

Datapunt 3C – Visueel programmeren

Faye Canton
Studenten nummer:
2152618
Stamgroep: 1A
Jaar: 2025

Leeruitkomsten

BC 3.2.1 Je ontwikkelt een werkend technisch prototype waarmee je de basisprincipes van programmeren kunt aantonen.

BC 3.2.2 Je maakt in je uitleg navolgbaar dat je de basisprincipes van programmeren begrijpt.

Intro

In dit bestand staat het ontstaan proces van mijn interactief scratch verhaal met de bij behorende programmeer basis principes. Deze worden toegelicht met de code uit mijn scratch en daarna worden ze ook toegelicht aan de hand van 7 billion human screenshots.

Scratch link

<https://scratch.mit.edu/projects/1131824007>

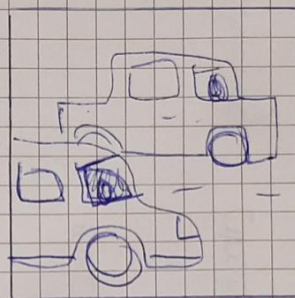
BC 3.2.1



Iemand start
in auto



Finish met
prijs geld



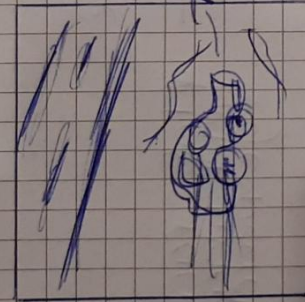
Ze race tegen
elkaar



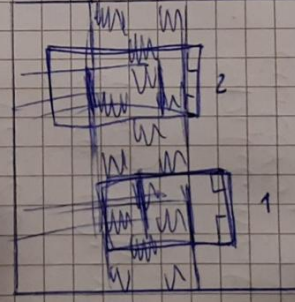
er wordt een
spijher mat op
de weg gegooid



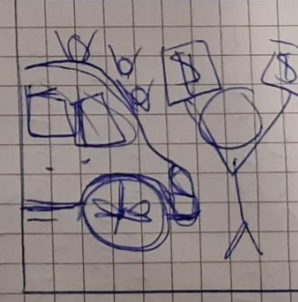
de spijher
mat word
ontwilt



de bestuurder
ziet een berg
en gebruikt
die als schans



Hoofd persoon
wint de race
op het nippertje



winnaar bij de
finish

Mijn verhaal

Mijn verhaal is dat 2 mensen in een straatrace tegen elkaar gaan race. Als je wint ontvang je een prijs van 15 Duizend euro. Omdat het om zoveel geld gaat speelt jou tegenstander (De slechterik) vals door voorwerpen op de weg te gooien. Deze voorwerpen moet je ontwijken om de wedstrijd te winnen.

3 act story verhaal

Probleemdecompositie:

Game: **Story** – Plot: race voor geld waar word vals gespeeld
Levels – Punten per ontweken obstakels, voor gevangen geld en dit moet binnen een bepaalde tijd
End goal – Race winnen en zoveel mogelijk geld krijgen

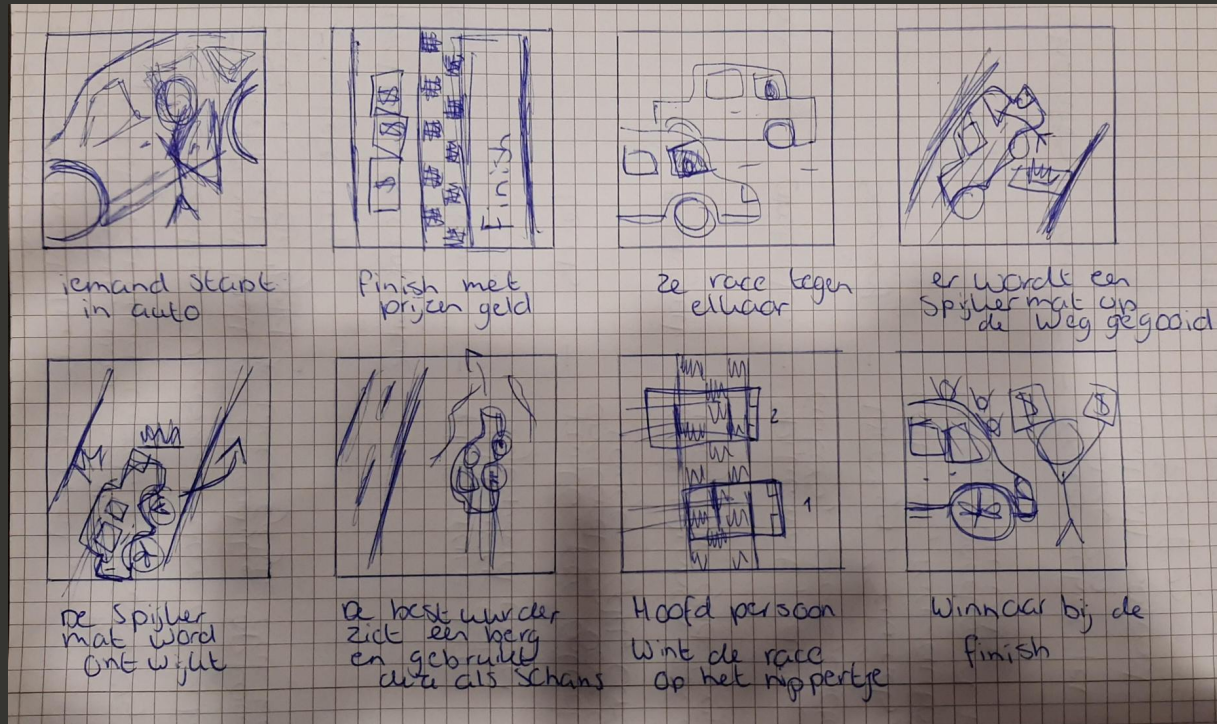
Setting: **Obstakels** – Spijkermat, andere auto's, schans en cactussen
Scenery – Woestijn, 's avonds, cactussen, aanmoedigende mensen en zandheuvels en race auto's

