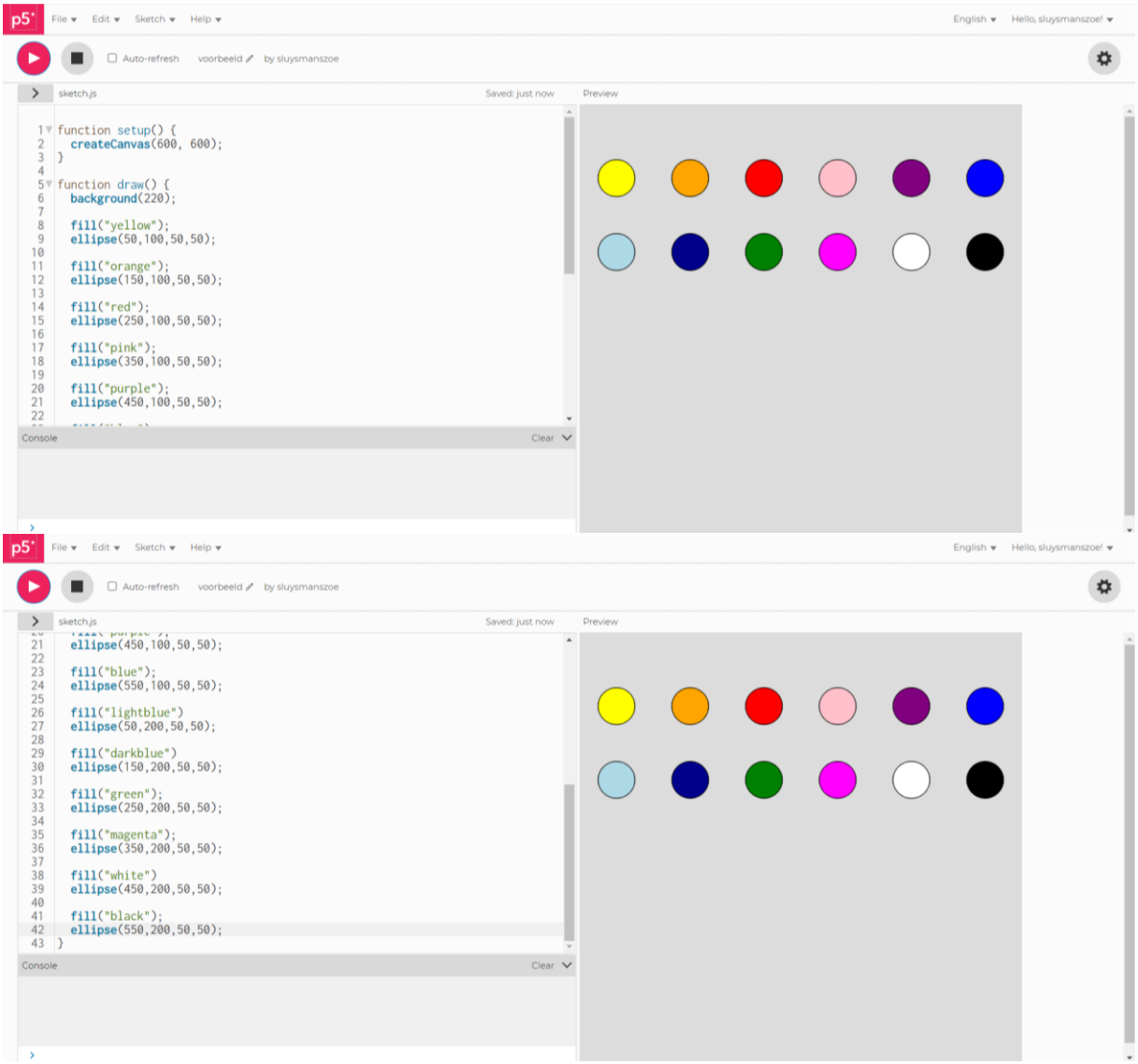
Achterin het boekje vind je nog wat voorbeelden van hoe je vormen kunt gebruiken.

Kleuren

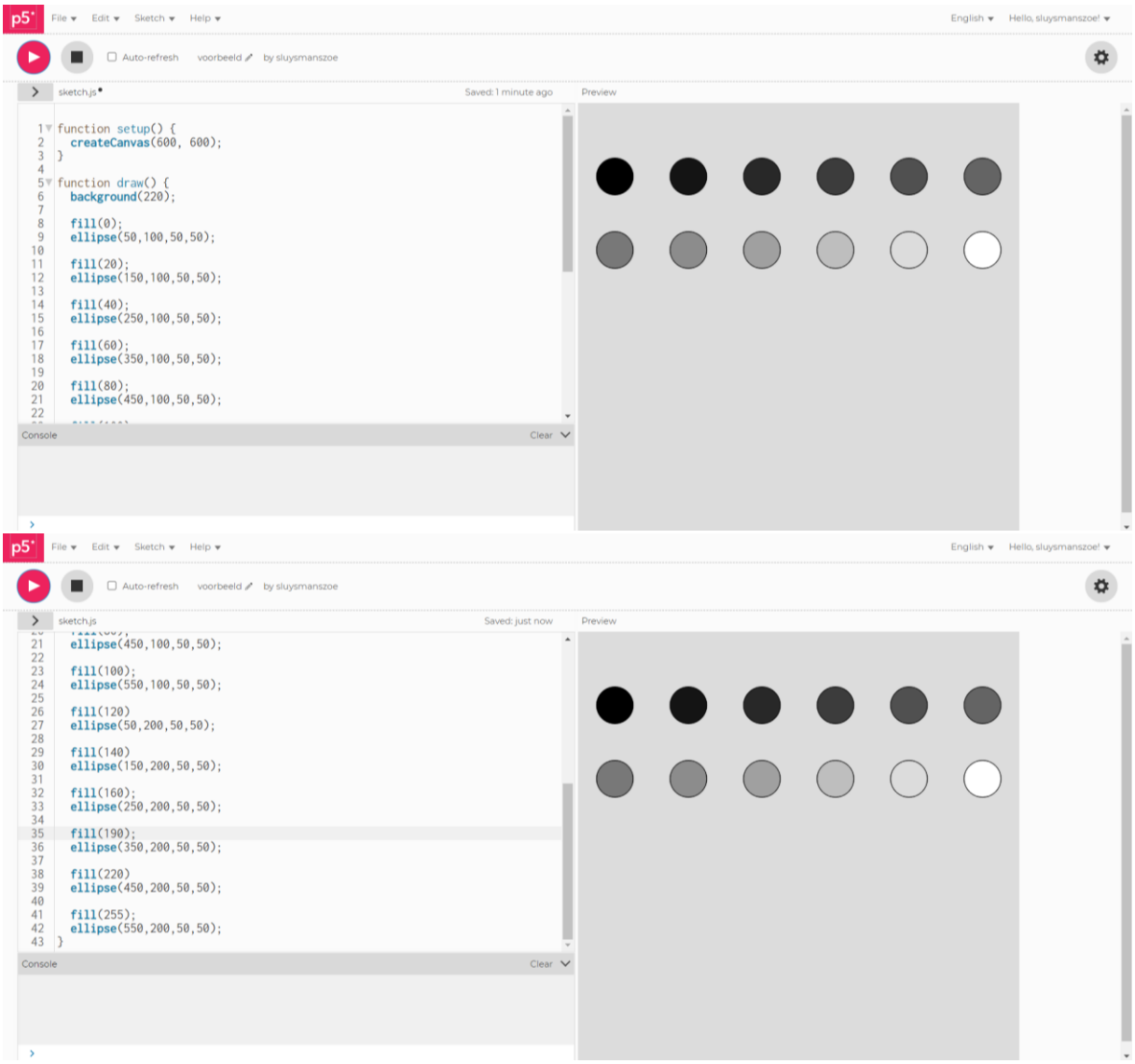
Tot nu toe hebben we alleen maar saaie witte vormen getekend; het is tijd voor wat kleur!

Bij een vorm breng je een kleur aan met **fill**. Bij de rand van een vorm of bij een line breng je een kleur aan met **stroke**. Er zijn drie manieren om kleur te gebruiken in p5.js: met woorden, met grijstinten en met rgb-kleurcodes.

Woorden zijn het gemakkelijkst te gebruiken, maar de mogelijkheden zijn beperkt. Je kunt alleen de kleuren gebruiken die al voorgeprogrammeerd zijn in p5.js. Het is belangrijk om de kleuren tussen dubbele aanhalingstekens te plaatsen. Hieronder zie je een voorbeeld van wat verschillende kleuren.

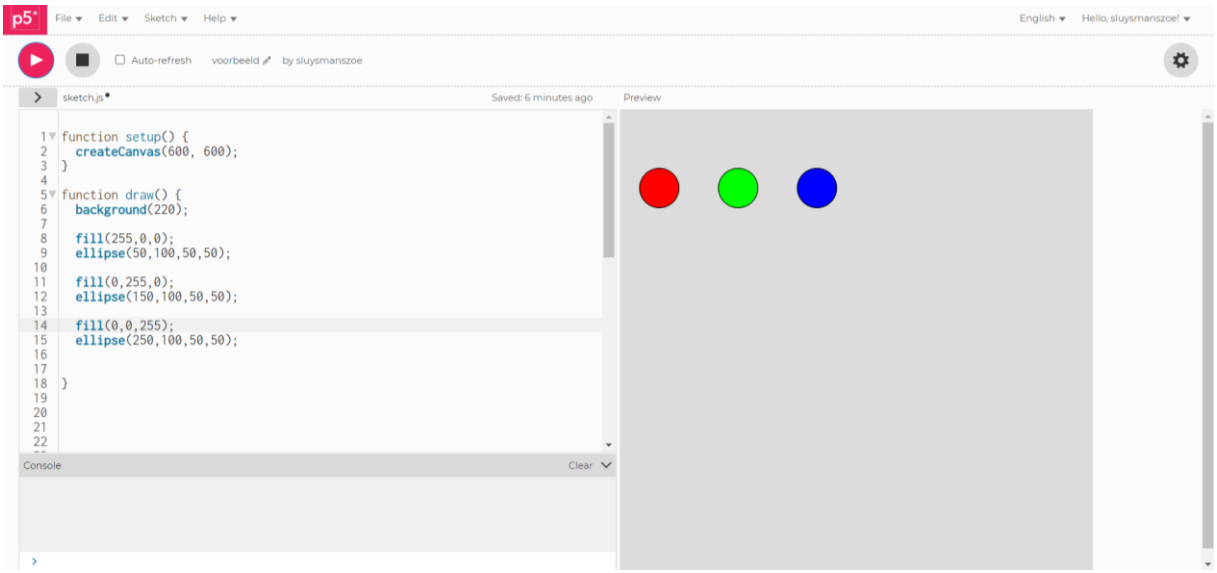


Grijstinten zijn wat lastiger; om een grijstint te gebruiken, typ je een getal in plaats van een kleur. 0 betekent zwart en 255 betekent wit – hoe hoger het getal, hoe lichter de kleur grijs. Hieronder staan wat voorbeelden.

