



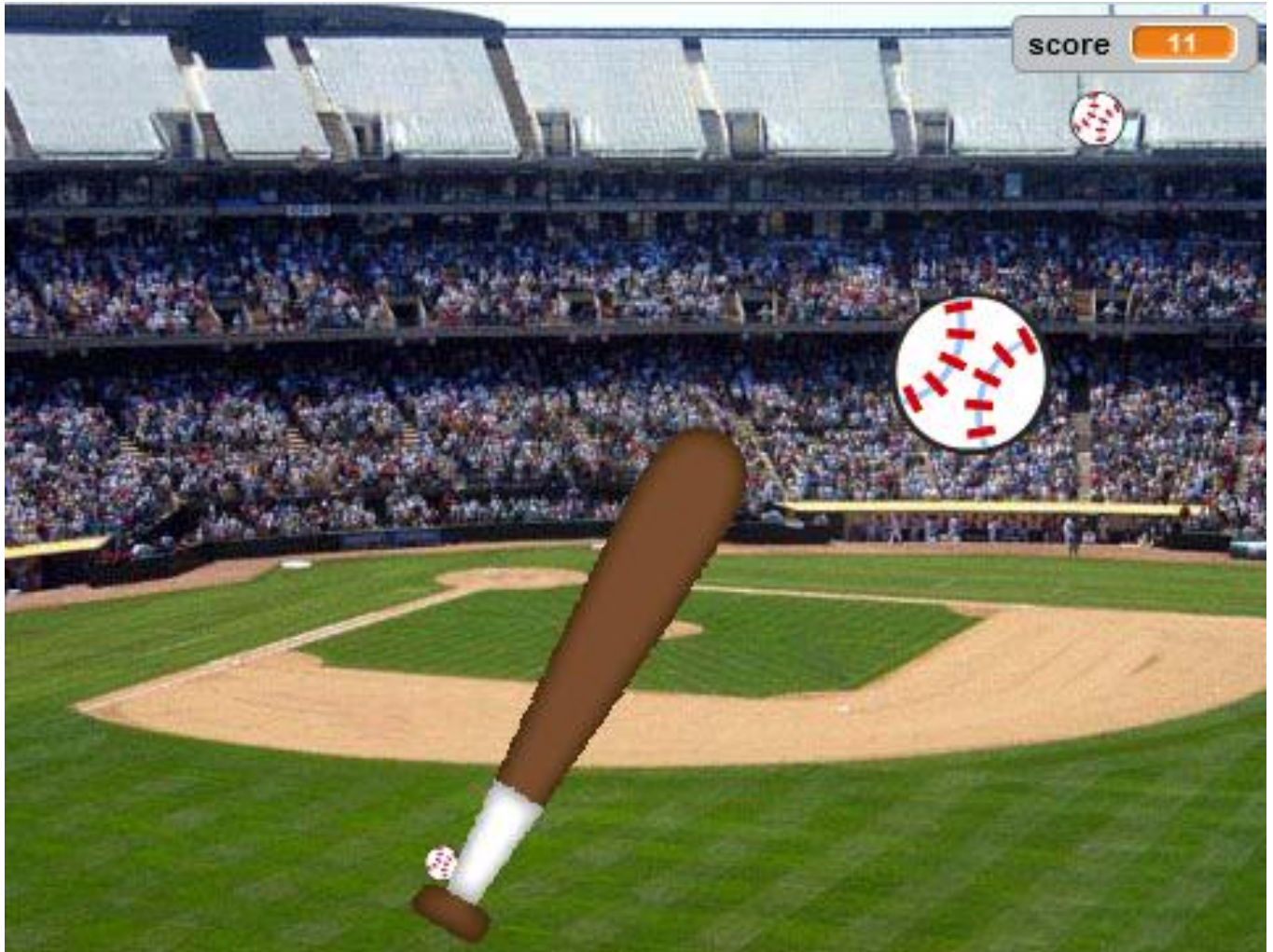
Let's play baseball!