Character: **Music** –Auto geluiden
Costumes – Race auto's en coole outfit

moodboard



Feedback verhaal



Feedback Ferhan en Naomi:

Spannend en leuk verhaal ook de schans is een erg cool idee! Je zou wel meer obstakels en spelletjes kunnen toevoegen zodat het meer interactief word.

Is het verhaal verassend?

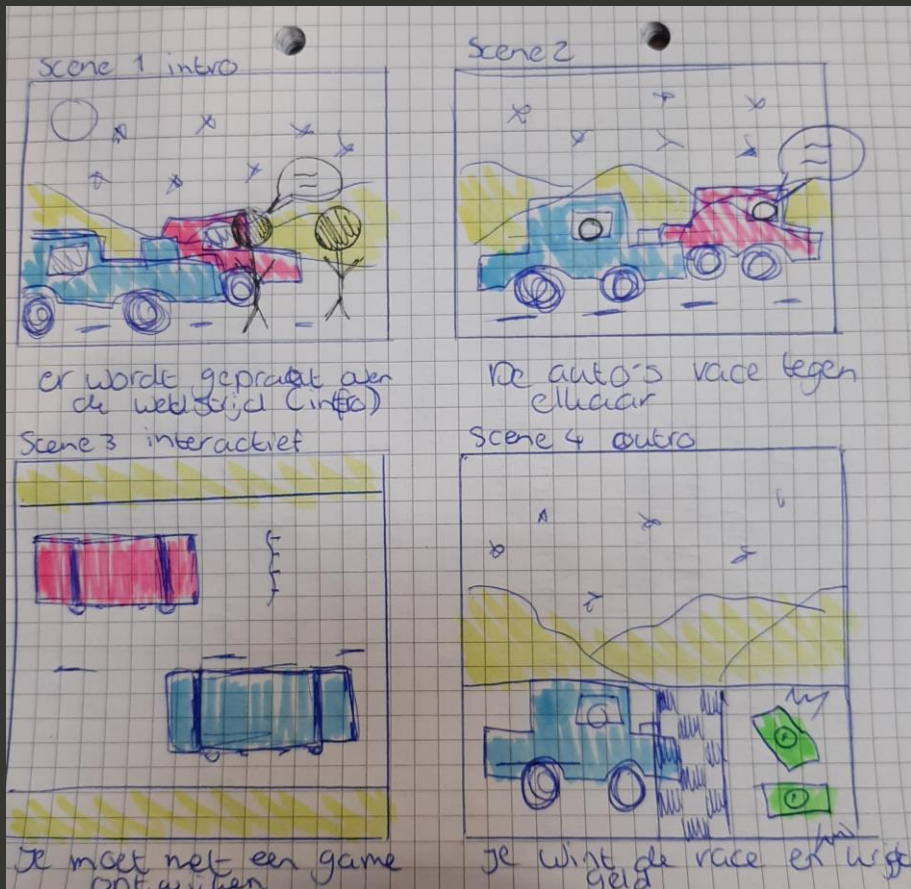
- Het verhaal is verassend door de schans

Heeft het verhaal een duidelijke opbouw?

- Het verhaal heeft een erg duidelijke opbouw omdat je gelijk van af het begin weet dat het een race is voor geld.

Feedback Ruben:

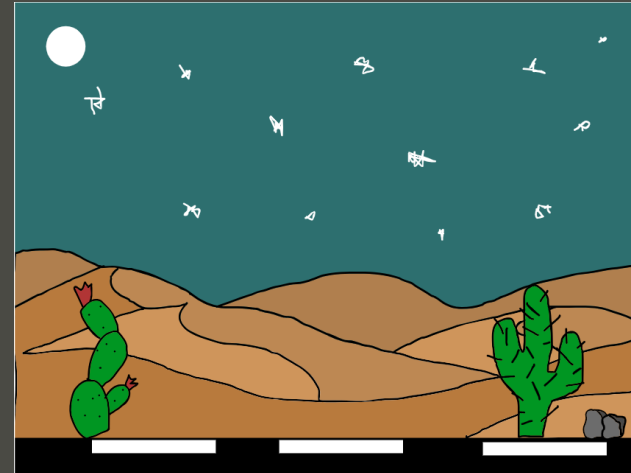
Goed en spannend verhaal maar erg complex voor een scratch beginner. Probeer een paar elementen te kiezen en maak het wat minder complex voor jezelf.



