

Activiteitenblad:

Titel: Creëer een doolhofspel met de BBC Micro:bit

Objectief:

- Begrijp het concept van perceptie in AI door de accelerometer van de Micro:bit te gebruiken om een gamekarakter in een doolhof te besturen.
- Ontwikkel codeervaardigheden om een interactief spel te maken.

Materialen:

- BBC Micro:bit met USB-kabel.
- Computer waarop de MakeCode-coderingsomgeving is geïnstalleerd.

Instructies:

Stap 1: Introductie

- Sluit de Micro:bit via USB aan op uw computer.
- Open de MakeCode-coderingsomgeving in een webbrowser.

Stap 2: Maak een nieuw project

- Start een nieuw MakeCode-project voor je doolhofspel.

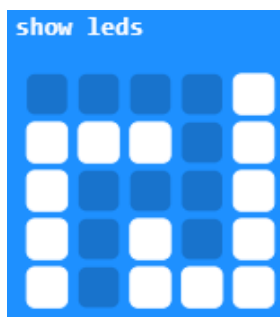
Stap 3: Begrijp perceptie

- Bij perceptie bij AI gaat het om het waarnemen en begrijpen van de omgeving. In deze activiteit gebruik je de accelerometer van de Micro:bit om kantelbewegingen te detecteren, waardoor het spelpersonage door het doolhof kan bewegen.

Stap 4: Ontwerp en pas uw doolhof aan

In deze stap ontwerpt u het doolhof met behulp van het raster in de MakeCode-coderingsomgeving. Je kunt het doolhof aanpassen door muren en open paden toe te voegen om zo een uitdagende puzzel te creëren. Het doolhof moet een duidelijk beginpunt en een bestemming hebben, het eindpunt van het spel. Houd het uitdagend: houd rekening met de moeilijkheidsgraad van je doolhof. Het pad van het begin tot het einde moet een uitdaging voor de speler zijn. De speler moet door het doolhof navigeren door de Micro:bit te kantelen en tegelijkertijd muren te vermijden om de bestemming te bereiken.





Figuur 1 LED-raster met behulp van het blok "show led" in MakeCode

Maak de doolhofindeling met behulp van het raster in MakeCode. Gebruik de blokvormen om muren, open paden, het begin ($x = 0$, $y = 0$) en het einde ($x = 1$, $y = 4$) van het doolhof weer te geven. Pas de lay-out aan zodat deze past bij het meegeleverde doolhof of maak uw eigen doolhofontwerp. Je kunt ook meer niveaus voor de speler maken.

Opmerking

De volledige set coördinaten x,y voor het raster dat micro:bit biedt, wordt weergegeven in de onderstaande tabel.

Tabel 1 X-, Y-coördinaten voor micro:bit-raster

(0,0)	(1,0)	(2,0)	(3,0)	(4,0)
(0,1)	(1,1)	(2,1)	(3,1)	(4,1)
(0,2)	(1,2)	(2,2)	(3,2)	(4,2)
(0,3)	(1,3)	(2,3)	(3,3)	(4,3)
(0,4)	(1,4)	(2,4)	(3,4)	(4,4)

De locatie van de speler op het Micro:bit-scherm wordt aangegeven door een knipperende rode LED. Effen rode LED's symboliseren muren, terwijl onverlichte LED's de doolhofpaden aangeven.

Er worden coördinaten gebruikt om de Micro:bit-LED's effectief te manipuleren! De x-coördinaten variëren van 0 aan de linkerkant tot 4 aan de rechterkant, terwijl de y-coördinaten variëren van 0 bovenaan tot 4 onderaan. Bijgevolg wordt de LED linksboven aangegeven als $x=0$, $y=0$, en dienovereenkomstig wordt de LED rechtsonder weergegeven als $x=4$, $y=4$.

Stap 5: Codeer het spel

Gebruik de volgende blokken om het gedrag van het spel te programmeren:



Figuur 2 Begin van het programma in MakeCode

Eerst moeten we een paar variabelen maken. Bedenk dat variabelen functioneren als containers waarin informatie wordt opgeslagen. In dit geval zijn twee variabelen nodig om de locatie van de speler te controleren. Eén is bedoeld om de x-positie van de speler vast te leggen, terwijl de andere is bedoeld voor het volgen van de y-positie van de speler.