Voorkennis:

Sprites, Lopen, Variabelen, Scores, xy

Leerdoelen:

3D illusie, Klonen



Let's get ready!

Jullie weten ongetwijfeld wat het belangrijkste is van het succes van elk goed spel... Ja, precies, hoe het er uit ziet. Dump die standaard scratch cat en kies een coole baseball achtergrond en een baseball sprite. De honkbalknuppel komt later.

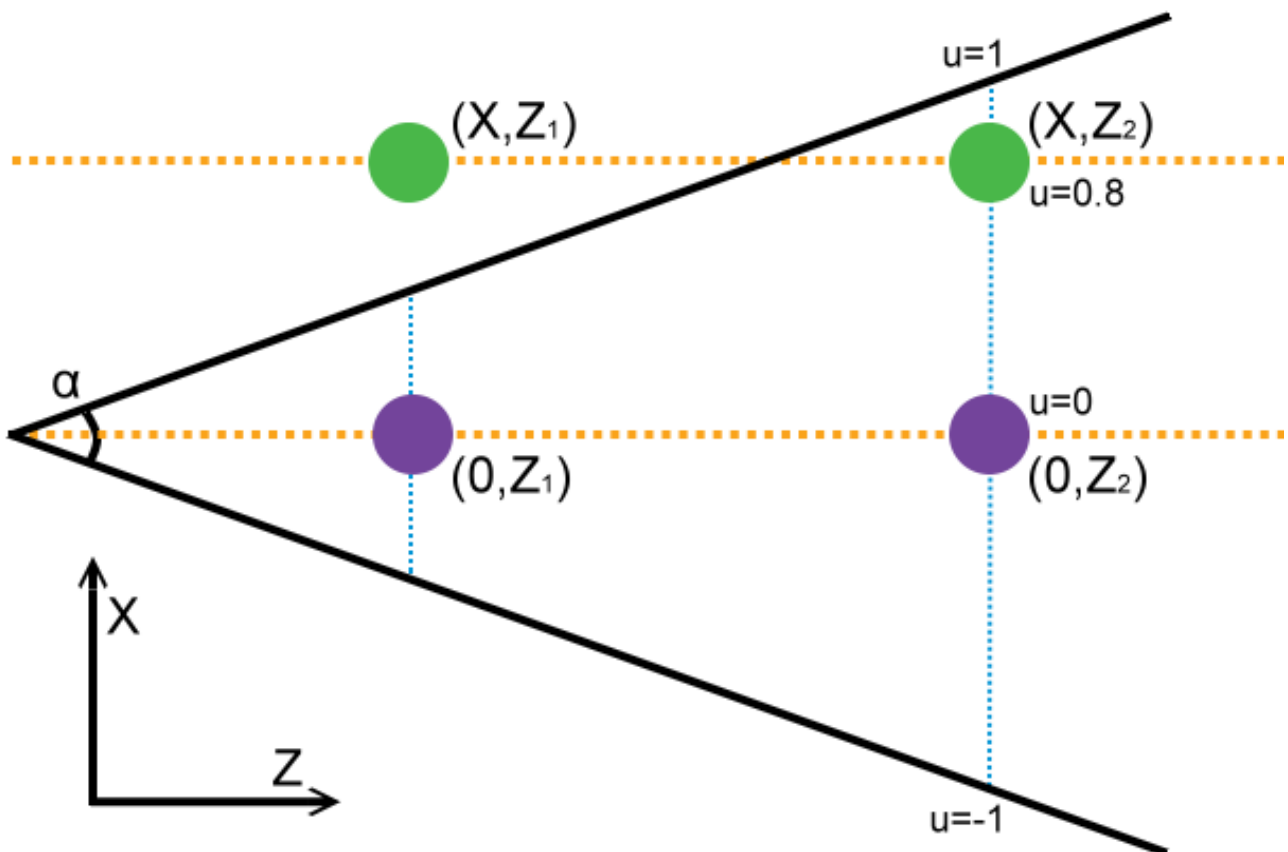
3D illusie

Als je honkbal speelt, en je bent de slagman, dan gooit de pitcher de bal naar je toe, waarna jij de bal moet slaan. Wanneer de pitcher de bal in zijn handen heeft, ziet die bal er vrij klein uit. Immers, de pitcher staat heel ver weg! Wanneer de bal op je afvliegt, dan zie je de bal steeds groter en groter worden. Zo werkt onze 3D wereld: hoe ver een object van je af is, bepaalt hoe groot jouw ogen dat object zien.

Maar Scratch heeft slechts 2 dimensies: x en y. Wij moeten dus zelf een 3D illusie maken!

!!! SLA DIT OVER INDIEN JE JONGER DAN 14 BENT !!!

De 3D illusie kun je je als volgt voor je zien. Jij staat links, bij de hoek α . Wat jij ziet valt in de wig tussen de zwarte lijnen. We beschouwen vier ballen op vier verschillende posities en bekijken hoe wij die zien.



De linker paarse bal bevindt zich dichterbij dan de rechter paarse bal. Je ziet dat de linker bal een veel groter gedeelte van de blauwe stippellijn bestrijkt dan de rechter bal. Oftewel, objecten die dichterbij zijn zie je groter. We willen weten hoe groot de bal is t.o.v. de blauwe stippellijn: dan weten wij hoe groot we de bal in Scratch (2D) moeten tekenen. Daarom noemen we de bovenkant van deze stippellijn $u = 1$ en de onderkant $u = -1$. De bovenste zwarte lijn wordt gegeven door de formule $X = A \cdot Z$ (formule voor rechte lijn).