Na de feedback heb ik een nieuw storyboard gemaakt waar ik alleen de belangrijkste punten uit mijn verhaal laat zien. Nu is het verhaal minder complex en ben ik hem gaan uit werken in scratch.

Scratch

Voor mijn verhaal/game besloot ik om alles zelf te tekenen in de zelfde simpele stijl animatie stijl. Als eerst ben ik begonnen met de achtergronden



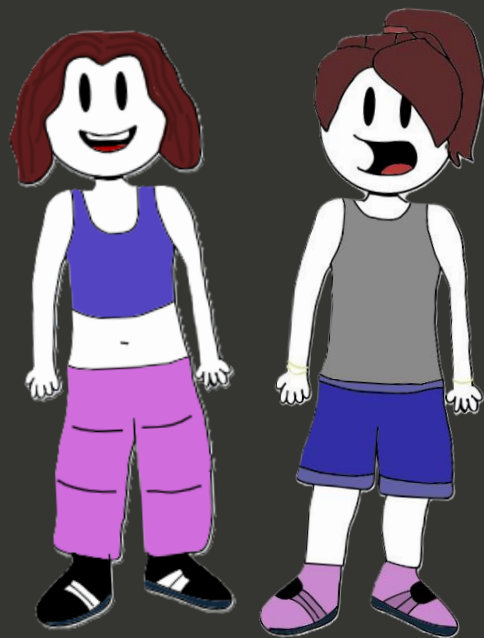
Achtergrond van de intro

Finish met prijzen geld

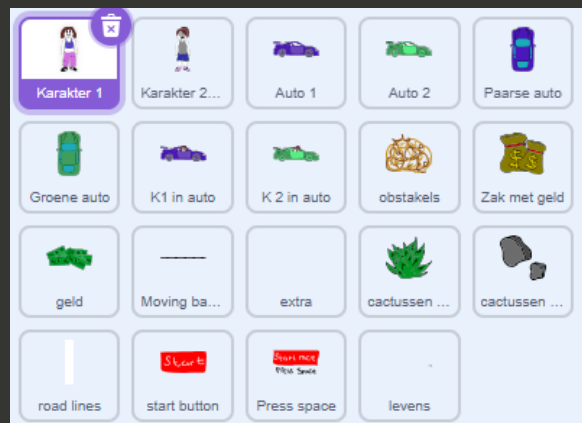


Sprites maken

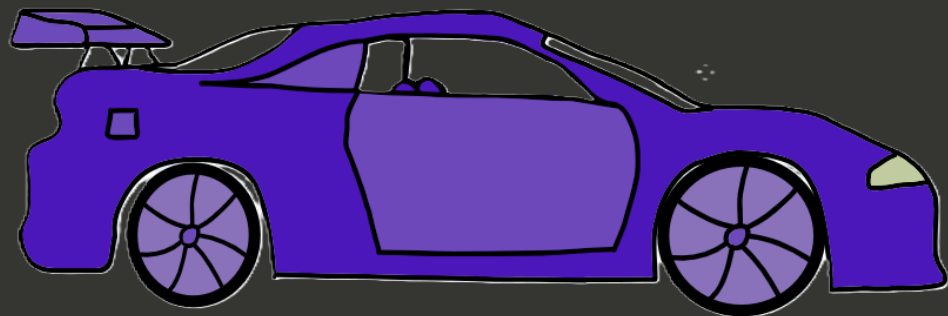
Alle sprites



Hoofd karakters

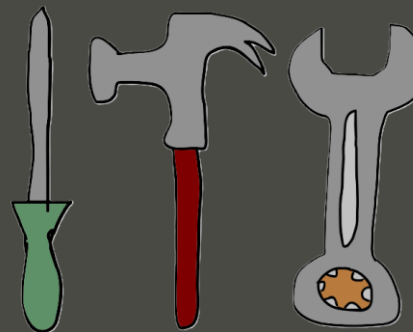


Cactus



Race auto

3 levens in het spel



Prijzen geld

Dune racer



Start

Als tweede ben ik de spites gaan maken en heb ik nog een paar achtergronden toegevoegd.