Bovendien hebben we een variabele nodig om het doolhofniveau te controleren, waardoor de mogelijkheid van meerdere niveaus mogelijk is. Een andere variabele is nodig om de status van het spel bij te houden en aan te geven of het actief is of is afgelopen.

De initiële waarden zijn ingesteld om te beginnen op niveau 1 en gameOn wordt geïnitieerd als True. Dit komt omdat het de bedoeling is om bij het inschakelen van de Micro:bit onmiddellijk met het spel te beginnen. Hoewel het startpunt voor de locatie van de speler willekeurig kan worden gekozen, moet dit later worden opgeroepen bij het configureren van het doolhofniveau om ervoor te zorgen dat de speler niet binnen een muur begint. Voor dit voorbeeld wordt de speler gestart op x=0 en y=0.



Figuur 3 Eerste forever-lus

Nu de initiële variabelen aanwezig zijn, zorgen we ervoor dat onze speler wordt weergegeven op het Micro:bit-scherm!

Om een onderscheidend knippereffect voor de speler te bereiken, gebruiken we het 'plot x y'-blok afgewisseld met het 'pauze'-blok binnen een eeuwige lus. Het is de bedoeling dat de speler continu aan en uit knippert. Wanneer doolhofmuren worden geïntroduceerd, overschrijft de Micro:bit de speler elke keer dat hij de muren tekent. Door hier een pauzeblok op te nemen, zorgen we ervoor dat de speler niet onmiddellijk opnieuw wordt geplott, wat resulteert in het gewenste knippereffect.

Het gebruik van de eerder gemaakte variabelen playerX en playerY is cruciaal. Waarom? Als numerieke waarden hier rechtstreeks zouden worden ingevoerd, zou dit de flexibiliteit beperken om de speler te laten bewegen. Het gebruik van variabelen stelt ons in staat de waarden van playerX en playerY te wijzigen, waardoor de forever-lus de nieuwe locatie van de speler kan plotten.

Het is essentieel om te weten dat het pauzeblok in milliseconden werkt (bijvoorbeeld 200 ms = 0,2 seconden), en dat de knippersnelheid kan worden aangepast door de duur van de pauze aan te passen.