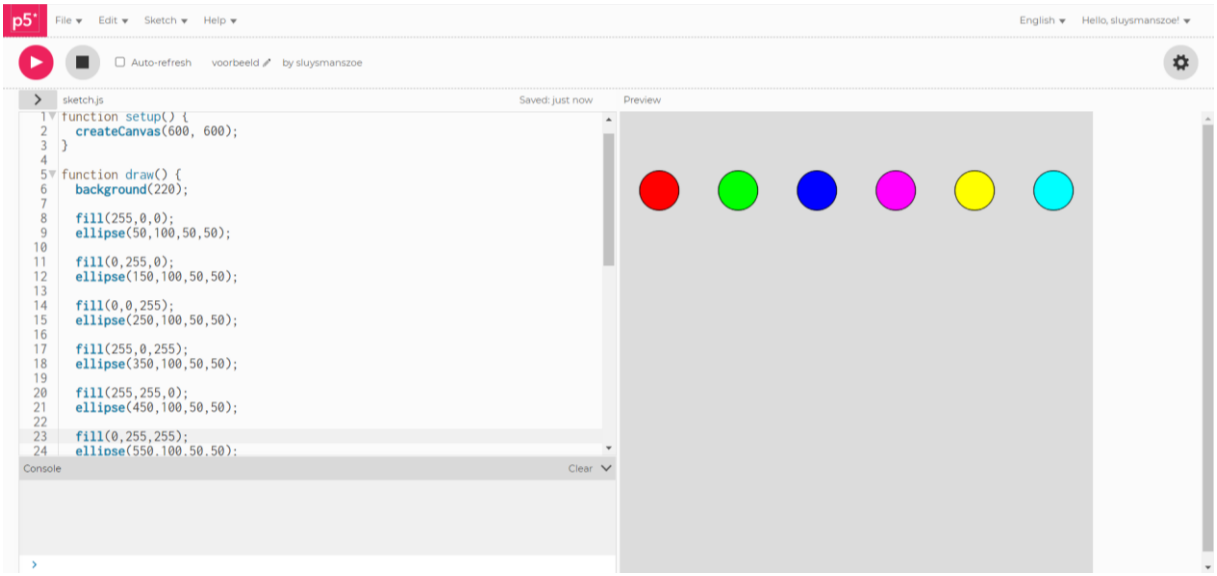


Met **rgb** gaan we in onze workshop niet per sé aan de slag – je kunt de workshop prima maken zonder rgb-codes te gebruiken – maar als je specifieke kleuren wilt gebruiken, is rgb misschien handig om te gebruiken. De afkorting ‘rgb’ staat voor ‘red, green, blue’. Een rgb-code bestaat uit drie getallen: het eerste getal staat voor de hoeveelheid rood, het tweede getal voor de hoeveelheid groen, en het derde getal voor de hoeveelheid blauw.

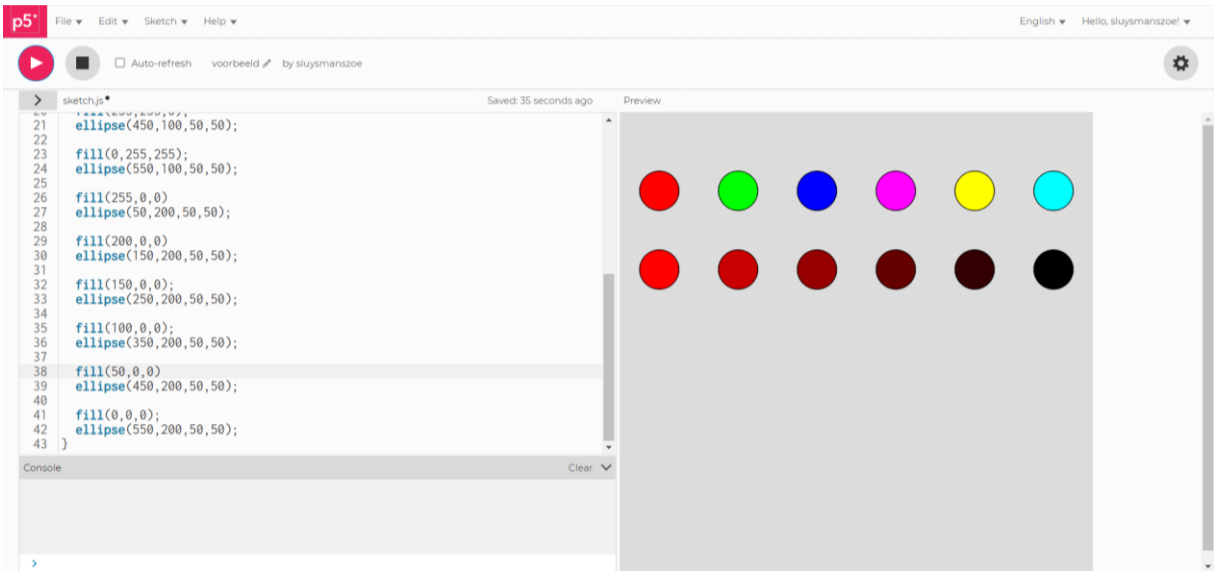
Een hoeveelheid is een getal tussen 0 en 255. 0 betekent dat je niets van die kleur gebruikt. Hieronder zie je hoe je rood, groen en blauw maakt.



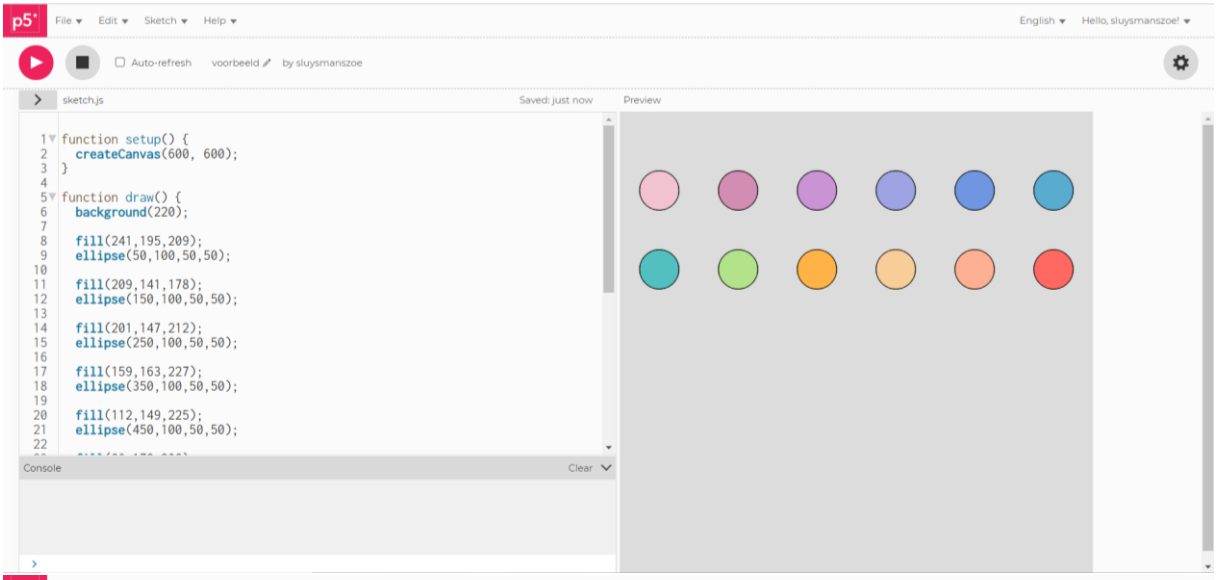
Het grote voordeel van rgb-codes is dat je kleuren kunt mengen. Blauw en rood maken samen paars. Blauw en groen maken samen turquoise. En rood en groen maken samen geel (dat klinkt onlogisch, maar helaas is het zo).