Beschouw nu een deeltje op $Z = Z_2$ (zoals de groene bal rechts bovenin). Hij bevindt zich op

$$u = \frac{X_{bal}}{X_{zwarte\ lijn}} = \frac{X_{bal}}{AZ},$$

oftewel op een fractie van de afstand tussen de middellijn ($u = 0$) en de zwarte lijn ($u = 1$). Merk op dat de linker groene bal $u > 1$ heeft en dus buiten het scherm valt, terwijl de rechter groene bal zichtbaar is!

Indien wij nu twee X 'n beschouwen, dan weten wij hoe groot wij een bal moeten tekenen:

$$\text{hoe groot wij de bal zien} = \Delta u = \frac{X_{bal\ bovenkant} - X_{bal\ onderkant}}{AZ} = \frac{\text{echte grootte}}{AZ}.$$

Voor de Y-richting geldt precies hetzelfde (vervang alle X door Y).

De eerste formule geeft aan waar de bal getekend moet worden op het scherm.

De tweede formule geeft aan hoe groot de bal getekend moet worden.

In Scratch, gaan we A in een variabele opslaan genaamd "viewAngle".

3D illusie

Diepte

De scratch stage heeft slechts twee dimensies, die we met een x en een y aangeven. Die bepalen de positie van de sprite op het scherm (dus hoe jij hem ziet). Om bij te houden hoe ver de bal van jou vandaan is, maken we een variabele Z aan. Vervolgens kunnen we de bal zo laten bewegen:

```

wanneer vlag wordt aangeklikt
  ga naar x: 0 y: 0
  maak Z -300
  maak snelheidZ 5
  herhaal tot Z > -15
    verander Z met snelheidZ
    wacht 0.01 sec.
  
```

Alleen voor deze sprite!

Maak dit op de achtergrond.

Uiteraard zie je nog niets gebeuren...
Laten we nu de grootte als functie van Z veranderen:

```

verander Z met snelheidZ
  maak grootte 100 / viewAngle * Z %
  wacht 0.01 sec.
  
```

```

wanneer vlag wordt aangeklikt
  maak viewAngle -0.02
  
```

Voor alle sprites!