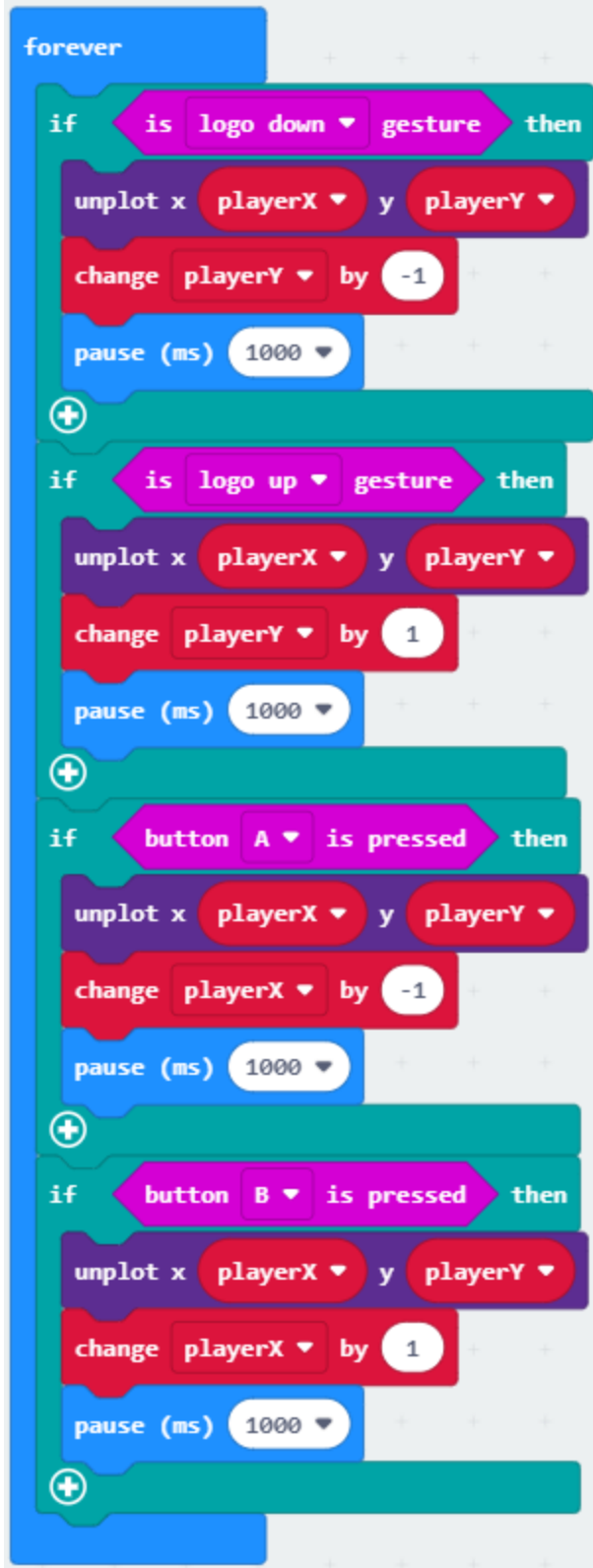


Figure 4 Second forever loop

Nu moeten we de bewegingen van de speler instellen (links, rechts, omhoog en omlaag). We zullen de twee geïntegreerde knoppen en de logo-veegfunctie gebruiken.

We stellen het gebaar 'Logo omhoog' in om omhoog te gaan, het gebaar 'Logo omlaag' om omlaag te gaan, de knop A om naar links te gaan en de knop B om naar rechts te gaan.

Om dit te bereiken gebruiken we if-statements. Deze uitspraken beoordelen of een voorwaarde waar is; Als dat zo is, worden alle blokken binnen het if-blok uitgevoerd. Wanneer we een if-instructie insluiten in een forever-lus, controleren we voortdurend of de voorwaarde waar is.

Voor spelersbewegingen passen we de variabelen playerX of playerY aan. Het is van cruciaal belang om te onthouden dat het verkleinen of vergroten van spelerX respectievelijk beweging naar links of rechts veroorzaakt, terwijl het verkleinen of vergroten van spelerY respectievelijk resulteert in een opwaartse of neerwaartse beweging. Aangezien we de locatie van de speler consequent in kaart brengen met behulp van deze variabelen, weerspiegelen alle wijzigingen automatisch de nieuwe positie van de speler.

Het is vermeldenswaard dat er na elke druk op de knop een korte pauze van 300 ms wordt toegevoegd. Dit voorkomt dat de Micro:bit de speler snel door meerdere ruimtes beweegt bij elke druk op de knop, omdat de code snel wordt uitgevoerd zonder pauze.



taken vereisen aandacht: ten

Figure 5 Third forever loop

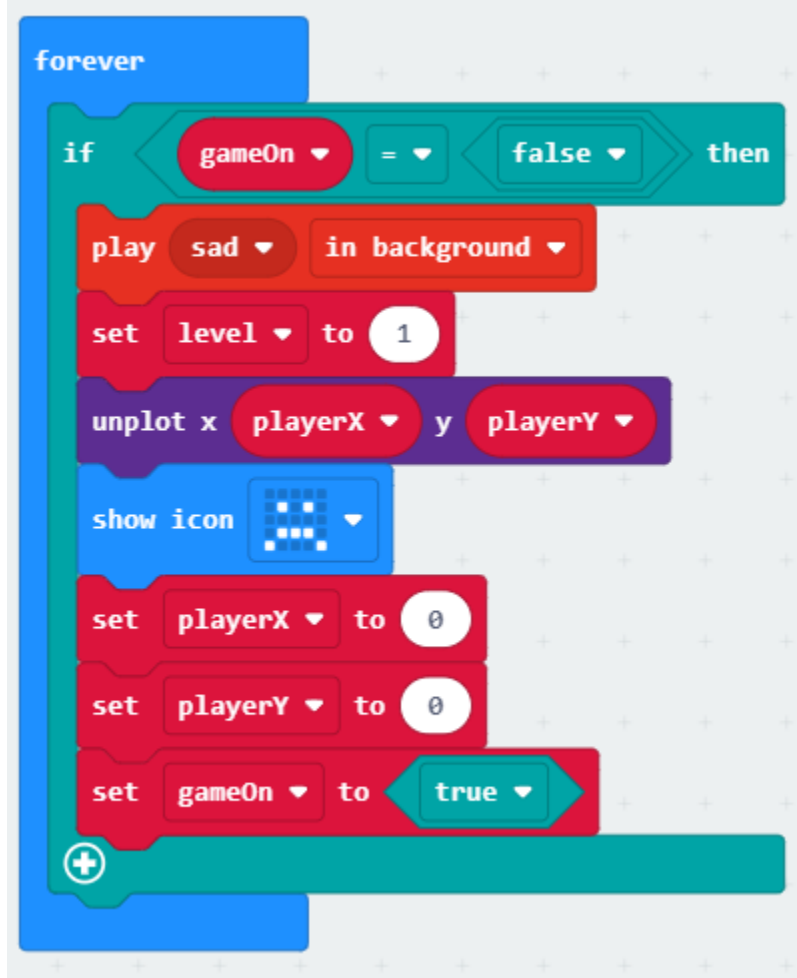
eerste het weergeven van de doolhofmuren op het LED-scherm; ten tweede, voortdurend controleren of de speler tegen een muur botst (wat aangeeft dat het spel voorbij is); en ten derde, voortdurend beoordelen of de speler het doolhofniveau met succes voltooit.