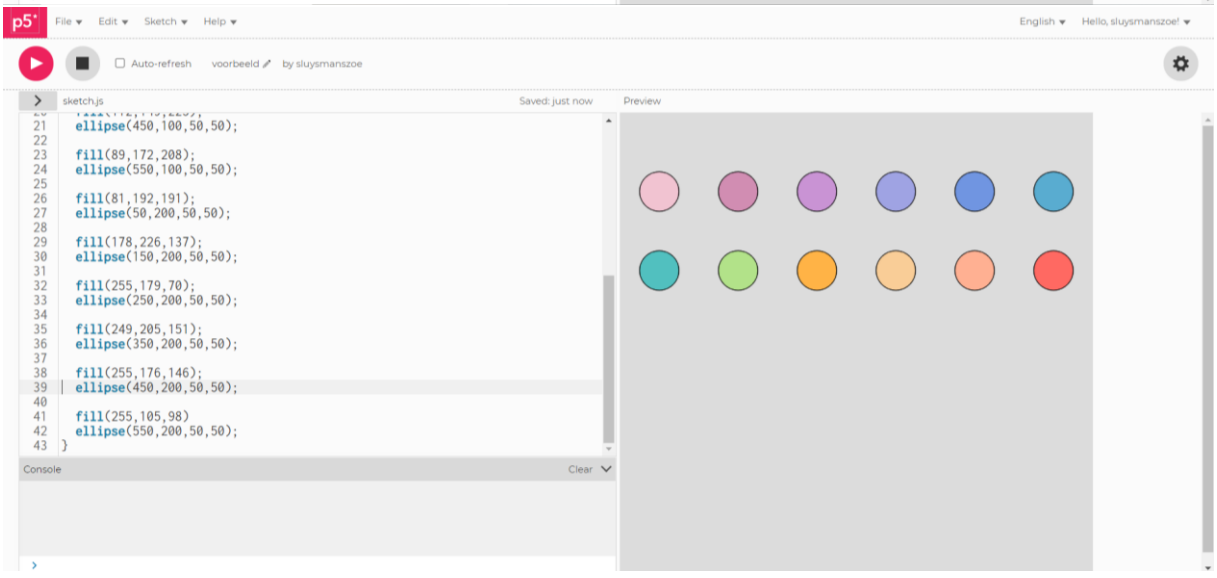


Nog een groot voordeel van rgb is dat je met de felheid van kleuren kunt spelen. 255 is het felste dat een kleur kan worden, maar hoe lager het getal, hoe donkerder de kleur. Hieronder is dat gedemonstreerd met rood, maar het kan met alle kleuren.



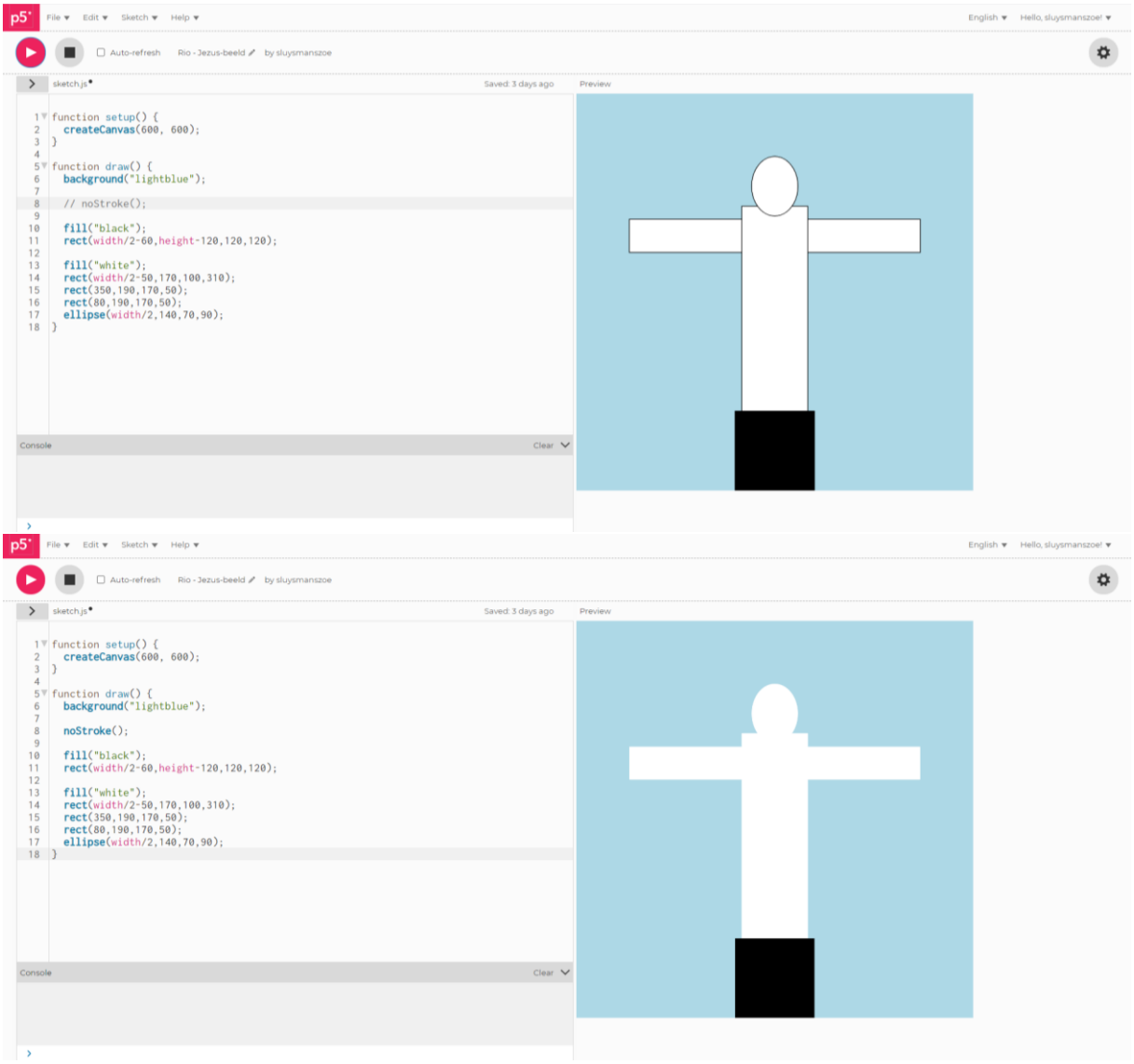
Hieronder nog wat voorbeelden van mooie kleuren die je misschien leuk vindt om te gebruiken. Door de getallen lager of hoger te maken, kun je zelf de kleuren aanpassen aan je eigen smaak!





Final touches
Je kunt nu intussen mooie vormen tekenen en die vormen een leuke kleur geven, maar misschien ziet je tekening er nog steeds niet uit zoals je wilt dat hij eruitziet. Misschien vind je de randen om de vormen heen niet mooi, of misschien wil je je vorm doorzichtig maken. Ook daar bestaan gemakkelijke oplossingen voor!

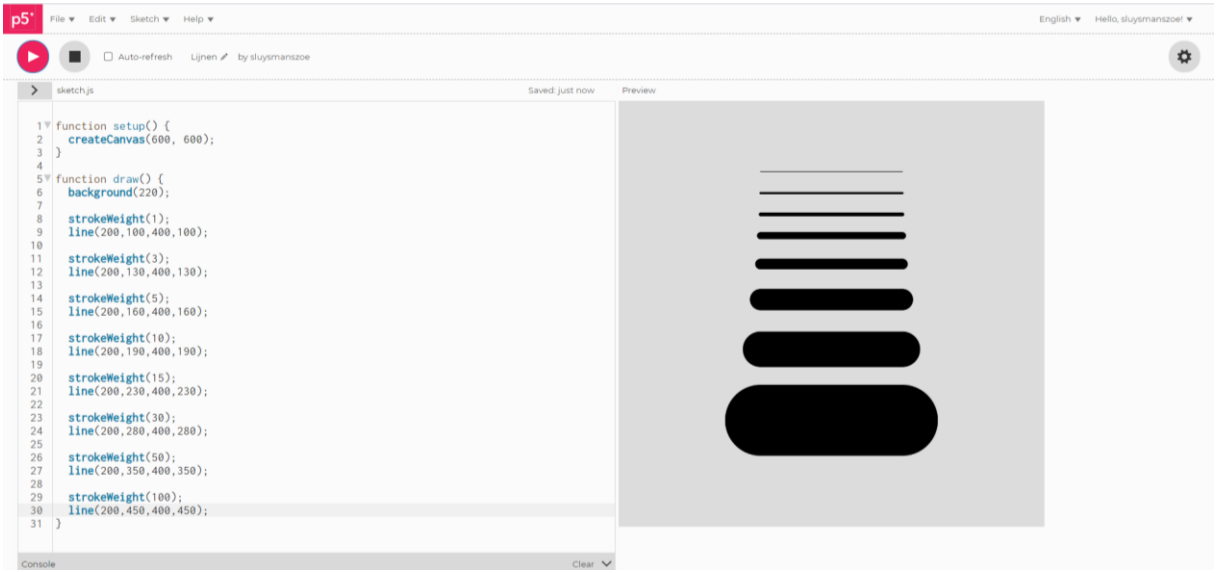
In onze ervaring is **noStroke** de optie die je het vaakste zult gebruiken. noStroke haalt de randen om een vorm weg. Vooral als je meerdere vormen wilt gebruiken om één tekening te maken, is dit handig. Hieronder zie je twee versies van dezelfde tekening; eerst zonder noStroke, daarna met noStroke.



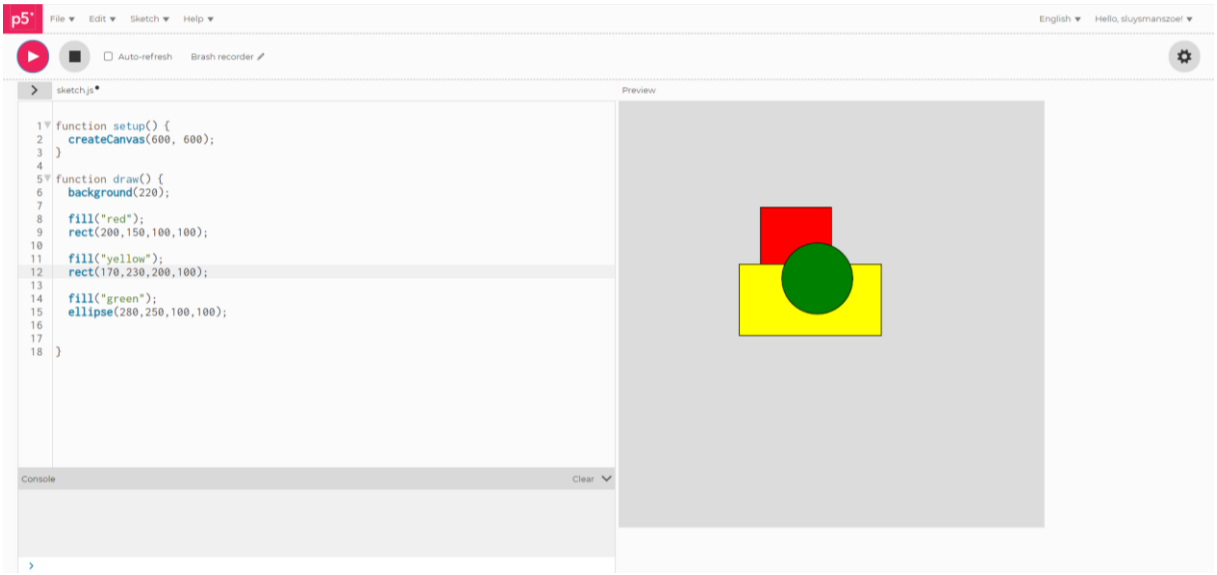
! Met noStroke moet je wel oppassen dat het ‘line’s ook laat verdwijnen. Zorg er dus voor dat je ‘line’s in de code *boven* je noStroke komen te staan.