Test het uit!

Hoogte en Breedte

Maar diepte doet meer dan alleen de grootte veranderen. Objecten die ver weg zijn, zie je op de horizon: je ziet erg veel lucht boven het object en erg veel grond onder het object. Dit komt omdat er ver weg veel meer ruimte is dan dichtbij: er is een andere X en Y ver weg. Laten we ook die variabelen aanmaken voor de bal:

```

verander Z met snelheidZ
  ga naar x: X / viewAngle * Z y: Y / viewAngle * Z
  maak grootte 100 / viewAngle * Z %
  
```

```

ga naar x: 0 y: 0
maak X 0
maak Y 0
maak Z -300
  
```



Speel nu eens met de beginwaarde van X en Y. Wat gebeurt er bijvoorbeeld wanneer X=100?

Is dit logisch? De bal beweegt recht op je af: hij heeft alleen een "snelheidZ", dus X en Y veranderen helemaal niet. Maar toch zie je hem afbuigen! Hoe kan dat???

Het spoor in het linkerplaatje is recht. Toch zie je hem schuin lopen. Dit is precies hetzelfde effect!



Tijd om de bal te slaan!

Poof, de bal verdwijnt.

Wanneer de bal dichtbij genoeg is, willen we hem slaan. Voeg deze code aan je baseball sprite toe:

```

wanneer op deze sprite wordt geklikt
als Z > -45 en Z < -15 dan
  verdwijnt
  maak snelheidZ 5
  verschijnt
  herhaal tot Z > -15

```

Kun je de bal slaan? Dit werkt alleen wanneer de bal nog in beweging is ($Z < -15$).

De bal vliegt weg

In plaats van de bal te laten verdwijnen, willen we de bal *weg slaan*. Vervang “verdwijn” door:

```

maak snelheidZ -5

```

Probeer het uit! De bal vliegt nu weg... Maar hij moet ook verdwijnen wanneer hij ver weg is:

```

maak grootte 100 / viewAngle * Z %
wacht 0.01 sec.
verdwijnt

```

Sneller bij een goede slag!

Leuker is wanneer de bal harder wegvliegt, als je hem beter in het midden raakt. Hiervoor gebruiken we “afstand tot muisaanwijzer”. Deze formule blijkt er redelijke leuk uit te zien:

```

maak snelheidZ -10 * e ^ van 2 / afstand tot muisaanwijzer

```

Hoe pakt dit uit?



Scores

Laten we nu een score bijhouden. Dit kan je zelf!

Uitdaging:

- o Maak een variabele genaamd "score" voor alle sprites.
- o Initialiseer deze op "0" op de achtergrond, net zoals we met viewAngle deden.
- o Wanneer de bal nu geraakt wordt, verander "score" met 1.

Slechts 1 punt per bal

Indien de speler nu heel snel op de bal klikt, dan kan het gebeuren dat hij 2 punten krijgt voor dezelfde bal... Dat willen we natuurlijk niet!

Uitdaging:

- o Maak een variabele genaamd "hit" voor deze sprite. Deze variabele is 0 wanneer de bal nog niet geraakt is, en 1 wanneer de bal al wel geraakt is. False en True dus.
- o Initialiseer "hit" op "0" waar je ook X, Y, Z, etc. initialiseert.
- o Vervang de geraakt-worden conditie door:

```
als hit = 0 en Z > -45 en Z < -15 dan
```

- o Maak "hit" gelijk aan "1" wanneer de bal geraakt wordt.

Beter raken = meer punten

Het is interessanter wanneer je meer punten krijgt, wanneer je de bal precies in het midden slaat:

Natuurlijk niet per ongeluk minpunten geven!



```
verander score met 1
als 25 - afstand tot muisaanwijzer > 0 dan
  verander score met boven van 25 - afstand tot muisaanwijzer
```

Geen komma getallen!

Afstand = 0 → maximale punten



De bal niet recht op je af!

