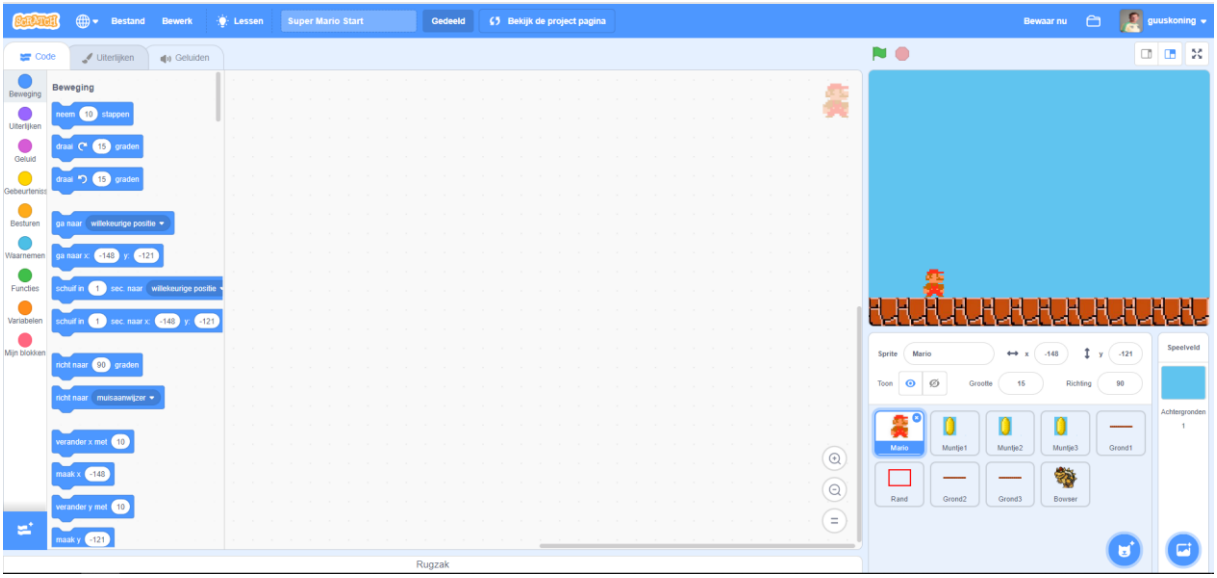
Super Mario

Alle scratchers willen graag spelletjes maken en Super Mario Bros staat dan bij velen bovenaan hun verlanglijstje. Een spelletje zoals Super Mario is best wel lastig te maken en je moet dan ook al redelijk wat ervaring hebben met Scratch.

Om deze handleiding te kunnen volgen zul je iets moeten weten over beweging, over zwaartekracht en over bewegende achtergronden.
Ik ga er dan ook vanuit dat je eerst die handleidingen doorwerkt voordat je aan deze begint.

Ga naar https://scratch.mit.edu/projects/285776253/Je ziet daar een Super Mario spel met een aantal sprites maar nog zonder de code.

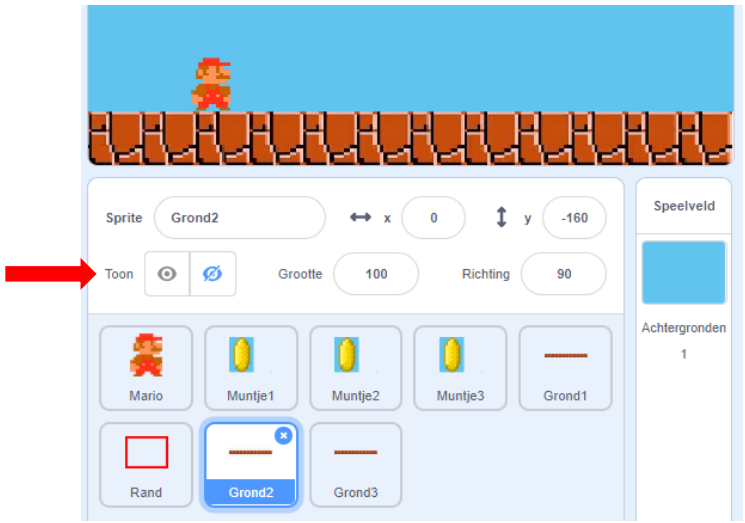
Maak een remix van dit programma zodat je al vast een beginnetje hebt voor jouw Super Mario spelletje.



Zoals je ziet gebruikt dit spelletje verschillende sprites

- Super Mario (onze held)
- Grond (drie platformen zodat Mario lekker veel kan lopen)
- Muntjes die Mario moet verzamelen
- Bowser, de vijand van Mario
- Een kader voor om het speelveld

Om het speelveld overzichtelijk te houden worden een aantal sprites niet getoond. Vergeet die niet “aan te zetten” als je ze gaat gebruiken.



Verder heeft dit spel nog een paar geluiden, het bekende Super Mario thema en een geluidje om af te spelen als Mario een muntje pakt en nog een voor als Mario een sprongetje maakt.

Laten we maar met het thema beginnen. Als het spel wordt opgestart beginnen we gelijk dat muziekje af te draaien.

Selecteer de sprite van Super Mario en voeg daar deze code aan toe.

Ik heb het volume maar even op 20% gezet anders wordt iedereen gek.