! De ‘//’s die je in het eerste screenshot voor ‘noStroke’ ziet staan, betekenen dat die regel code commentaar is. Als er twee schuine strepen voor de code staan, wordt die regel code niet uitgevoerd. Zo kun je dus ook opmerkingen opschrijven zonder dat de computer er iets mee doet!

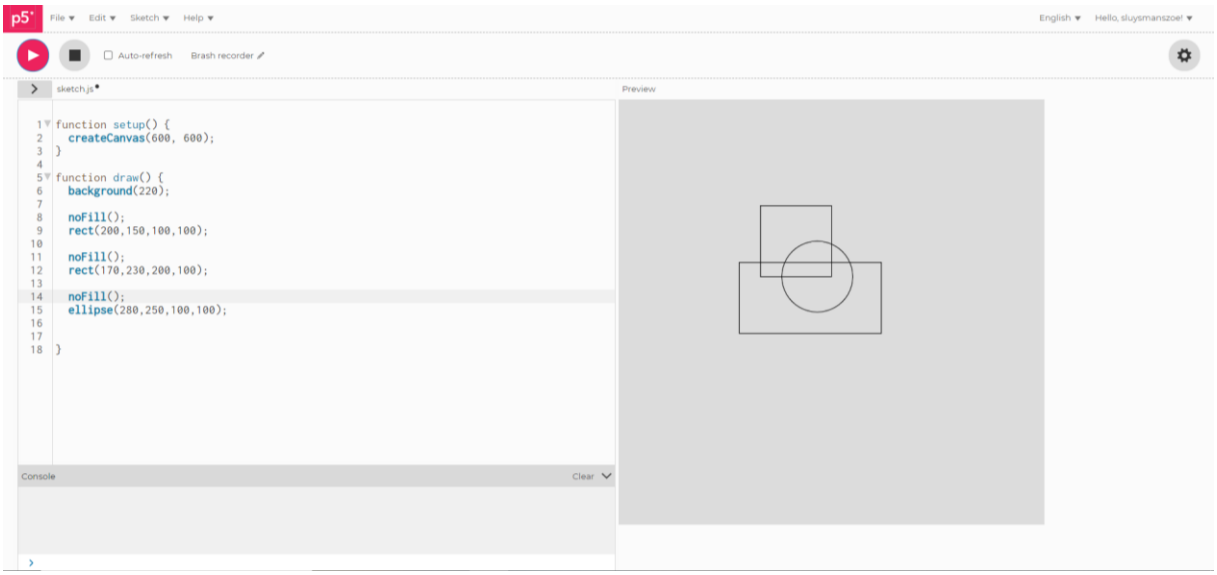
Met **strokeWeight** geef je de dikte van je rand of line aan, in pixels. Standaard staat de strokeWeight op 1 pixel. In onze workshop wordt de strokeWeight meestal niet hoger dan 5 pixels, maar je kunt de strokeWeight zo hoog maken als je wilt. Hieronder wat voorbeelden.



Met **noFill** kun je de vulling van een vorm weghalen. Dit is handig als je alleen de rand van een vorm over wilt houden.



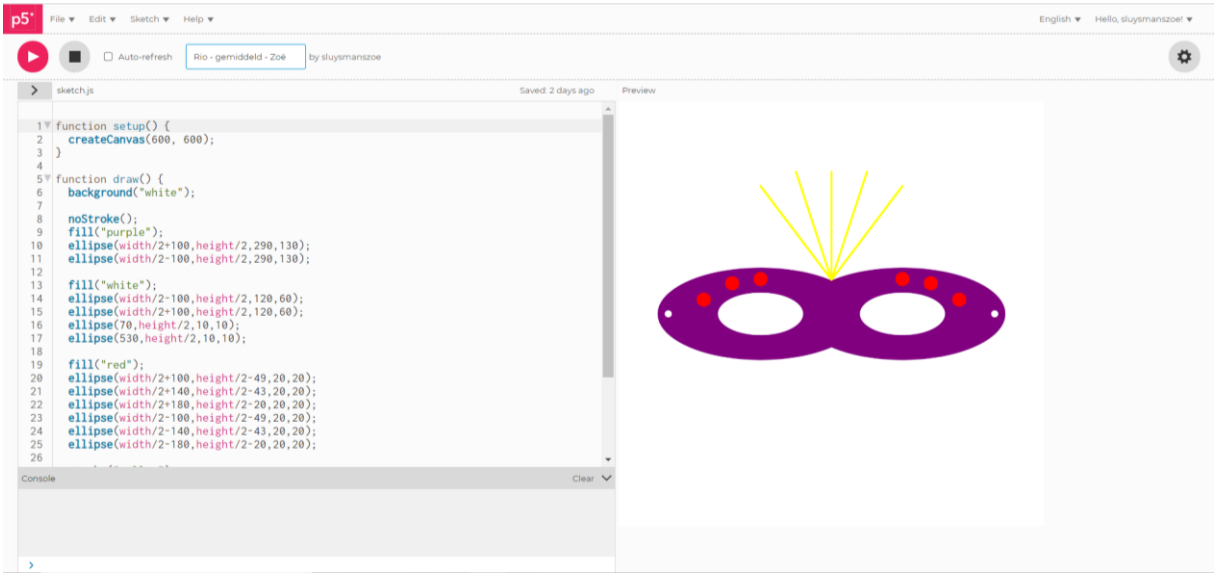
Hier zijn drie vormen met een fill…



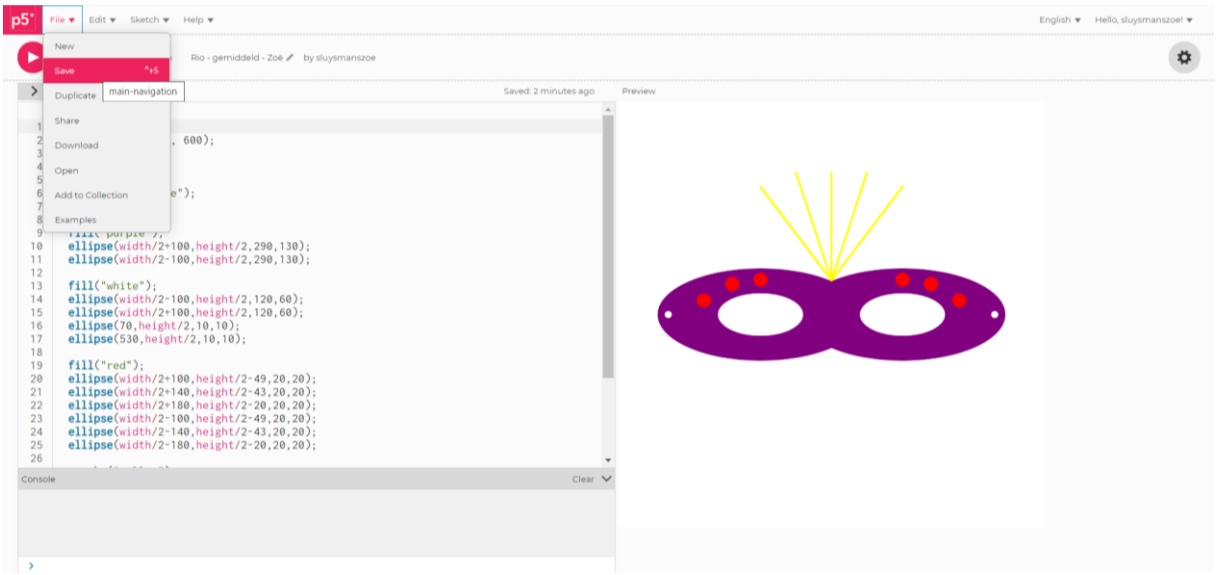
…en hier zijn diezelfde vormen zonder fill.