Horizontaal Wegslaan

Op dit moment sla je de bal altijd recht vooruit weg, waar je de bal ook raakt. Laten we hem nu naar de zijkant slaan wanneer je de bal aan de zijkant raakt.

Uitdaging:

Net als voor "snelheidZ", maar een "snelheidX" en een "snelheidY" variabele aan *voor deze sprite*. Doe alles na wat je ook eerder met "snelheidZ" gedaan hebt.

Gebruik nu deze formules wanneer de bal geslagen wordt:

```

maak snelheidX willekeurig getal tussen -3 * afstand tot muisaanwijzer en 3 * afstand tot muisaanwijzer
maak snelheidY willekeurig getal tussen -2 * afstand tot muisaanwijzer en 2.5 * afstand tot muisaanwijzer
maak snelheidZ -10 * e ^ van 2 / afstand tot muisaanwijzer

```

Nu zal de bal ook naar de zijkant bewegen wanneer je hem raakt! Hoe beter je het midden van de bal raakt, hoe rechter hij vooruit zal gaan. Probeer het uit!

Verdwijn aan de Rand

Wanneer je de bal nu weg slaat, kan de bal de rand van het scherm raken. Dan gedraagt de bal zich erg raar! Laten we de bal nu laten verdwijnen wanneer hij het scherm verlaat. Vervang daartoe de herhaalconditie:

```

verschijn
herhaal tot absoluut van x-positie > 239.9 of absoluut van y-positie > 179.9 of Z < -400 of Z > -15
  verander X met snelheidX
  verander Y met snelheidY
  verander Z met snelheidZ

```



Willekeurige Richting Gooien

Speel nu eens met deze getallen:



Uitdaging:

- o Wat gebeurt er wanneer je X verandert?
- o Hoe groot kan X maximaal worden, voordat de bal niet meer op je scherm begint?
- o Hoe groot mag snelheidX worden wanneer je X maximaal kiest, zodat de bal op het scherm blijft?
- o En hoe zit het met Y?
- o Hoe klein/groot kunnen wij snelheidZ maken, zodat het spel niet onmogelijk saai/moeilijk wordt?

Door deze getallen willekeurige waarden te geven, kunnen we de bal op een willekeurige manier gooien. Zo wordt het spel minder voorspelbaar, en dus ook leuker om te spelen!

Door gewoon wat getallen te proberen, blijken deze getallen redelijk mooi te werken:

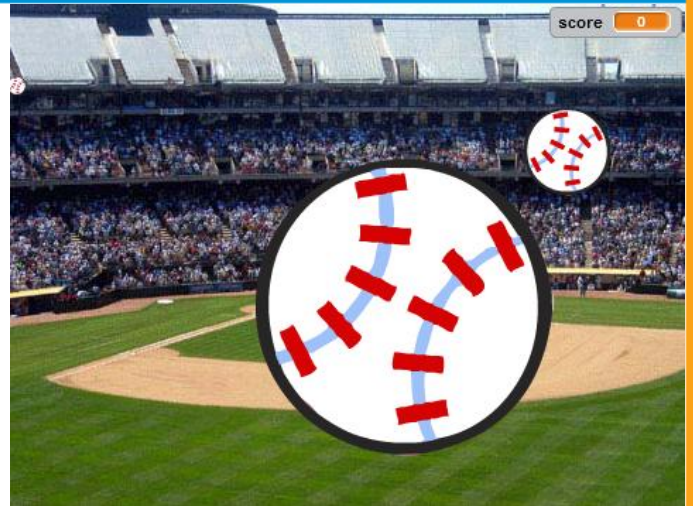


Speel je spel nu eens! Kun je nog winnen?

Meer Ballen! [Scratch 2.0+]

Het spel is pas echt af, wanneer we meerdere ballen naar de speler gooien; net zoals het plaatje op de voorkant. Je zou je sprite een aantal keer kunnen klonen, maar dat is een zeer vervelend klusje...