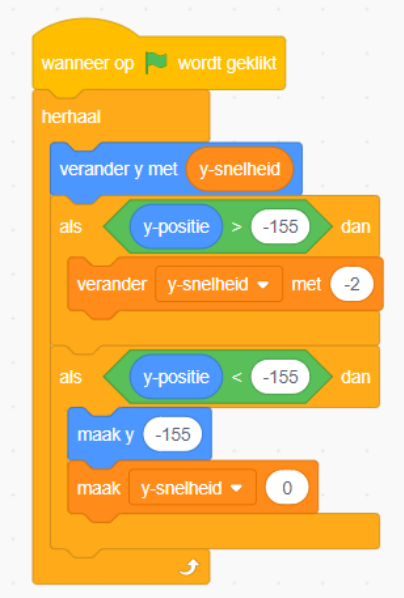
Probeer het maar uit, als het goed is hoor je nu constant dat vervelende geluidje.

Aan het beging van het spel worden meestal een aantal variabelen van hun beginwaarde voorzien en worden de sprites op hun plaats gezet.
Laten wij dat ook maar even doen.

Voeg onderstaande code toe:

We hebben een aantal variabelen gemaakt die we een beginwaarde geven en we hebben Mario op de juiste plaats gezet en naar rechts laten kijken.
De variabele *y-snelheid* werd ook in de handleiding over zwaartekracht gebruikt en geeft aan hoe snel Mario naar beneden gaat vallen. Met *verschuiving* geven we aan hoe ver de ondergrond naar links of rechts moet schuiven. Als het goed is weet je nog dat je twee sprites naast elkaar moet verschuiven om een bewegende achtergrond te krijgen.

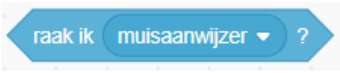
Laten we de code voor de zwaartekracht maar toevoegen. Dit heb je al een keer gedaan in de handleiding over zwaartekracht en die code kunnen we dus mooi hergebruiken.



Denk er om dat je bij de bovenste conditie een **groter-dan** teken gebruikt en bij de onderste een **kleiner-dan**.
Even kijken of het werkt. Sleep Mario een stukje omhoog en druk op de groene vlag. Als het goed is valt Mario nu naar beneden. Het lijkt goed te gaan, maar als je goed kijkt zakt Mario door de grond en stopt hij pas op de bodem van het speelveld. Eigenlijk moet Mario al stoppen zodra hij de grond raakt.

Weet jij hoe we dat zouden kunnen oplossen?

Waarschijnlijk kun je al zo goed programmeren dat je begrijpt dat je het blok

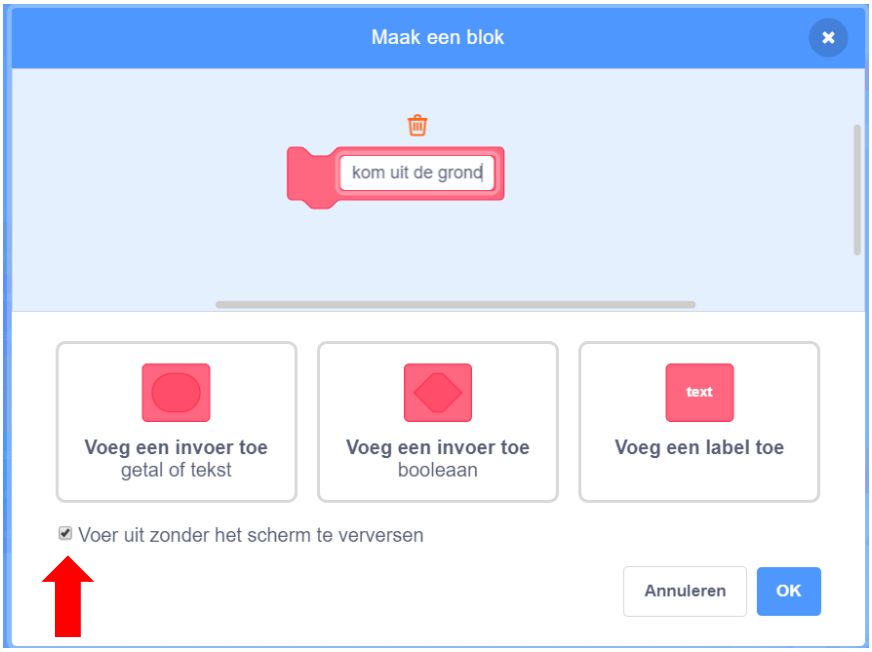


hiervoor kunt gebruiken. Je moet dan natuurlijk niet kijken of Mario de muisaanwijzer raakt maar of Mario de grond raakt.

Als je deze code uitvoert zal je zien dat Mario netjes naar beneden valt maar soms een beetje doorschiet. Hij staat dan niet boven op de grond maar staat er een beetje in. Gelukkig kunnen we dat vrij eenvoudig herstellen. Zolang hij de grond raakt moet hij een klein beetje omhoog gaan, net zolang tot hij de grond net niet meer raakt.
Verander de code tot onderstaande.

Als Mario nu naar beneden valt zie je hem een stukje in de bodem zakken en daarna weer heel langzaam omhoog komen. Het werkt wel, maar we willen het eigenlijk nog beter. Het moet eigenlijk zo snel gaan dat we het helemaal niet zien, dan is het net alsof Mario precies op de grond land.