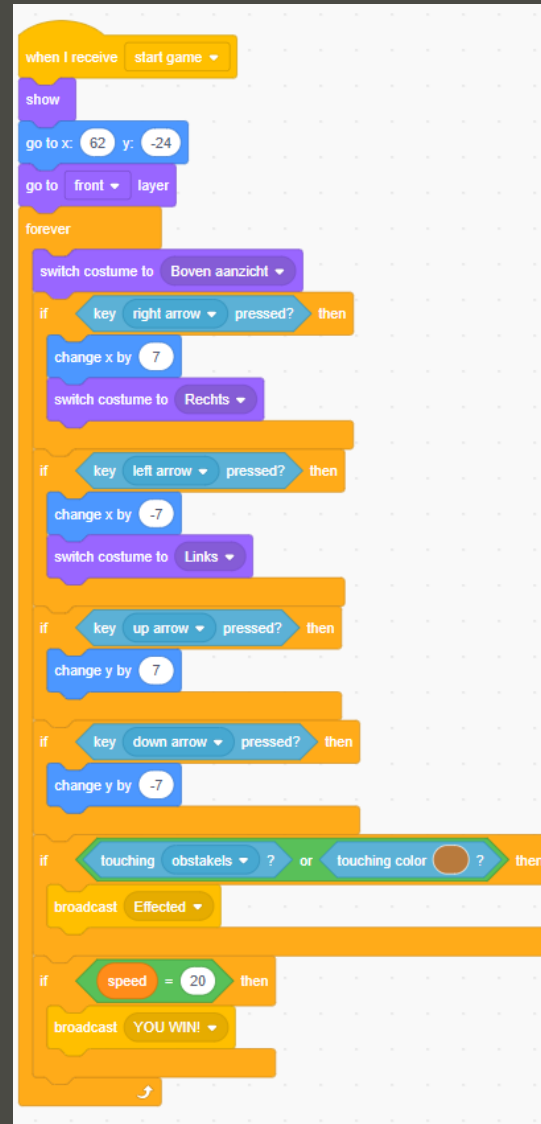


Probleem decompositie

Je deelt een groot probleem op in kleine stukjes



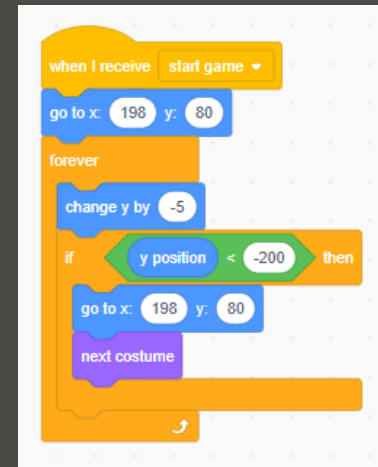
Bewijs last:



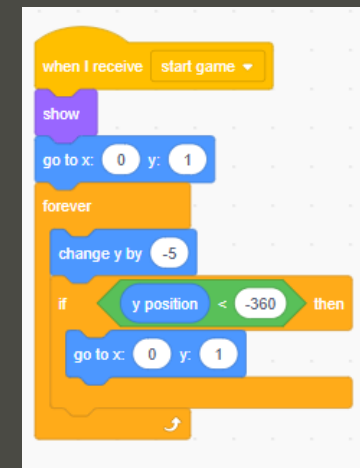
De auto bestuurbaar maken



Bewegende obstakels



Bewegende obstakels



Bewegende achtergrond

Probleem decompositie

Je deelt een groot probleem op in kleine stukjes

Bestuurbare auto

Hij moet naar voren en
achter kunnen

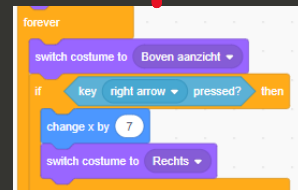
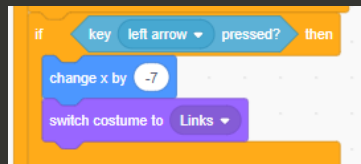
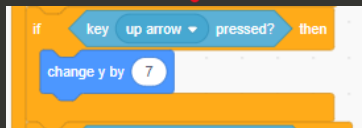
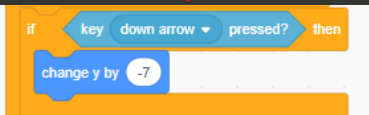
Naar achter