Je code opslaan

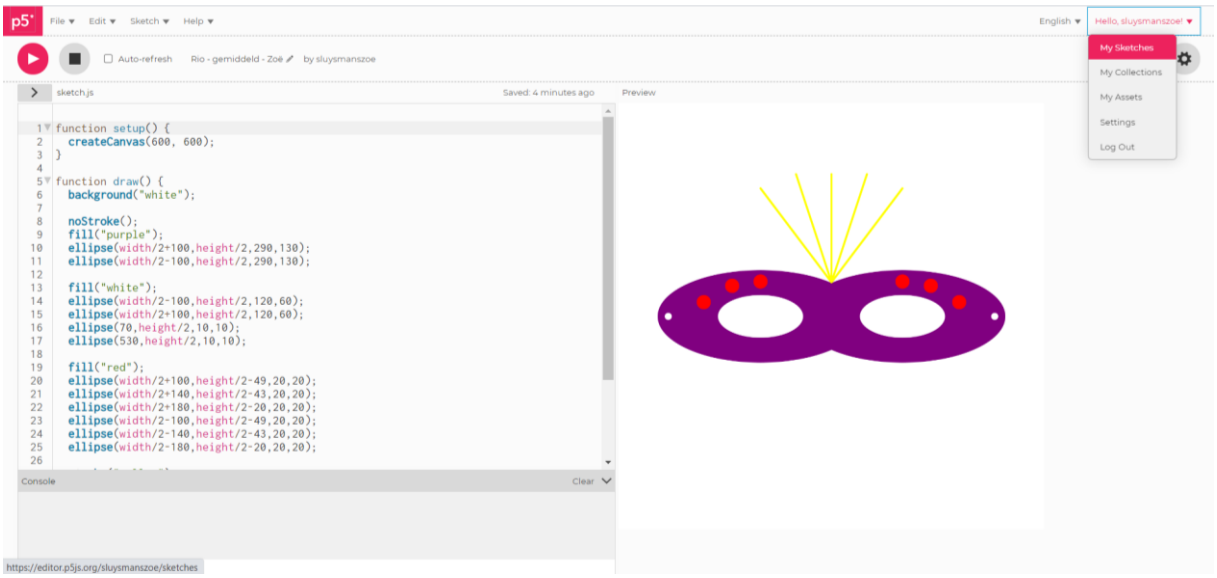
Als je je tekening af hebt, wil je ‘m natuurlijk niet kwijtraken. (Natuurlijk wil je ‘m niet kwijtraken! Hij is vast supermooi!) Om je tekening op te kunnen slaan, moet je hem eerst een titel geven. In deze workshop willen we graag dat je je tekening opslaat met de naam van de stad, de moeilijkheidsgraad en jouw naam. Dus als je de gemakkelijke tekening van Londen hebt gemaakt, is je titel: ‘Londen – gemakkelijk – jouw naam’.



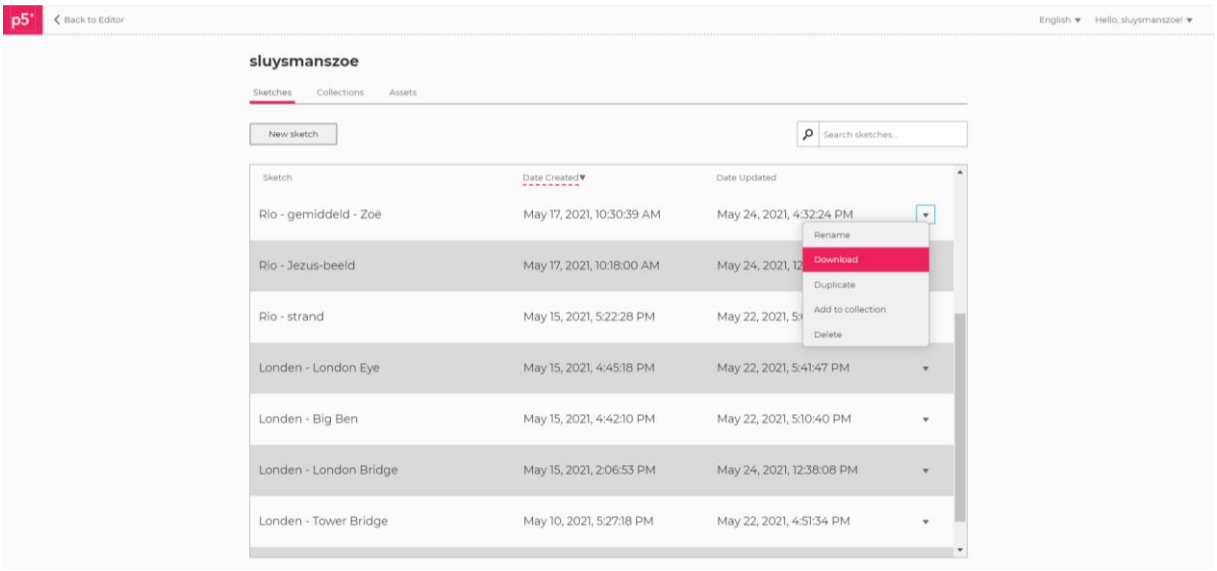
Het is aan te raden om je tekening een titel te geven zodra je begint en tussendoor op te slaan. Linksboven kun je de knop ‘file’ uitklappen en kun je daaronder op ‘save’ klikken. Dan wordt je tekening opgeslagen.



Als je tekening helemaal af is, kun je ‘m downloaden. Om dat te doen, klap je de knop rechtsboven uit en klik je op ‘My Sketches’.

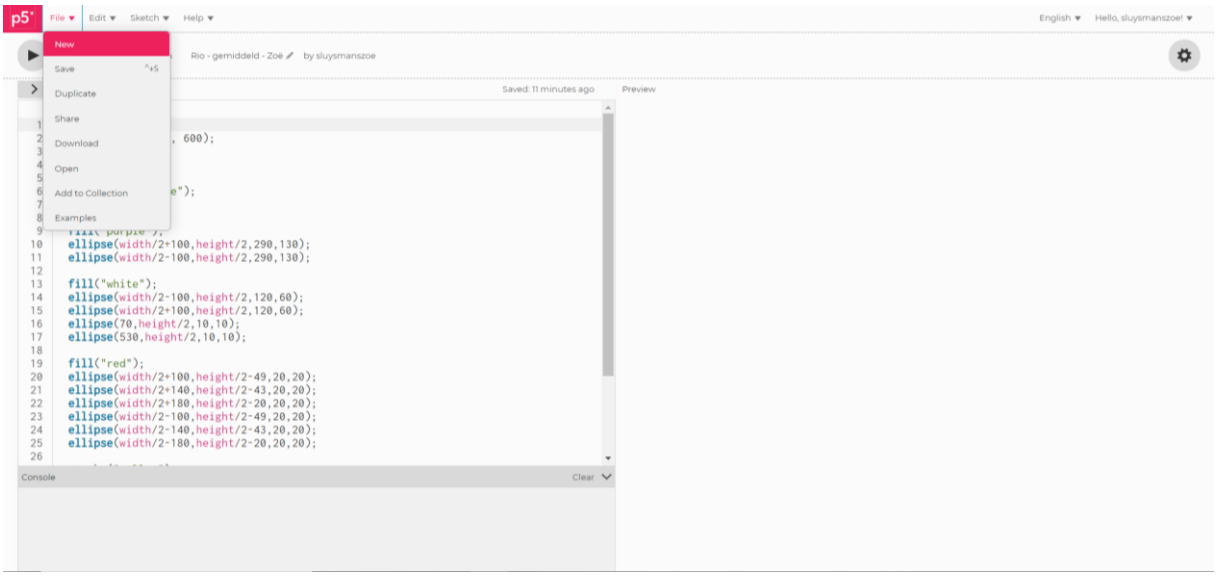


Je komt dan uit bij de plek waar de bestanden van de klas staan opgeslagen. Je zoekt het bestand met de juiste stad en je eigen naam. (Het nieuwste bestand staat bovenaan, dus je bestand zou redelijk bovenaan moeten staan.) Als je rechts op het pijltje klikt bij dat bestand, klapt er een schermpje uit. Daar kun je op ‘Download’ klikken.



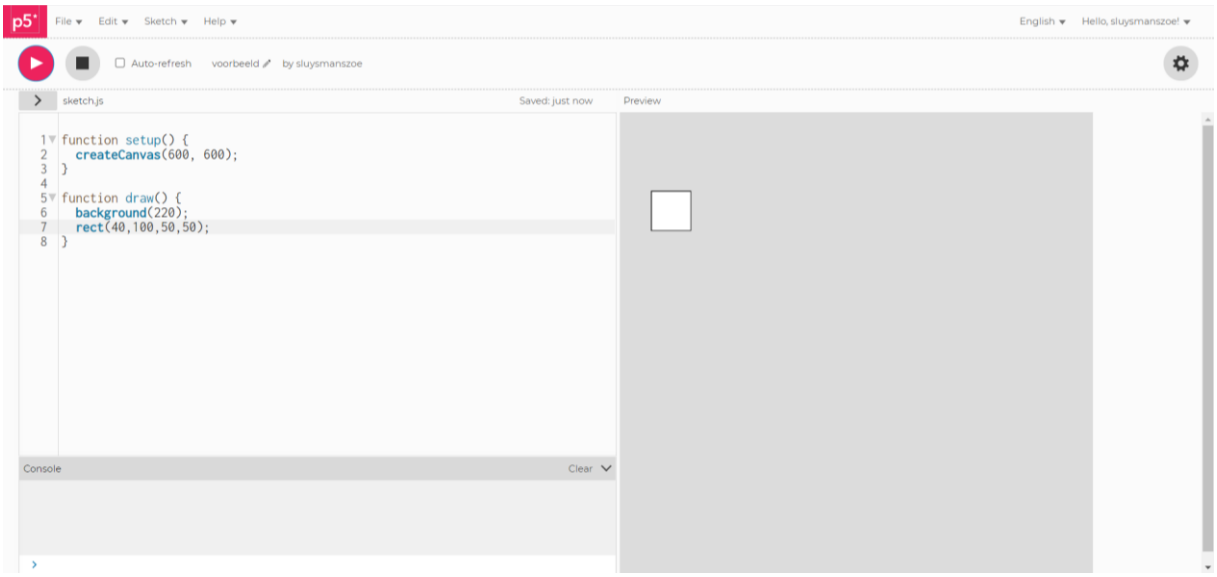
Je tekening wordt dan gedownload. Gefeliciteerd! Je kunt je tekening nu niet meer kwijtraken!

Om een nieuw bestand aan te maken, kun je het beste op ‘Back to Editor’ klikken (linksboven) en dan onder ‘file’ op ‘new’ klikken.

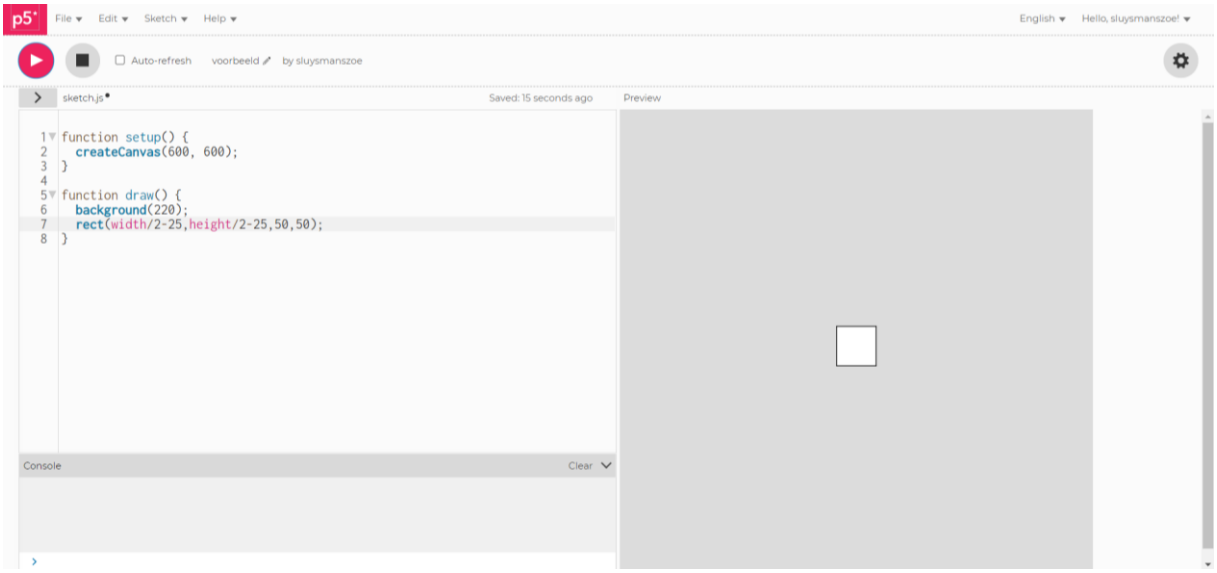


Er wordt dan een nieuw bestand geopend, zodat je een nieuwe tekening kunt maken!

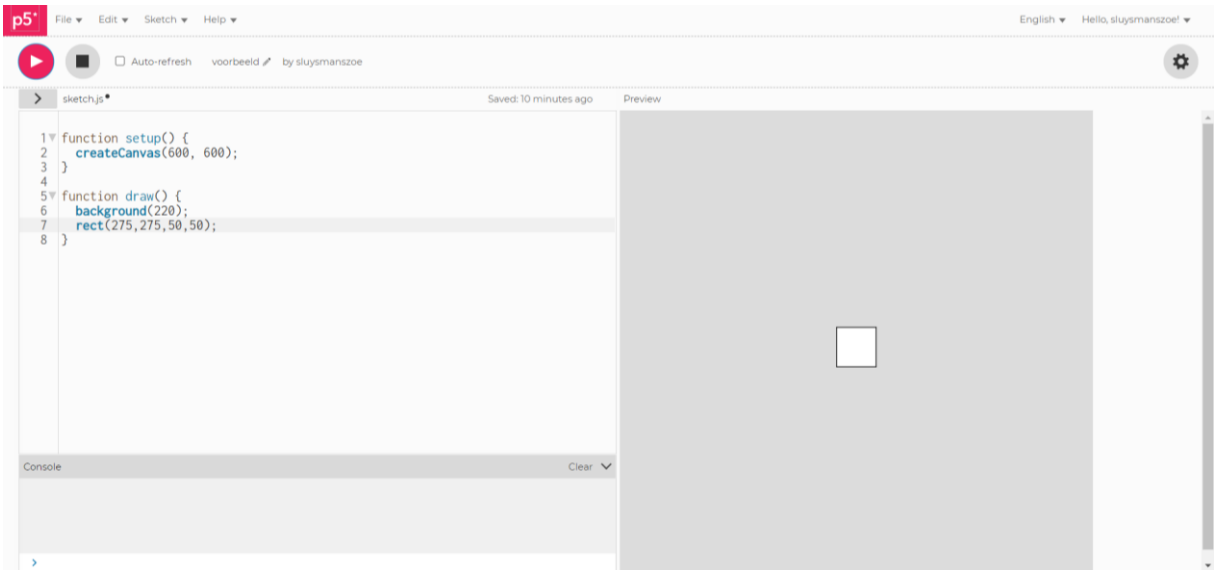
Meer vormen: voorbeelden



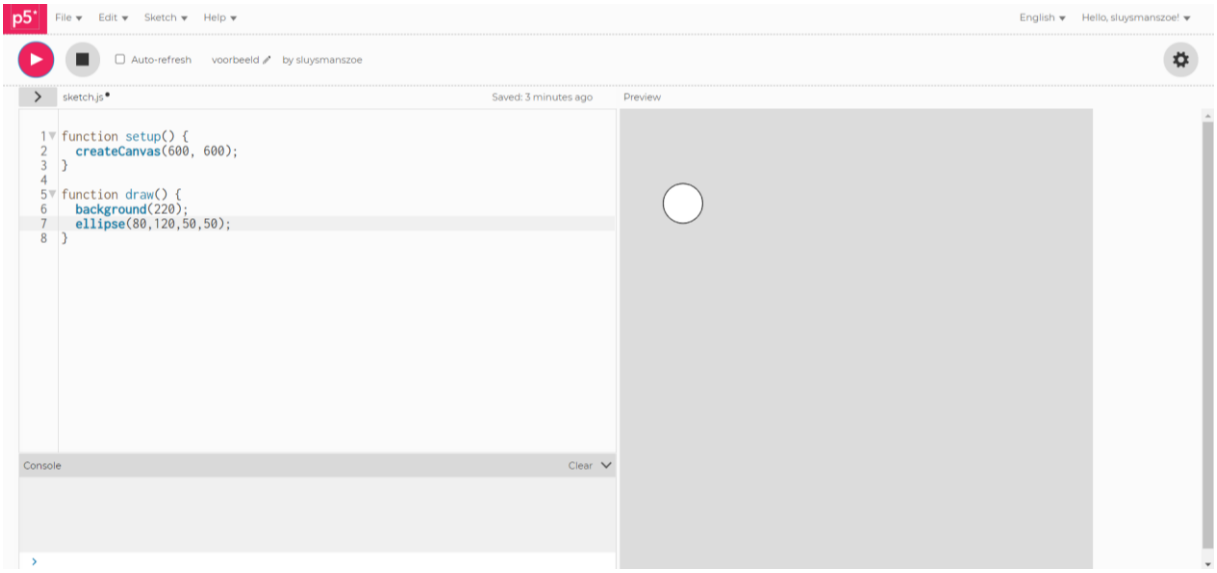
Als de breedte en hoogte van een rect gelijk zijn, heb je een vierkant.



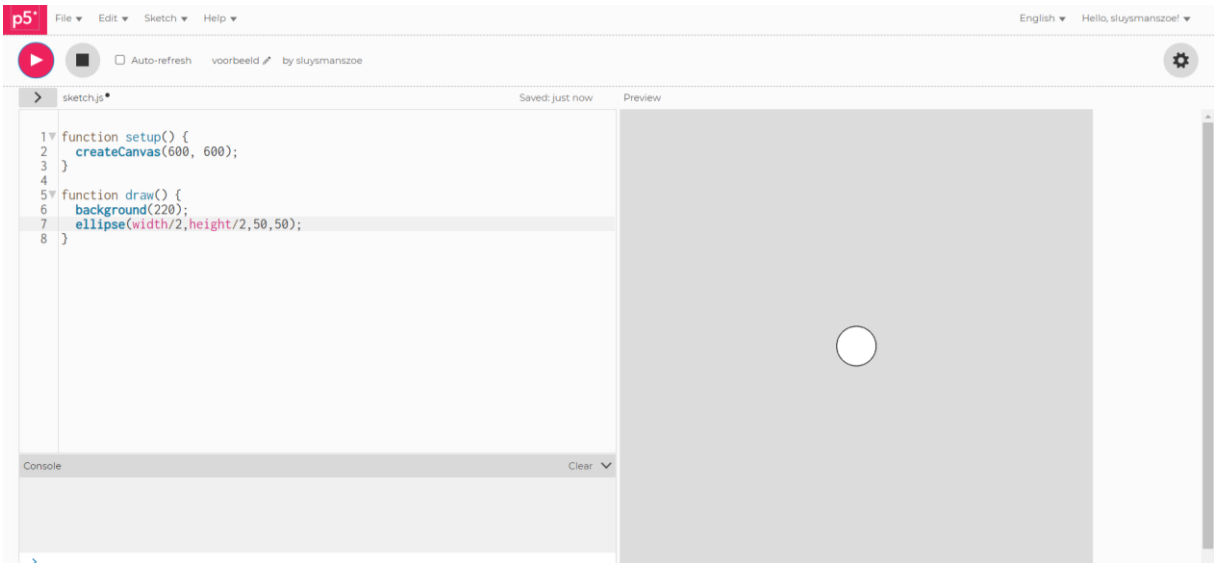
Je kunt x en y van een vorm vervangen door **width** of **height** – dit betekent de breedte en hoogte van het canvas. Een handig trucje om je rect in het midden van de pagina te krijgen, is ‘width/2’ (breedte van het canvas gedeeld door twee) of ‘height/2’, min de helft van de breedte van de rect. Je vervangt x door width en y door height.



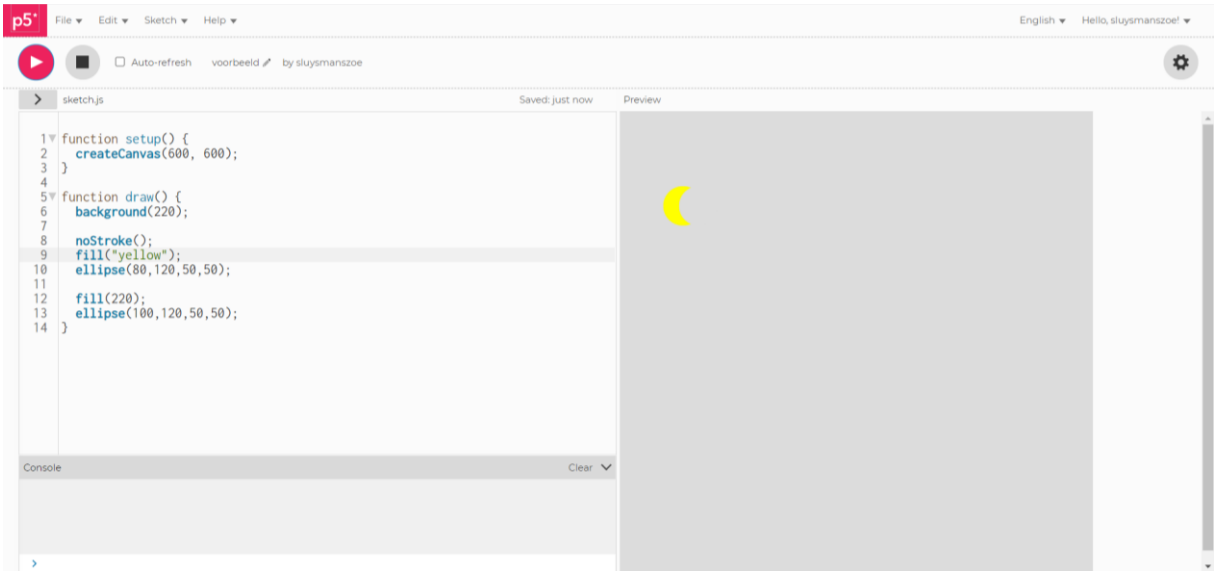
Deze code geeft hetzelfde resultaat als ‘width/2-25’ en ‘height/2-25’ (de uitkomst van die rekensommen is in dit geval 275) maar dat is meer rekenwerk. Je mag zelf kiezen wat je handiger vindt.



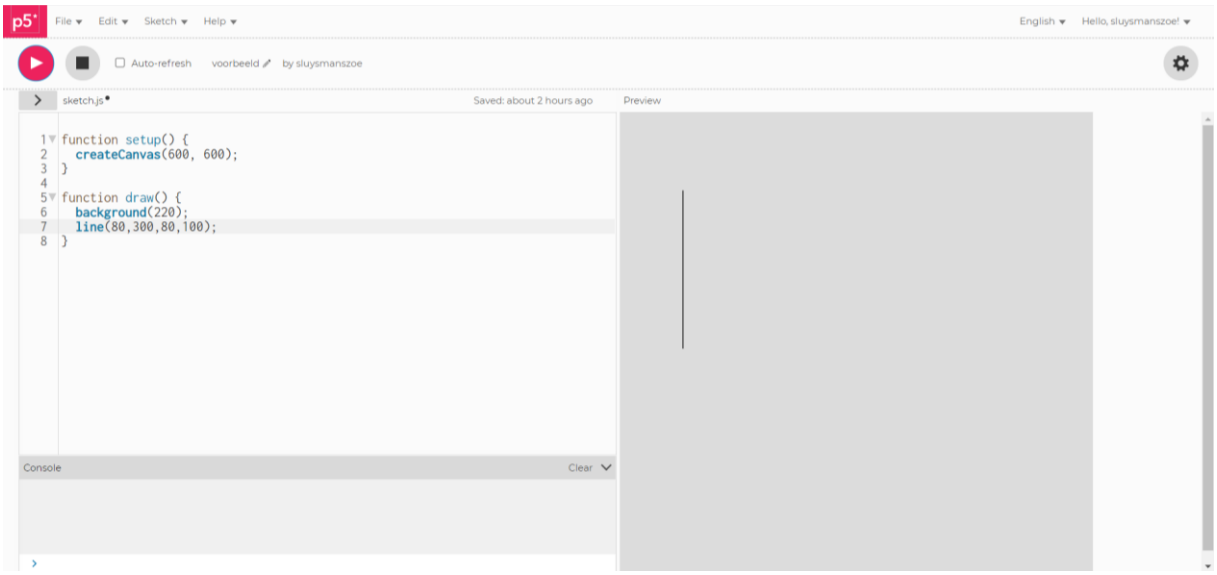
Als de x en y van je ellipse gelijk zijn, heb je een cirkel.



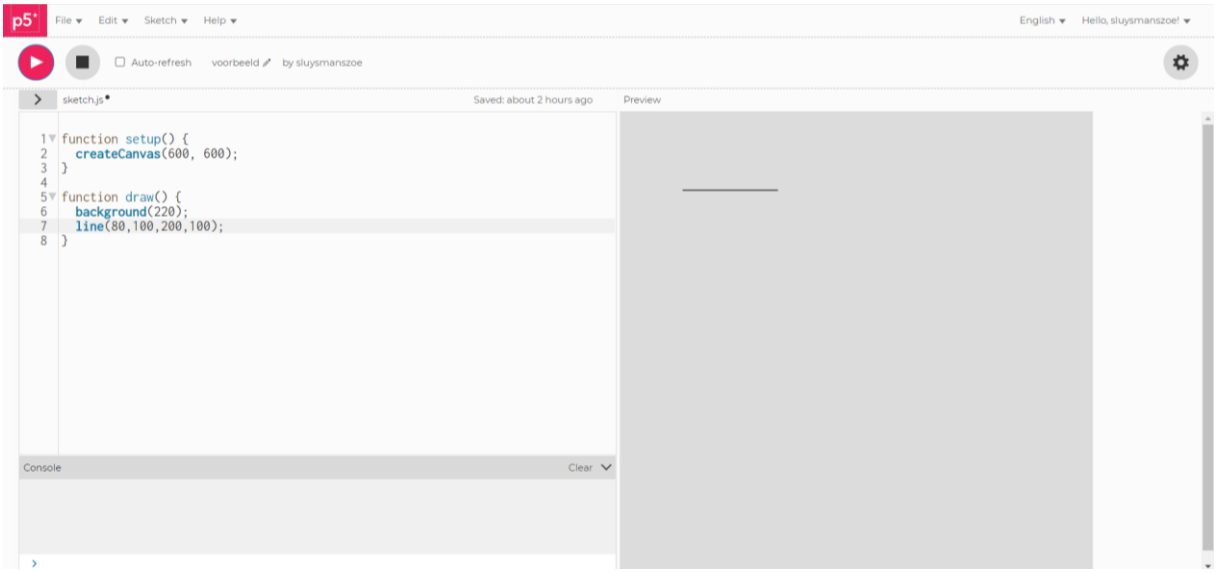
Bij een ellipse, net als bij een rect, kun je x en y vervangen door width en height. Je hoeft hierbij niet de helft van de hoogte en breedte van width en height af te trekken.



Je kunt ellipses gebruiken om ingewikkeldere vormen te maken. Hier is bijvoorbeeld een maan gemaakt door een cirkel in de kleur van de achtergrond deels over een gele cirkel heen te plaatsen.



Als x bij allebei de punten gelijk is, is je line verticaal recht.

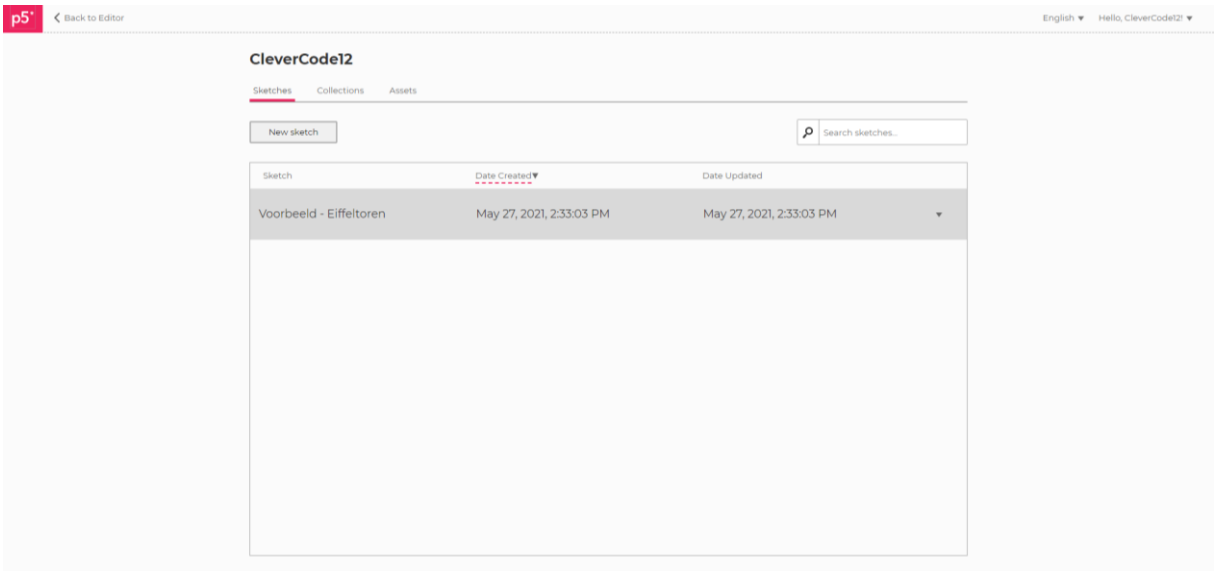


Als y bij allebei de punten hetzelfde is, is je line horizontaal recht.