In Scratch kun je iets heel snel uitvoeren in turbo mode en dat willen we dat laatste stukje code gaan doen. Niet alles natuurlijk wat dan valt Mario ook super snel.
Maak een nieuwe code blok met als naam “kom uit de grond” en zorg er voor dat je aangeeft dat de code uitgevoerd moet worden zonder het scherm te verversen (dit zorgt ervoor dat het supersnel kan worden uitgevoerd).



Als je dit vinkje niet zet wordt na elke minuscule verplaatsing van Mario het hele scherm opnieuw getekend wat een wat schokkerig beeld zal opleveren.

Plak de code van het onderste herhaal-blok in de code-definitie van het nieuwe blok.

En roep dit blok aan in het zwaartekracht blok.

Probeer het eens uit. Land Mario nu netjes op de grond?

De volgende stap is Mario te laten springen. Ook dat heb je al een keer geprogrammeerd en die code kunnen we dus ook weer overnemen.

Jammer genoeg werkt deze code nu niet omdat de vorige code er juist voor zorgde dat Mario net boven de grond zweeft. We zullen dus wat anders moeten verzinnen.
Nu we het toch gaan veranderen kunnen we er misschien ook een power boost voor instellen. Drukken we kort op de pijltjes toets dan gaat Mario een klein stukje omhoog, drukken we er wat langer op dan springt Mario een stuk hoger. Om dit te bereiken hebben we een nieuwe variabele nodig die bijhoudt hoe lang Mario al aan het springen is. Alleen als die variabele klein is, als Mario dus vlak bij de grond, mag Mario springen.

Maak een nieuwe variabele *inDeLucht* en verander de code van het pijltje omhoog:

Aan de code zijn nog twee blokken toegevoegd. Een om een geluidje af te spelen en de andere om Mario een iets andere vorm te geven wanneer hij springt.

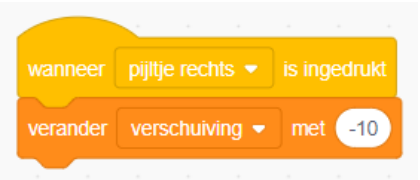
Als Mario de grond raakt moet de variabele *inDeLucht* natuurlijk op nul gezet worden. Pas de code even aan.

En als Mario gaat springen moet de variabele steeds een beetje opgehoogd worden.

Ja, het is best wel lastig om een mooi spel te maken. Als het goed is kan Mario nu lekker op en neer springen en kun je hem zelfs een power boost geven.

In de handleiding over de bewegende achtergronden heb je geleerd hoe je de wereld met twee sprites als het ware twee keer zo groot kan maken. Zo'n truc gaan we ook hier toepassen maar nu met drie sprites. De wereld wordt dan drie keer zo groot als het scherm en kan Marion lekker heen en weer rennen. Eigenlijk blijft Mario stil staan, maar omdat de grond onder hem beweegt is het net alsof Mario naar links en naar rechts kan lopen.

Selecteer *Mario* en voeg daar deze code aan toe:



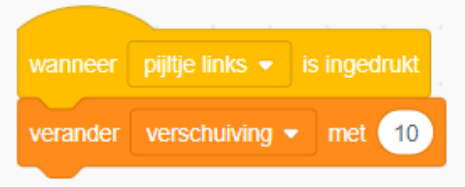
Als je deze code uitprobeert zal je zien dat Mario hier helemaal niet op reageert, hij blijft gewoon staan en beweegt helemaal niet naar rechts. Je denkt misschien dat het niet goed werkt, maar zoals eerder gezegd is dit de bedoeling. Mario moet op zijn plaats blijven maar de grond moet gaan verschuiven.
Misschien is het je al opgevallen dat we in de code de variabele *verschuiving* gebruiken. Deze variabele geeft aan over hoeveel afstand de grond verschoven moet worden.

Selecteer de sprite *grond1* en voeg daar deze code aan toe:

Aan het begin van het spel zetten we de sprite op de juiste plaats en daarna wordt de sprite steeds verschoven over een afstand *verschuiving*. Om te laten zie hoe dat werkt kun je het beste even de variabele *verschuiving* op het speelveld laten zien en dan een beetje gaan stoeien met het rechterpijltje.

Je ziet dat de waarde van *verschuiving* steeds met 10 afneemt en dat de grond onder Mario steeds 10 pixels naar links schuift. Zo lijkt het net alsof Mario naar rechts rent.

Mario kan nu wel naar rechts rennen maar nog niet naar links. Laten we die code ook maar toevoegen.



De code is natuurlijk bijna hetzelfde als voor het rechterpijltje, alleen moet de waarde van de variabele *verschuiving* nu positief in plaats van negatief zijn.

Werkt het nog?

Het begint er al aardig op te lijken. De grond verschuift nu wel maar als Mario te ver naar rechts of links gaat is er helemaal geen grond meer.
Dat kunnen we oplossen door niet alleen met de eerste grond-sprite te gaan schuiven maar met alle drie.

Selecteer *grond2* en voeg daar deze code aan toe:

De code is bijna gelijk aan die van *grond1* alleen moeten we de sprite nu verschuiven over een waard van 480 + verplaatsing. Deze sprite wordt als het ware over dezelfde afstand verplaatst maar daarnaast ook nog eens 480 pixels naar rechts zodat hij precies naast de eerste sprite komt te staan.

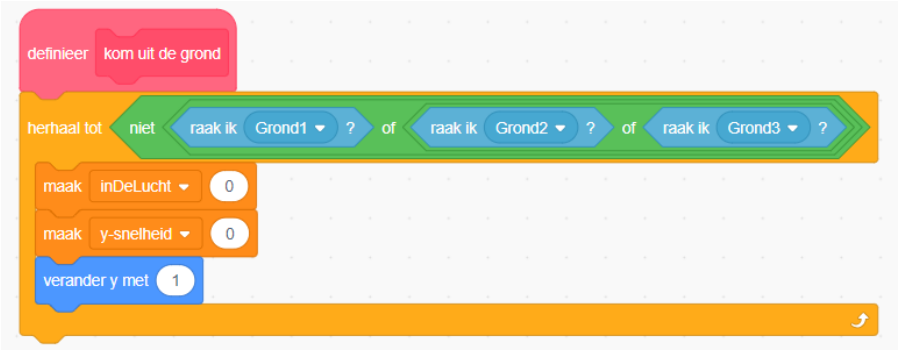
De code voor *grond3* is natuurlijk bijna hetzelfde, alleen de verplaatsing wordt nu 2*480=960 pixels.
Geef deze sprite de volgende code:

Wow, zou dat werken? Probeer het maar.

Ach, wat jammer. In het begin gaat het heel goed maar als Mario een tijdje naar rechts heeft gerend zakt hij opeens door de grond. Hoe kan dat nu, we hadden toch geprogrammeerd dat Mario niet door de grond kon zakken.
Laten we nog even naar die code kijken.

Zie je wat er mis gaat? We controleren alleen of Mario de sprite *grond1* raakt, maar als Mario een tijdje naar rechts gerend heeft komt hij bij *grond2* en daar zakt hij gelijk door heen. We moeten dus niet alleen testen of hij *grond1* raakt, maar ook de andere twee grond-sprites.

Verander de code het blok als volgt.



Zoals je ziet testen we nu op het raken van alle grond-sprites.

Als Mario springt krijgt hij een iets ander uiterlijk, iets wat we ook willen tijdens het rennen.
Laten we die code ook maar even aanpakken.

Tijd om het spel weer eens uit te proberen. Werkt alles nog?

De basis van het spel werkt nu wel. Mario kan lekker rennen en springen. Dat springen is wel leuk maar daar heeft Mario nog niet zo veel aan. Laten we op verschillende plekken wat muntjes plaatsen die Mario moet proberen te pakken. Als Mario zo'n muntje te pakken heeft moet er een geluidje afgespeeld worden en moet het muntje verdwijnen.

Selecteer *muntje1* en voeg daar deze code aan toe.

Het muntje wordt op zijn positie gezet en gaat net als de ondergrond schuiven als er op de pijltjes gedrukt wordt. Als Mario het muntje raakt moet het muntje daar mee stoppen, het geluidje afspelen en verdwijnen.

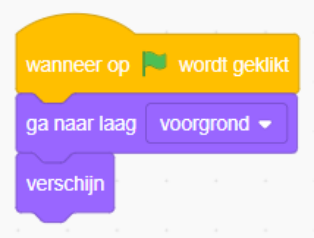
Eenzelfde soort code gebruiken we voor de andere muntjes.

Als je wilt kun je zelf nog veel meer muntjes plaatsen, je begrijpt nu wel hoe het werkt neem ik aan.

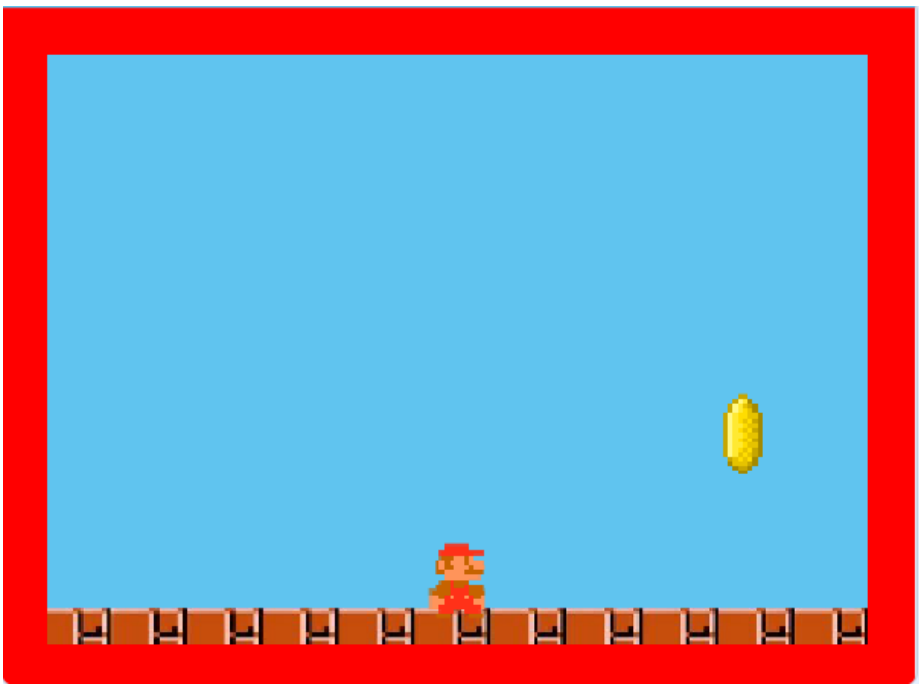
Als je het spel nu gaat spelen zie je dat het tweede en derde muntje aan de rechterkant voor een deel getoond worden terwijl ze eigenlijk buiten het beeld zouden moeten staan. Hier kunnen we weinig aan veranderen omdat de Scratch omgeving er voor zorgt dat er altijd een klein stukje van een sprite

zichtbaar blijft. Met een truc kunnen we wel doen alsnog ze er niet meer staan. Als we een rand om het speelveld plaatsen zien we er niets meer van.

Selecteer de *Rand* en voeg de volgende code toe.

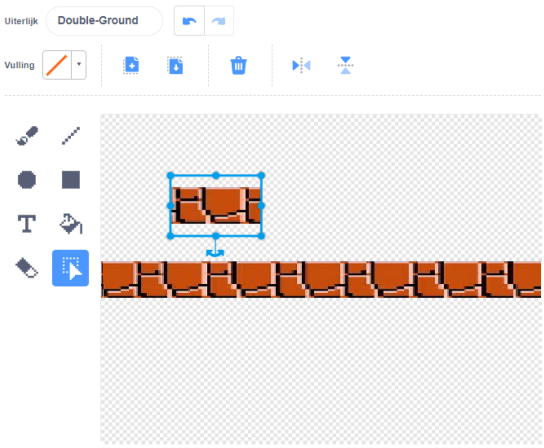


Kijk, nu zijn de sprites aan de rand tenminste niet meer zichtbaar.

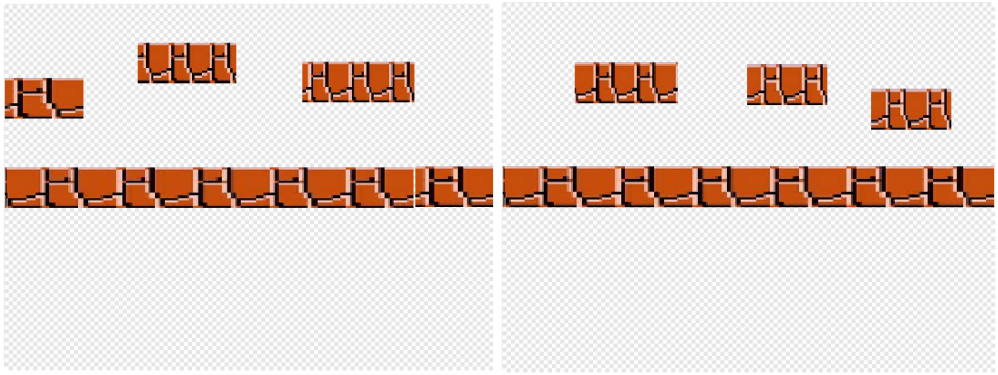


We hebben al heel veel gedaan maar we gaan nog een ding doen. In een echt Mario spel heb je altijd verschillende platformen waar Mario op kan springen. Zouden wij ook zoiets aan ons spel kunnen toevoegen?

Eigenlijk is dat helemaal niet ingewikkeld, we hoeven alleen onze grond-sprites maar een beetje aan te passen. Dit kun je het beste doen door een stukje van de grond te selecteren, deze te kopiëren en naar een andere plaats te slepen.



De eerste laat ik maar zoals het is, maar de tweede en derde heb ik een beetje aangepast.



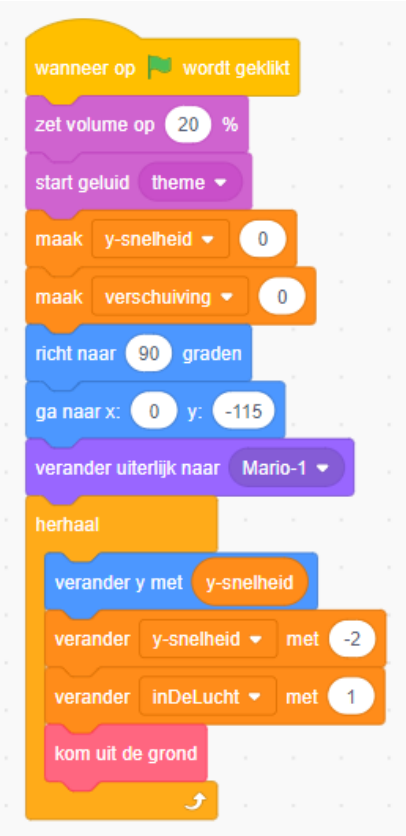
Als het goed is werkt het spel nog steeds hetzelfde alleen kun je nu ook lekker heen en weer springen.

Door het maken van de verschillende platformen kan het voorkomen dat een muntje zich nu onder of in een platform bevindt. Dit kun je natuurlijk herstellen door de beginpositie van een muntje opnieuw in te stellen. Misschien wil je er zelfs nog wel een paar bij maken, doe dat gerust.

Het spel heeft twee verschillende 

blokken en deze kunnen we beter samenvoegen. Als je meerdere van die 

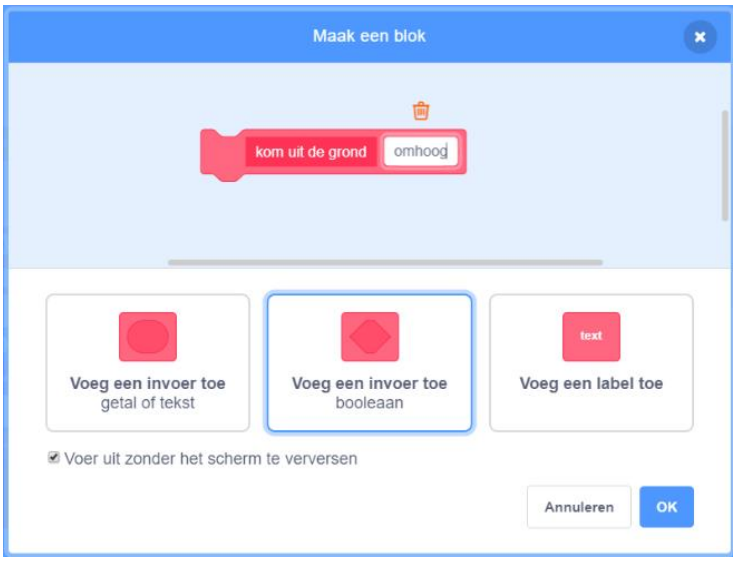
blokken hebt weet je nooit precies in welke volgorde de opdrachten worden uitgevoerd en bij één start-blok is dat wel duidelijk.



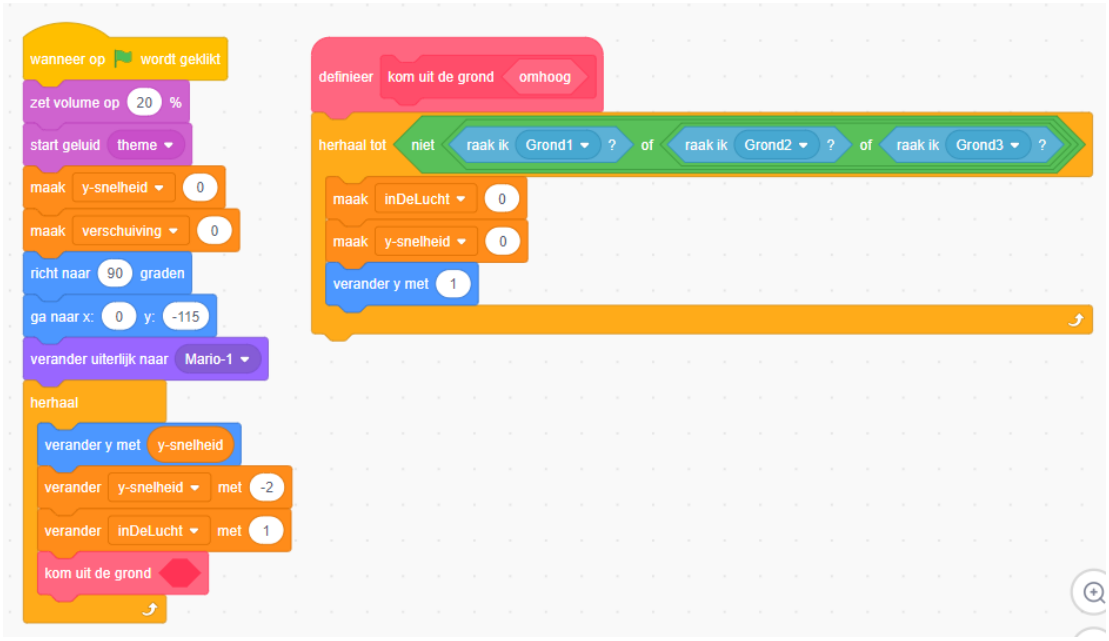
Als je het spel nu speelt zal je zien dat er nog wel wat eigenaardigheden in zitten. Raakt Mario bijvoorbeeld de onderkant van een platform dan schuift hij door het platform naar boven. We kunnen dat oplossen door te controleren in welke richting Mario beweegt als hij het platform raakt. Komt hij van onderen dat moet hij stoppen, komt hij van boven dan moet hij op de bovenkant van het platform gezet worden.

Om aan te geven of Mario van boven of van onder komt heeft het kom-uit-de grond blok een parameter nodig.
Klik met je rechtermuisknop op het kom-uit-de-grond blok en klik op bewerk.

Klik in het nieuwe scherm op de optie **Voeg een invoer toe van het type boolean** (de middelste van de drie mogelijkheden) en vul daarna achter de titel bij de parameter de waarde “omhoog” in. Druk vervolgens op **OK**.



Dit heeft als resultaat dat de code er iets anders uit ziet.



In het linkerblok zien we dat we aan de opdracht kom-uit-de-grond nog een andere blokje moeten toevoegen. Bij de definitie van het kom-uit-de-grond blok zien we de parameter “omhoog” toegevoegd.

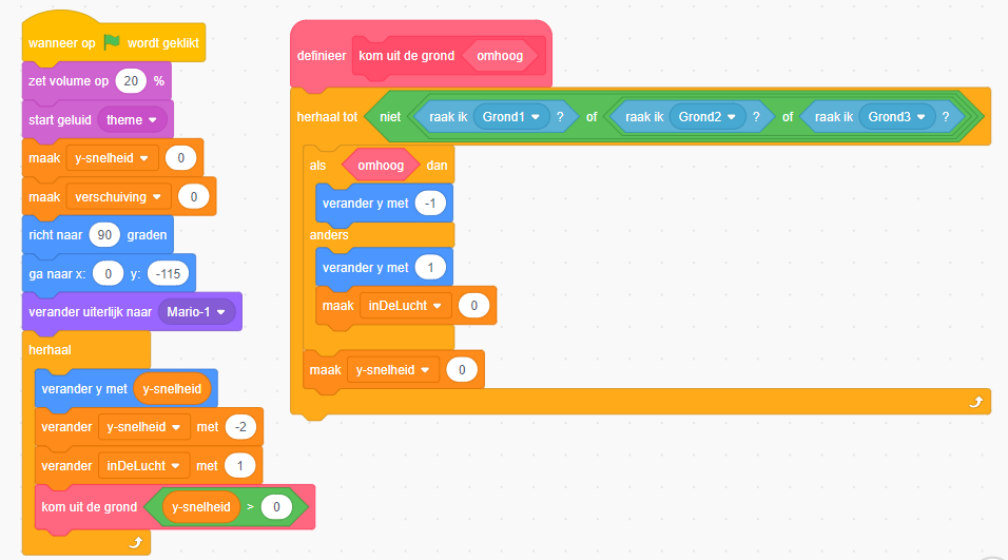
Verander de definitie van het kom-uit-de-grond blok als volgt:

We hebben een als-dan blok toegevoegd waarin we testen van welke kant Mario komt. Gaat Mario omhoog dan moet de y-waarde met 1 verminderen (het is dan net alsof Mario zijn hoofd stoot). In het andere geval komt Mario van boven en verandert er niets.

Bij de aanroep van het kom-uit-de-grond blok moeten we dan nog wel aangeven of Mario omhoog of naar beneden gaat. Als de y-snelheid > 0 is gaat Mario omhoog en dit kunnen we dus invullen in de aanroep.

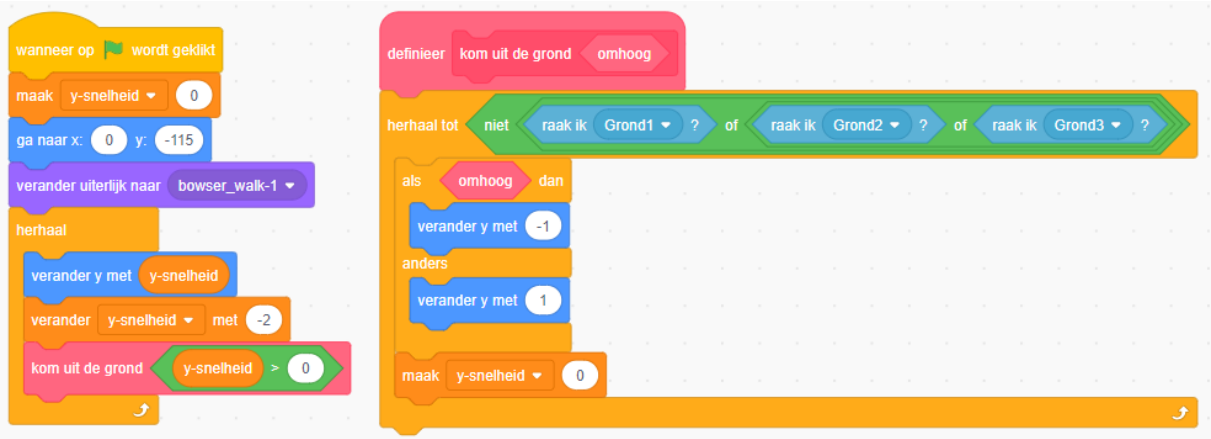
Mario kan nu lekker rennen, springen en muntjes verzamelen, maar om het spannender te maken heeft Mario een echte vijand nodig. Deze vijand springt net als Mario van platform naar platform en als hij Mario te pakken krijgt is het spel afgelopen. Mario laten we heen en weer rennen en springen met de pijltjestoetsen maar Bowser (de vijand) moet dat allemaal uit eigen beweging doen. Gelukkig kunnen we heel veel code van Mario hergebruiken voor Bowser, dat bespaart ons flink wat werk.

Kopieer het blok dat begint met **wanneer-op-groene-vlag-wordt-geklikt** en het blok dat begint met **definieer-kom-uit-de-grond** naar de sprite Bowser.



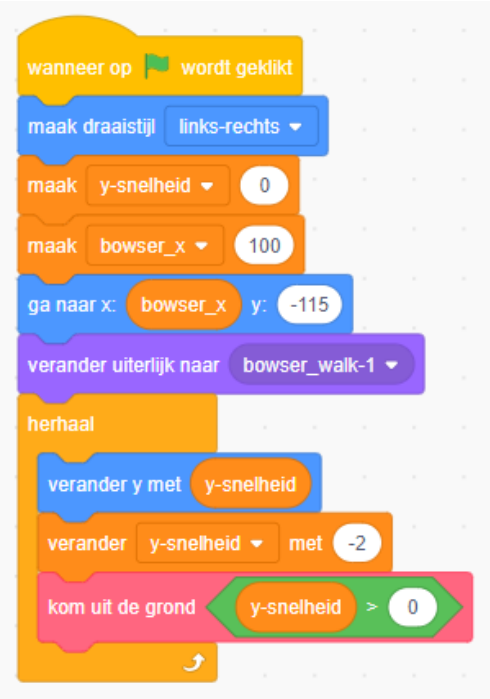
Omdat het thema van Super Mario al gestart wordt in de sprite van Mario kunnen de twee regels over de muziek uit deze code verwijderd worden. De verschuiving wordt bepaald door Mario en ook die regel kan weg.
De beginpositie van Bowser laten we nog maar even staan, dat passen we zometeen nog wel aan.
In de volgende stap wordt het uiterlijk van Mario aangepast en dat moet natuurlijk een uiterlijk van Bowser zijn. Zet het uiterlijk op **bowser_walk_1**.
Verder werd de variabele in de lucht gebruikt voor de supersprongen maar Mario en ook die opdracht hebben we hier niet nodig.

Uiteindelijk blijf deze code over:



Met deze code begint Bowser op precies dezelfde positie als Mario en dat is natuurlijk niet de bedoeling. Om de beginpositie van Bowser in te stellen gaan we gebruik maken van een variabele. Als we later Bowser willen laten bewegen kunnen we dat doen door die variabele aan te passen.

Maak dus een variabele “bowser_x” aan en geef deze aan het begin van het spel de waarde 100. Gebruik vervolgens deze variabele om Bowser op zijn plaats te zetten.



Aan het begin van het blok is ook nog de opdracht **maak draaistijl links-rechts** toegevoegd zodat Bowser netjes wordt gespiegeld als hij de andere kant op moet bewegen.

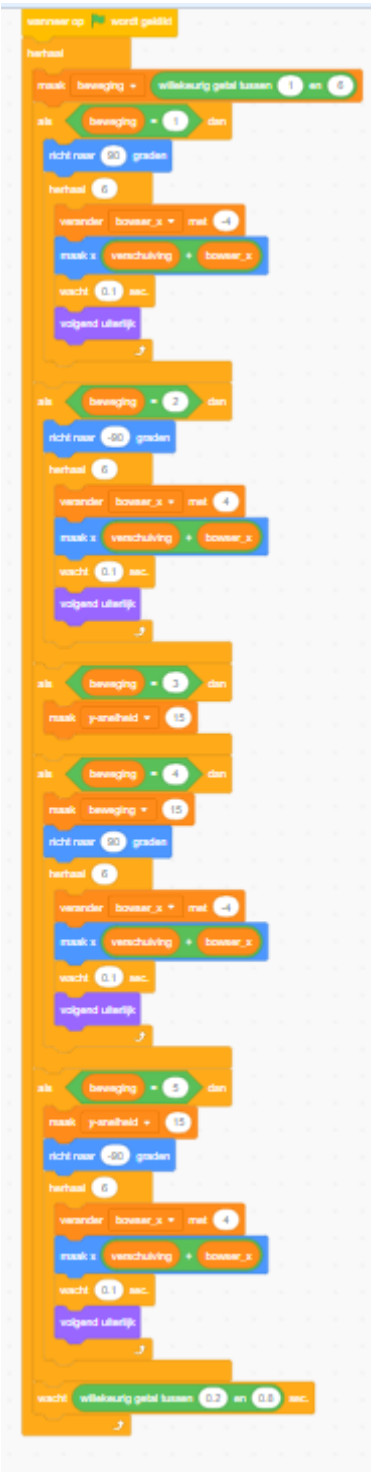
Voor het bewegen van Bowser hebben we een variabele nodig die aangeeft op welke manier Bowser beweegt. Daarvoor gebruiken we de getallen van 1 t/m 6 waarbij elk getal een bepaalde beweging aangeeft, te weten:

1. Loop naar links
2. Loop naar rechts
3. Spring omhoog
4. Spring naar links
5. Spring naar rechts
6. Doe niets

Maak een nieuwe variabele “beweging” aan voor alleen de Bowser sprite.

In een herhaling moet steeds een willekeurig getal tussen 1 en 6 gekozen worden. Afhankelijk van de getrokken waarde wordt een bepaalde beweging uitgevoerd waarna een tijdje gewacht wordt voordat het proces zich weer herhaald.

Voeg onderstaande code toe aan de sprite Bowser.



Een hele lap code en je zie vast wel dat stukken codes precies hetzelfde zijn. In zo'n geval wil je de codes die zich herhalen in een apart blok weergeven. Misschien een goede oefening om te controleren of je dat begrijpt.

Als het goed is beweegt Bowser nu ook over het scherm.

Als Bowser Mario te pakken krijgt moet het spel afgelopen zijn. Voeg daarvoor de volgende code toe aan het begin van het blok met de groene vlag.

Zolang Bowser Mario niet raakt gebeurt er niets, maar zodra dat wel het geval is stopt het spel.

Wow, het was veel werk maar wat hebben we een mooi spel gemaakt.

Misschien heb je nog wel zin om het spel te verbeteren. Je zou dan kunnen denken aan:

- Meerdere vijanden
- Muntjes die op een willekeurige plek verschijnen
- Een teller bijhouden van het aantal muntjes dat Mario gepakt heeft
- Een nog groter speelveld
- Een speciaal scherm voor als het spel afgelopen is
- Enzovoort, enzovoort