Naar voren

Hij moet naar links en
rechts kunnen

Naar links

Naar rechts



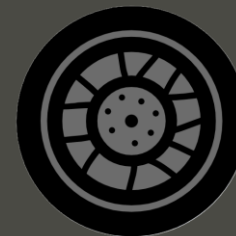
Probleem decompositie

Je deelt een groot probleem op in kleine stukjes

Bewegende obstakels

Het moeten verschillende
obstakels zijn

Auto
onderdelen
tekenen

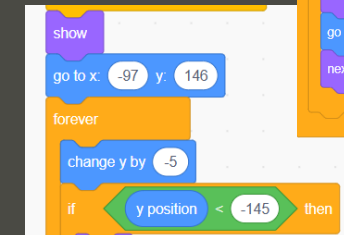


Stenen
tekenen

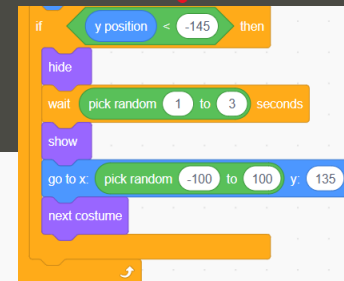


De obstakels moeten
bewegen

Ze bewegen
naar beneden

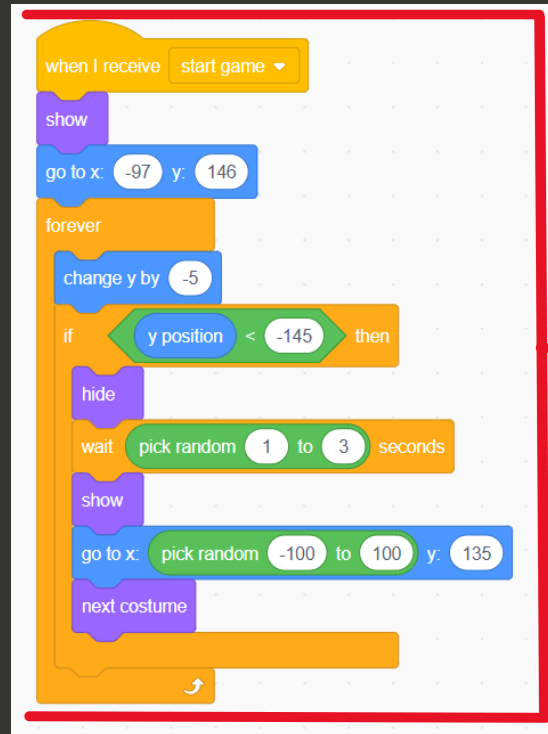


Ze bewegen
naar
verschillende
plekken



Algoritme

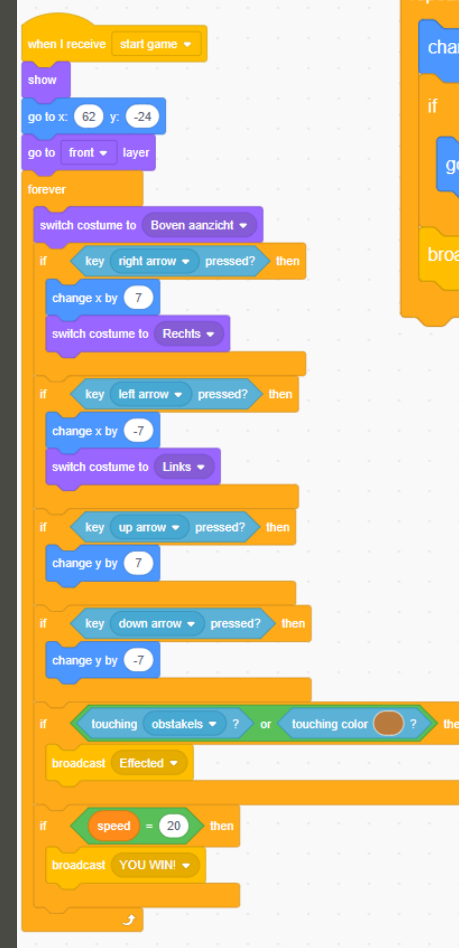
Een stappenplan dat wordt gebruikt om een specifieke taak uit te voeren of problemen op te lossen.



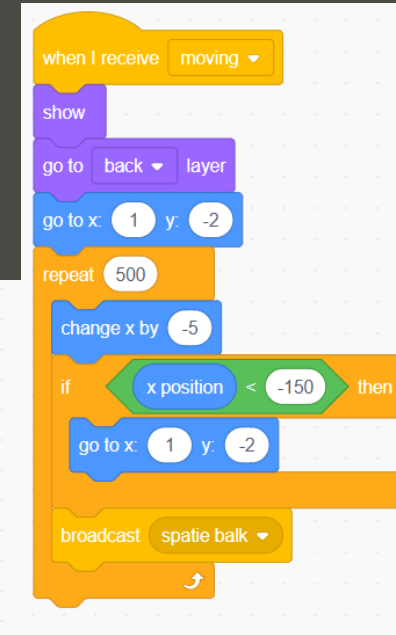
Dit is het **algoritme** voor de **bewegende obstakels** in mijn spel. Wanneer het signaal wordt ontvangen (**start game**) begint het algoritme. Het eerste obstakel komt te voor schijnt (**show**) en gaat naar een bepaalde positie (**-97x en 146y**) en gaat vervolgens naar beneden (**-5**). zodra hij onder de **-145y** positie komt verdwijnt hij voor **1 tot 3 sec** en komt dan weer tevoorschijn op een random positie tussen **-100 tot 100 op de y135**. Hij verander ook van kostuum dit herhaald zich **oneindig** door het **forever** blok.

Bewijs last:

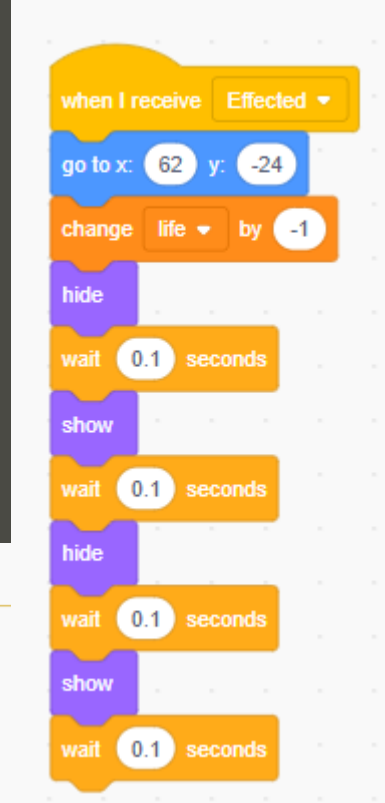
Bewegende weg



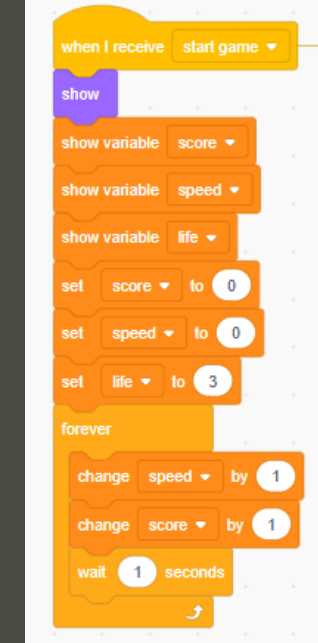
Bestuurbare auto



Knipper effect wanneer je een leven verliest



Reset van verschillende variable



Statement

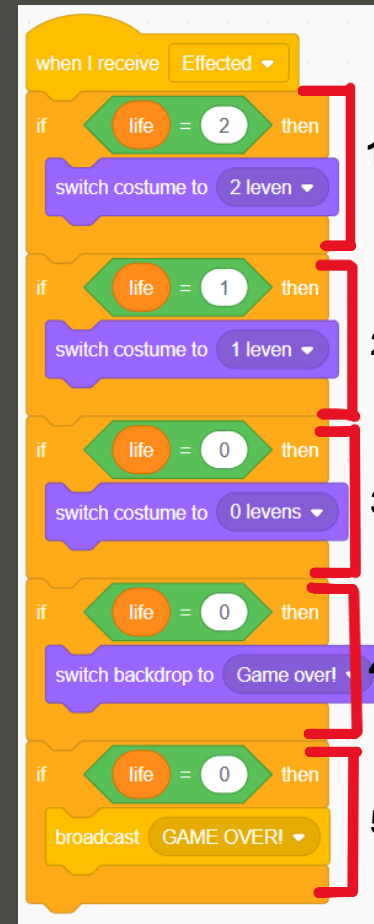
Een enkele uitvoerbare instructie



Je ziet in dit screenshot **6 verschillende statements** die allemaal een **enkele uitvoerbare instructie** zijn.

Conditioneel statement

*Een statement met een bepaalde voorwaarde.
Het stukje code controleert of de voorwaarde
waar of niet is en reageert hierop.*



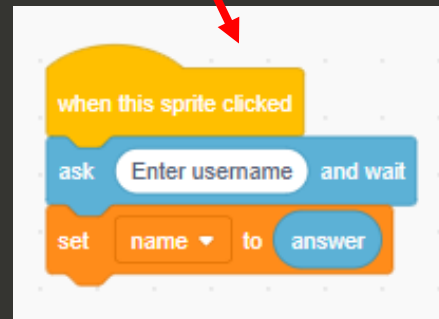
Je ziet hier **5 conditional statements**. Deze zijn voor de 3 levens die je hebt in het spel. Wanneer je een obstakel raakt of van de weg afgaat stuurt dit het **signaal effected** en gaat er een leven af. **De code controleert het aantal levens wanneer het signaal word gestuurd en reageert hier op door de kostuum te veranderen.** Elke verandering van kostuum heeft dus de **voorwaarde** van het hebben van een **bepaald aantal levens**. Als je 0 levens hebt veranderd de achtergrond en verstuurd de code en signaal.

Variabele

Een stukje code dat een waarde opslaat

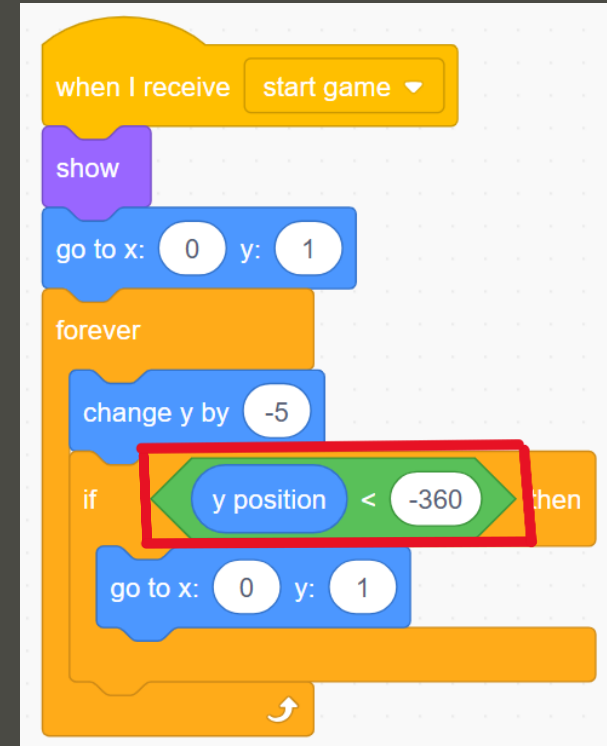


In mijn spel zitten **3 verschillende variables: Username, Speed en Life**. De speed variable **begint op 0**. De variable **Life begint op 3** omdat je 3 levens hebt. De code slaat op dat je bij **Speed** op 0 start en er **steeds 1 bijkomt**. Door een ander stuk code gaat van de 3 levens er steeds 1 af (Op de vorige dia te zien). De waarde van de username word ook door een ander stuk code opgeslagen.