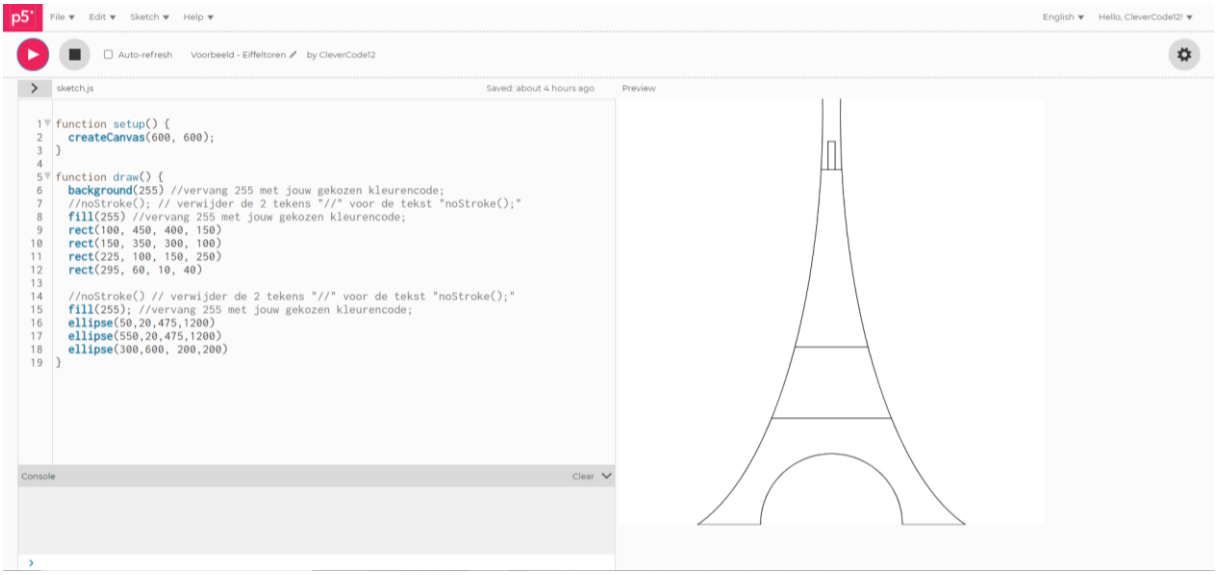
Opdrachten

En dan zijn we eindelijk aangekomen bij de opdrachten van onze workshop! We reizen naar Londen, Rio de Janeiro, New York, Hongkong en Parijs. In elke stad kun je kiezen tussen een tekening van makkelijke, gemiddelde of moeilijke moeilijkheidsgraad. Die tekening ga je namaken! (Maar als je er een creatieve draai aan wilt geven, is dat natuurlijk helemaal geen probleem.)

Als het je nu echt niet lukt om de tekening te maken, kun je in het klassenaccount een voorbeeld vinden – ‘startercode’, heet dat. In die code hebben we wat hints opgeschreven om het je wat gemakkelijker te maken. Want coderen kan soms best ingewikkeld zijn, en dat weten we zelf ook heel goed. Alle bestanden met startercode beginnen met ‘voorbeeld’, en dan wat er op de tekening te zien is. Dus ‘Voorbeeld – Dubbeldekker’, of ‘Voorbeeld – Eiffeltoren’. Door op de titel te klikken, open je het bestand.



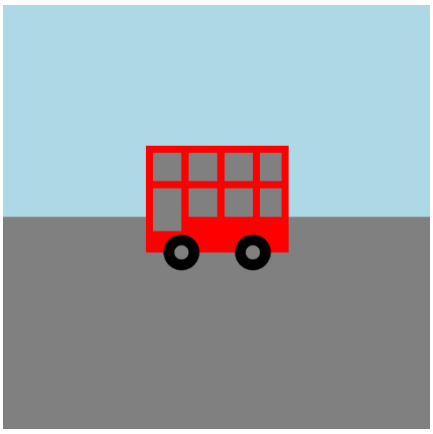
De startercode zelf ziet er dan bijvoorbeeld zo uit:



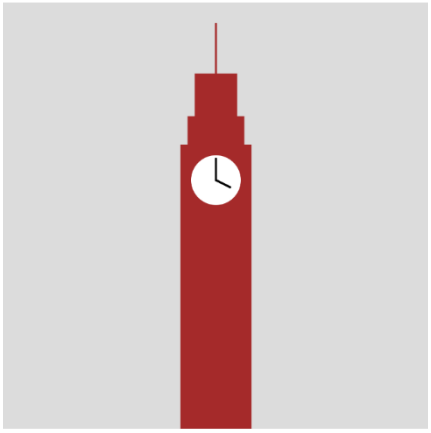
Hieronder vind je de opdrachten dus. Kies bij elke stad uit de makkelijke, gemiddelde of moeilijke optie. Probeer eerst zelf de tekening na te maken, en als het niet lukt, kun je altijd onze startercode nog gebruiken. Veel succes!

Londen

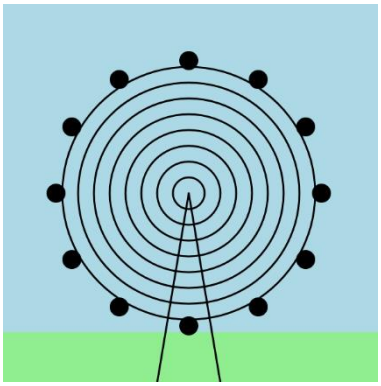
Gemakkelijk: een Londense dubbeldekker.



Gemiddeld: de Big Ben.



Moeilijk: de London Eye.

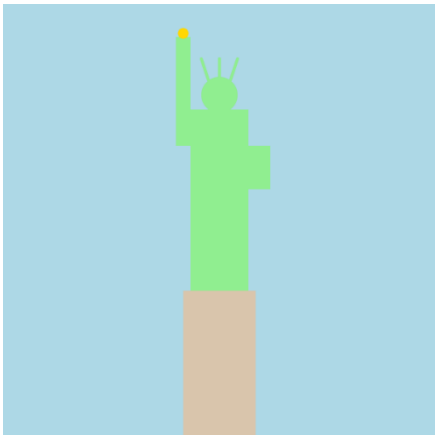


New York

Gemakkelijk: Empire State Building.

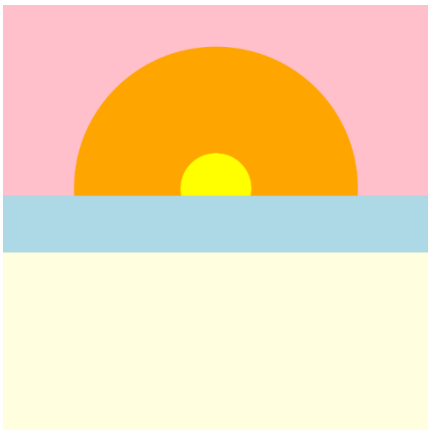
Gemiddeld: landschap met wolkenkrabbers.

Moeilijk: het Vrijheidsbeeld.

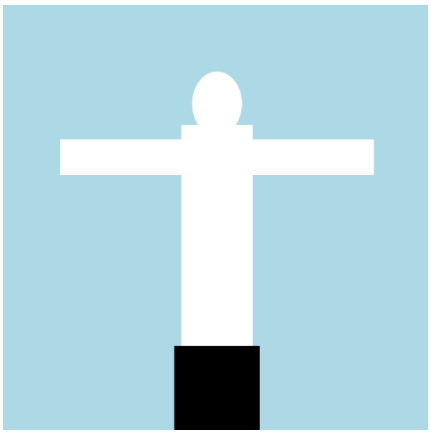


Rio de Janeiro

Gemakkelijk: strand.



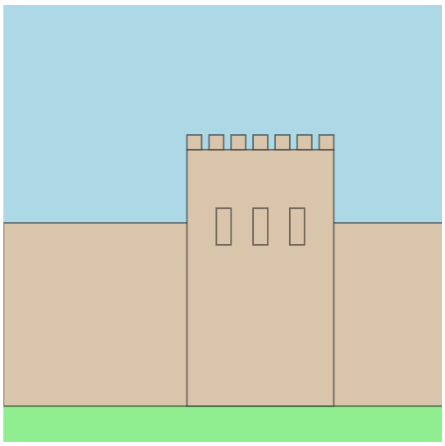
Gemiddeld: Christus de Verlosser.



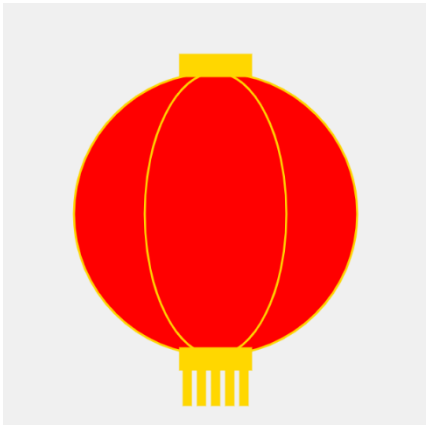
Moeilijk: Carnavalsmasker.

Hongkong

Gemakkelijk: de Chinese Muur.



Gemiddeld: een lampion.



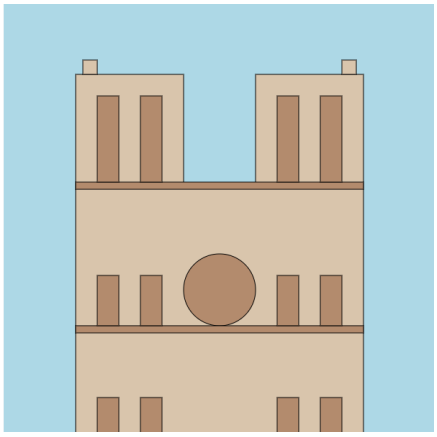
Moeilijk: een panda.

Parijs

Gemakkelijk: De Arc de Triomphe.

Gemiddeld: de Eiffeltoren.

Moeilijk: de Notre Dame.



Nawoord

Dankjewel dat je met ons op wereldreis bent gegaan! We vonden het erg leuk om met je op pad te gaan, en we hopen dat je iets nieuws hebt geleerd!

Groetjes,
Gemma, Max, Milène en Zoë

Waar vind je wat?