Makkelijker is om gebruik te maken van deze stukjes code:



Maak de Klonen!



Zet nu helemaal onderaan deze code het “verwijder deze kloon” blok:



Wanneer de kloon klaar is, verwijder hem weer. Immers, hij is niet meer nodig.

En voeg vervolgens dit stukje code toe om klonen te maken:



De echte bal doet niet mee aan het spel!

De echte bal maakt alleen maar klonen van zichzelf. Dit doet hij elke 1 seconden.



De Honkbalknuppel

Om er nu echt honkbal van te maken, voegen we ook een honkbalknuppel toe. De honkbalknuppel kun je downloaden op <http://bit.ly/2ph3YWY>



Volg de Muis

```

wanneer vlag wordt aangeklikt
  richt naar 80 graden
  wacht 0.5 sec.
  herhaal
    ga naar muisaanwijzer
    als muis ingedrukt? dan
      zend signaal batSwing
    wacht 0.05 sec.

```

De honkbalknuppel moet je muiscursor vervangen. Daarom willen we dat hij altijd op de positie van je muisaanwijzer staat.

Wanneer je de muis indrukt, sturen we een signaal "batSwing". Dit signaal zal door de baseball ontvangen worden. De baseball weet dan dat je met de honkbalknuppel (bat) slaat (swing).

Om de baseball het signaal te laten ontvangen:

```

wanneer op deze sprite wordt geklikt
  wanneer ik signaal batSwing ontvang
    als raak ik muisaanwijzer? dan

```

Vervang:

Door:

```

als muis ingedrukt? dan
  draai 15 graden
  ga naar muisaanwijzer
  verander x met 20
  verander y met 10
  wacht 0.05 sec.
  draai 5 graden
  ga naar muisaanwijzer
  verander x met 30
  verander y met 20
  wacht 0.05 sec.
  draai 10 graden
  ga naar muisaanwijzer
  verander x met 30
  verander y met 40
  zend signaal batSwing
  wacht 0.1 sec.
  ga naar muisaanwijzer
  verander x met 20
  verander y met 20
  draai 15 graden
  wacht 0.1 sec.
  draai 15 graden

```

Animeer de Honkbalknuppel

Laten we tot slot een simpele animatie aan de honkbalknuppel mee geven. Het is niet makkelijk om een animatie met een 3D illusie te maken, dus laten we een soort van slag animatie in 2D maken.

← Maak wat links staat na, of:

Uitdaging:

Maak je eigen slaganimatie. De blokken die links zijn gebruikt kunnen van pas komen.



En nu?

Deze opdracht zit er op... Maar je kunt zelf nog veel meer leuke dingen toevoegen!
Zit hier wat leuks voor je bij?

Uitdaging:

De bal ziet er statisch uit als hij op je af komt vliegen. Animeer de bal door hem te laten draaien.

Uitdaging:

In plaats van alleen maar baseballen te gooien, gooi ook eieren!
Wanneer je een ei raakt, krijg je minpunten.

Uitdaging:

Wanneer je een ei raakt, laat het scherm viezer en viezer worden.

Uitdaging:

Maar hoe kun je het spel "winnen"? Nu gaat het oneindig lang door...
Voeg een "win-condition" of een andere voorwaarde toe om het spel te beëindigen.

Enkele ideeën zijn:

- De tijd is op!
- Je bent af wanneer je te veel ballen mist: "three strikes -- out". Dat is pas echt baseball!

Uitdaging:

Maak het spel steeds moeilijker, naarmate de speler meer punten krijgt.
Bijvoorbeeld, gooi meer ballen per seconde of verhoog hun snelheid.
Heb je nog andere interessante ideeën?

Uitdaging:

Maak de honkbalknuppel bestuurbaar met video motion.
Of vervang de honkbalknuppel zelfs door video motion.
Deze blokken kunnen hiervoor handig zijn:

