Zelf Leuke Spelregels Ontwerpen

Riemer van Rozen
rozen@cwi.nl
Centrum Wiskunde & Informatica
Amsterdam, Nederland

SAMENVATTING

Het ontwerpen van spellen is moeilijk en kost heel veel tijd. Om dit eenvoudiger te maken, hebben we de Vie app ontwikkeld [4].

Vie is een visuele programmeertaal waarmee je snel 2D spelletjes kunt ontwerpen [2, 3]. Na een korte introductie krijg je de kans om zelf spelregels te verzinnen en uit te proberen. Om te leren hoe Vie werkt geven we je enkele opdrachten en een handig spiekbriefje.

OPDRACHTEN

Er zijn twee opdrachten, een makkelijke en een moeilijke. We gebruiken de taal Machinations om de spelregels uit te drukken [1]. Daarmee maken we een digitale knikkerbaan die werkt in een spel.

OPDRACHT 1: VIE EN HAAR KONIJN

We beginnen met een voorbeeld over een meisje dat haar konijn kwijt is. Ze heet Vie, net als de app. In Figuur 1 zie je de plaatjes: Vie, Konijnen, Appels en Hartje. Deze spel-elementen gaan we stap voor stap gebruiken in het spelontwerp. Je kunt het direct spelen.

Stap 1: Vie doet mee

Stap 1.1. **Doel.** We beginnen met Vie aan het spel toe te voegen.

Actie. Start de Vie app op. Onder Machinations (links-boven) staat een rondje. Sleep het rondje naar het midden van het scherm.

Resultaat. Er staat nu een rondje in beeld. Dat is een plek waar zich grondstoffen (knikkers) kunnen bevinden. In Machinations heet dat een *pool*. Als er een knikker in zit, dan is Vie er.

Stap 1.2. **Doel.** We zorgen ervoor dat er maximaal één Vie is.

Actie. Links-onder staat de Node Editor waarmee je de pool aanpast. Zet de waarden van de velden At en Max beide op 1.

Resultaat. Wanneer je in de tabs UI Design of Game klikt, zie je dat er een Sprite (een plaatje) en een Label (naam en waarde) zijn toegevoegd. Als Vie er is, dan zie je haar plaatje.

Stap 2: Vie gaat even weg

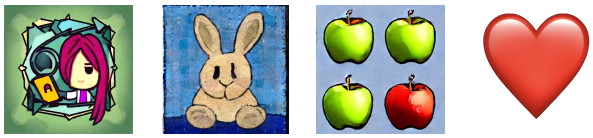
Stap 2.1. **Doel.** We voegen een Ga-regel voor "weggaan" toe.

Actie. Onder Machinations (links-boven) staat ook een driehoek met de punt naar beden. Sleep het rechts van Vie in het scherm. Zet in de Node Editor de waarde van When op "User". Dan wordt het in het spel een interactieve knop waar je op kan klikken.

Resultaat. Er staat nu een driehoekje in beeld. Dat is een plek waar knikkers kunnen verdwijnen. Dit heet een *drain*. Aan de twee lijntjes kun je zien dat de Ga drain een interactief spel-element is.

Stap 2.2. **Doel.** Knikkers hebben een pad nodig om langs te rollen. We voegen een verbinding toe waarlangs Vie weg kan gaan.

Actie. Klik nu op het puntje aan de rechterkant van de Vie pool, en verbind het lijntje met het puntje links van de Ga drain.



(a) Vie **(b) Konijnen** **(c) Appels** **(d) Hartje**

Figuur 1: Spelelementen

(a) Spelregels na Stap 2 **(b) Spelregels na Stap 3**

Figuur 2: De eerste versies van de spelregels

Resultaat. Er staat nu een lijntje tussen Vie en Ga. In Machinations heet dat een *bronverbinding*. De spelregels zien er nu uit zoals in Figuur 2a. De blauwe elementen hebben we in Stap 2 toegevoegd. Er is ook een Ga Button (een knop) toegevoegd. Als erop klikt in de tabs UI Design of Game, dan gaat Vie weg. Probeer maar.

Valsspelen mag ook. Je kunt ook het antwoord van Stap 2 laden. Klik Menu, selecteer "1. Vie (stap 2)", en klik Load.

Stap 3: Vie komt weer terug

Stap 3.1. **Doel.** We voegen een Kom-regel voor "terugkomen" toe.

Actie. Onder Machinations (links-boven) staat ook een driehoekje met de punt naar boven. Sleep dit driehoekje links van Vie in het scherm. Zet daarna in de Node Editor de waarde van When op "User". Dan wordt er ook een knop toegevoegd in het spel.

Resultaat. Er staat nu nog een driehoekje in beeld. Dat is een plek waar knikkers vandaan kunnen komen. Het heet een *source*. Aan de twee lijntjes kun je zien dat Kom een interactief element is.

Stap 3.2. **Doel.** We voegen een pad toe zodat Vie kan terugkomen.

Actie. Klik nu op het puntje aan de rechterkant van de Kom source, en verbind het lijntje met het puntje links van de Vie pool.

Resultaat. Er staat nu een lijntje tussen Kom en Vie. Dat is een bronverbinding. De spelregels zien er nu uit zoals in Figuur 2b. De blauwe regels hebben we in Stap 3 toegevoegd. Als je op de Kom knop klikt in de tabs UI Design of Game komt Vie terug.

Valsspelen mag ook. Je kunt ook het antwoord van Stap 3 laden. Klik Menu, selecteer "1. Vie (stap 3)", en klik Load.

Stap 4: Vie neemt steeds een appel mee

Stap 4.1. **Doel.** We voegen nu Appels toe aan het spel.

Actie. We beginnen weer bij Machinations (links-boven). Sleep nog een rondje naar een een goede plek, bijvoorbeeld onder Kom.

Resultaat. Er staat nu nog pool in beeld. In de tabs UI Design en Game zijn weer een Sprite en een Label toegevoegd.

Stap 4.2. **Doel.** Iedere keer dat Vie terugkomt neemt ze appels mee.

Actie. Klik nu op het puntje aan de rechterkant van de Kom source, en verbind het lijntje met het puntje aan de linkerkant van de Appels pool. Klik daarna op source Kom. Selecteer in de Node Editor voor het veld How de waarde "All" zodat Vie moet komen.

Resultaat. Er is nu ook een bronverbinding tussen Kom en Appels. Als je nu de Kom knop klikt in de tabs UI Design of Game, dan komt Vie terug en neemt ze een appel mee. Probeer maar uit.

Valsspelen mag ook. Je kunt ook het antwoord van Stap 4 laden. Klik Menu, selecteer "1. Vie (stap 4)", en klik Load.

Stap 5: Vie ruilt appels voor haar konijn

Gelukkig kan Vie haar konijn terugkrijgen voor precies vier appels.

Stap 5.1. **Doel.** We voegen Konijnen toe aan het spel.

Actie. We beginnen weer bij Machinations (links-boven). Sleep nog een rondje naar een goede plek. Pas in de Node Editor de waarde van Max aan naar 1 zoadat er maximaal één konijn is.

Resultaat. Er staat nu nog een pool in beeld. Als er knikkers op deze plek komen zijn dat Konijnen. In de tabs UI Design en Game zijn weer een Sprite en een Label toegevoegd om het konijn te zien.

Stap 5.2. **Doel.** We voegen een Ruil-regel toe, en een Ruil-knop.

Actie. Onder Machinations staat ook een driehoekje dat naar rechts wijst met een streep erdoor. Sleep het naar een goede plek.

Resultaat. Er staat nu een nieuw element in beeld. Ruil is een converter. Converters kunnen één soort knikkers kan ruilen voor een andere soort. Er wordt in de tabs UI Design en Game ook een Button toegevoegd die de converter activeert.

Stap 5.3. **Doel.** We voegen toe dat het konijn vier appels kost.

Actie. Voeg een bronverbinding toe tussen de Appel en Ruil. Klik vlak boven het lijntje, midden tussen Appels en Ruil. Vul 4 in.

Resultaat. De nieuwe bronverbinding geeft aan dat de kosten van ruilen 4 appels zijn. Alleen levert ruilen nu nog niets op.