Operator

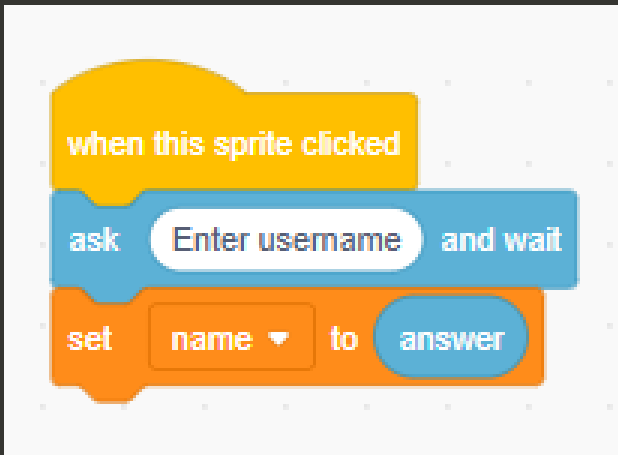
Een bewerking die uitgevoerd wordt op een variable



In dit screenshot zie je een **logische operator**. Deze operators worden gebruikt om **logische vergelijkingen uit te voeren** en geven waar of onwaar terug op basis van de logische vergelijking. De operator die je hier ziet **bekijkt of de Y positie van de Sprite Kleiner is dan (<) -360** en als dit waar is geeft hij dit door en gaat de Sprite terug naar positie 0x ; 1y.

Data types

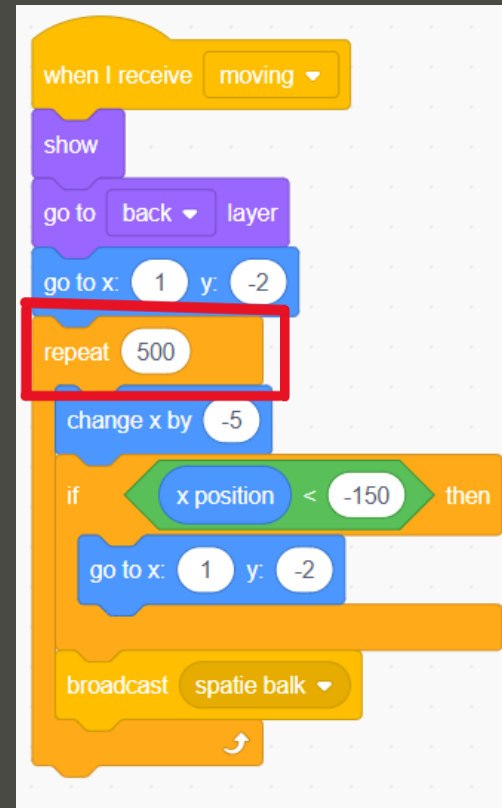
Data Types geven aan tot welk soort gegevens een waarde behoort



Hier zie je een **string** data type die de **username opslaat** die door de gebruiker word ingetypt. Vervolgens word deze laten zien boven de andere variable.

Loop

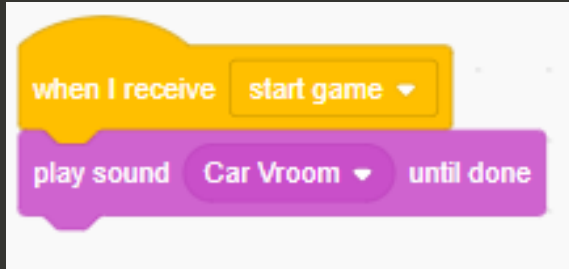
Een stuk code dat wordt herhaald totdat aan een bepaalde voorwaarde wordt voldaan



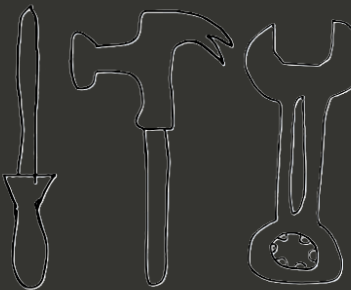
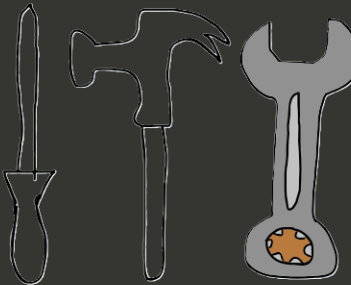
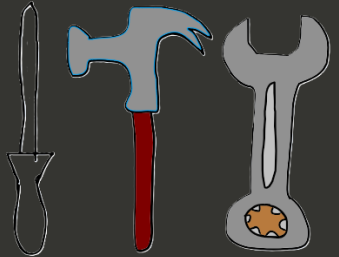
De loop die je hier ziet **heeft als voorwaarde dat deze 500x word herhaalt** en daarna stopt hij met herhalen en stuur hij de broadcast message "spatiebalk".