Er wordt gebruik gemaakt van een eeuwige lus. Binnen deze lus wordt een 'if'-instructie gebruikt om te verifiëren of de niveauvariabele gelijk is aan 1. Bijgevolg wordt dit codesegment alleen uitgevoerd als de niveauvariabele gelijk is aan 1. Als we meer niveaus willen toevoegen, moeten we ervoor zorgen dat dit variabele verandert dienovereenkomstig.

Binnen de 'if'-instructie worden de doolhofmuren weergegeven met behulp van het blok 'show leds' . LED's worden verlicht om muren weer te geven, terwijl onverlichte LED's de doolhofpaden aangeven. Er moet voorzichtigheid worden betracht om ervoor te zorgen dat de startpositie van de speler, eerder ingesteld op $x=0$, $y=0$, niet samenvalt met een doolhofmuur.

De volgende taak bestaat uit het controleren of de speler tegen een muur botst. Dit wordt bereikt door aanvullende 'if'-instructies, die verifiëren of de variabelen `playerX` en `playerY` uitgelijnd zijn met de coördinaten van een muur in het 5x5 LED-raster.

Ten slotte controleert de code of de speler met succes door het doolhof navigeert. In dit voorbeeld bevindt het einde van het doolhof zich op $x=1$, $y=4$. Als aan deze voorwaarden is voldaan, wordt er een succesvolle melodie afgespeeld, wordt de positie van de speler teruggezet naar het begin van het doolhof en verschijnt er een smiley op de Micro:bit. Als we extra niveaus hebben toegevoegd, moeten we ook het variabele niveau met 1 wijzigen.



Figuur 6 Vierde forever-lus

In het geval dat het spel voorbij is, moeten we een actie implementeren die wordt geactiveerd door de variabele ' gameOn ' die een botsing met een muur aangeeft.

Binnen een forever-lus wordt een 'if'-instructie gebruikt om de waarde van de variabele ' gameOn ' te bepalen. Als het gelijk is aan 'false', wordt de game-over-code uitgevoerd.

In dit geval speelt een droevige melodie op de achtergrond, wordt het 'niveau' gereset, gaat de LED van de speler uit, wordt een droevig gezicht weergegeven en begint het spel vanaf het begin.

Stap 6: Test je spel

- Test je spel door het personage door het doolhof te leiden. Werkt alles correct?

Stap 7: Speel en deel

- Deel je doolhofspel met anderen. Laad het op je Micro:bit en daag je vrienden uit om het doolhof te voltooien.



Met dit project kunnen leerlingen het concept van perceptie in AI ervaren door de accelerometer van de Micro:bit te gebruiken om een personage in een doolhof te besturen. Studenten kunnen hun eigen doolhoven ontwerpen, verschillende moeilijkheidsgraden creëren en hun games delen met leeftijdsgenoten voor extra plezier en leren.