Extra's

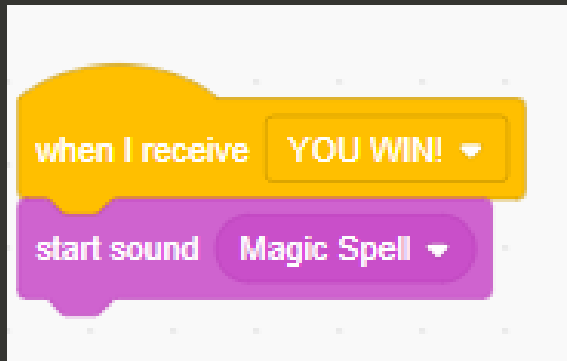
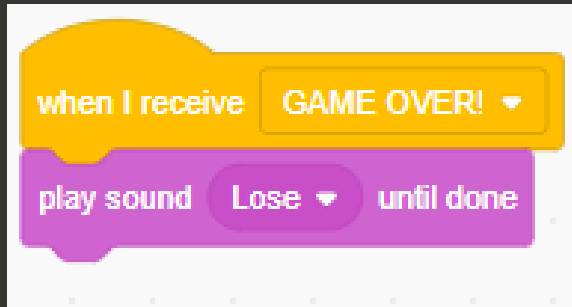
BC 3.2.2



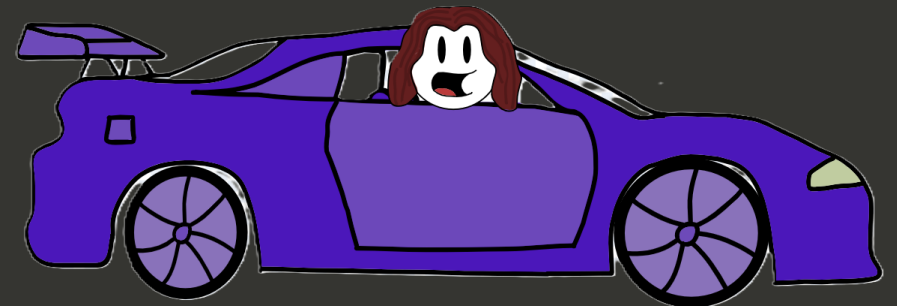
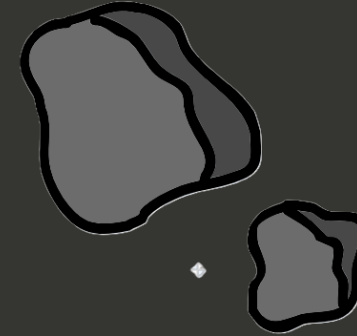
Levens die er
steeds afgaan



Sound effects



Nog meer zelf getekende Sprites



7 Billion humans

7 Billion Humans heb ik gespeeld tot level (jaar) 15

Termen begrijpen/oplossen:

- Probleem compositie
- Algoritme
- Statement
- Conditioneel statement
- Loop



Probleem decompositie

Je deelt een groot probleem op in kleine stukjes

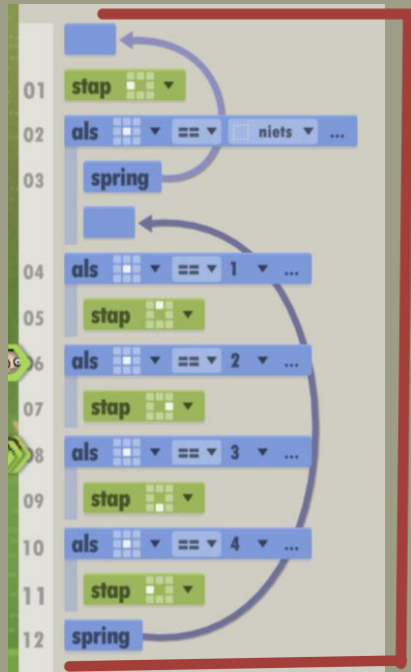
In deze code is er een **probleem compositie**. Het probleem is dat de datakubus als een soort rits (onder – boven - onder) neergezet moeten worden. Dit doe je door **verschillende stappen (kleine stukjes)** eerst pakt iedereen de datakubus op. Dan zorg je dat de eerste in de rij zijn kubus op de juiste plek zet van uit daar pas ga je ervoor zorgen dat de rest zijn datakubussen op de juiste plek zet.



Algoritme

Een eindige reeks eenduidige gedefinieerde instructies

De **hele code** bij elkaar is een algoritme het is namelijk een reeks aan specifieke instructies.



Statement

Een enkele uitvoerbare instructie

In deze code zie je **5 Statements**: Stap, pakOp en zetNeer
Ze zijn allemaal **losse uitvoerbare instructies**
01: De werknemer zet een **stap** naar beneden. **02**: De werknemer **pakt** de datakubus **op**. **03 en 04**: De werknemer zet een **stap** naar beneden. **05**: De werknemer **zet** de datakubus **neer**.



Conditioneel statement

Een statement met een bepaalde voorwaarde

In deze code zie je **2 conditionele statements**. Namelijk **als** het vakje boven de werknemer een **datakubus** heeft **pakt hij deze op** en zet hij een **stap terug**. **Als** hier onder het vakje een **versnipperaar** bevat **geeft hij hier aan** de datakubus. **Als** dit **niet is** gaat de werknemer **door (Loop)** tot dat hij **wel** een **datakubus** of **versnipperaar** vindt.

De **statements** hebben dus de een **voorwaarde alleen als** er een datakubus voor de werknemer is word deze opgepakt. **Alleen als** de werknemer een datakubus heeft gaat hij terug en geeft deze af **wanneer** er een versnipperaar is.



Loop

Een stuk code dat (oneindig) herhaalt

In deze code zorgt het statement: **Spring** voor een **loop**. De codes blijven zich steeds **herhalen** tot dat er op iedere lege plek een datakubus staat.



Programeer inspiratie