Stap 5.4. **Doel.** We voegen toe dat ruilen een konijn oplevert.

Actie. Voeg een bronverbinding toe tussen de Ruil en Konijnen.

Resultaat. Nu kunnen we appels ruilen voor het konijn. Klik maar op de Ruil converter, of op de Ruil knop in de tabs UI Design of Game. Als je genoeg appels hebt krijg je het konijn.

Stap 6: Eind goed al goed – of iets heel anders!

Een spel is nooit helemaal af, ook Figuur 3 niet. Voeg zelf regels toe en probeer ze uit. Zorg ervoor dat er een hartje in beeld komt, of verzin regels over Bananen of Koeien. Misschien hebben ze honger!

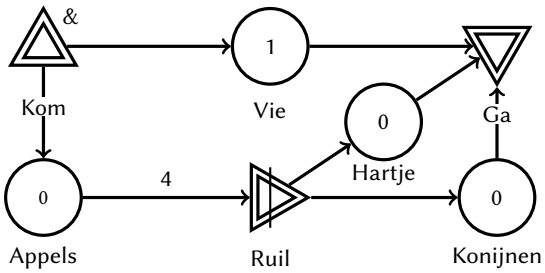
OPDRACHT 2: HET KLIMAAT

Met Vie kun je ook complexe games ontwerpen. In deze vrije opdracht maak je een leerzaam en leuk spel over het klimaat.

Doel. Ontwerp een serious game over het klimaat. Gebruik hierbij de spel-elementen: Warmte, Fabrieken, Bomen en CO2. Wat kunnen we doen om de opwarming van de aarde te voorkomen?

Actie. Klik Menu, selecteer thema "Het Klimaat" en klik Done. Ontwerp spelregels en probeer ze uit. Tip: overleggen en spelen!

Voorbeeld. Je kunt ook een voorbeeld laden. Klik Menu, selecteer "2. Klimaat", en klik Load. Het is nog niet zo'n leuk spel. Misschien beb jij goede ideeën om het te verbeteren?



Figuur 3: Een complete versie van de spelregels

REFERENTIES

[1] Ernest Adams and Joris Dormans. 2012. *Game Mechanics: Advanced Game Design.* New Riders.
[2] Riemer van Rozen. 2023. Cascade: A Meta-Language for Change, Cause and Effect. In *International Conference on Software Language Engineering (SLE 2023).* ACM.
[3] Riemer van Rozen. 2023. Game Engine Wizardry for Programming Mischief. In *International Workshop on Programming Abstractions and Interactive Notations, Tools, and Environments (PAINT 2023).* ACM.
[4] Riemer van Rozen. 2024. Vie app – v0.0.5. (Oct. 2024). https://vrozen.github.io/Vie

SPIEKBRIEFJE

Machinations is een visuele taal voor het ontwerpen van spelregels. Diagrammen (of programma's) zijn grafen die bestaan uit twee soorten elementen: knopen en verbindingen. Beide kunnen worden aangepast met extra informatie. Ze bepalen hoe grondstoffen (knikkers) stap voor stap worden herverdeeld (rollen) via de paden in het diagram (de knikkerbaan). Dit spiekbriefje beschrijft de belangrijkste elementen.

3 Appels	Een *pool* is een knoop met een naam die grondstoffen (knikkers) kan bevatten, zoals munten, kristallen of appels. Een pool ziet eruit als een cirkel met een geheel getal erin dat de huidige hoeveelheid knikkers weergeeft, en de hoeveelheid waarmee een pool begint (at). De maximale capaciteit (max) bepaalt wanneer de pool vol is.
→ / 4 →	Een *bronverbinding* is een verbinding met een bijbehorende hoeveelheid die de snelheid aangeeft waarmee knikkers tussen bron- en doelknopen kunnen stromen (of rollen). Tijdens elke stap kunnen knopen één keer werken, en knikkers herverdelen langs de bronverbindingen van de knikkerbaan. De ingangen van een knoop zijn bronverbindingen aan de linkerkant, en de uitgangen ervan staan rechts.
▽	De *activeringsmodifier* (when) bepaalt of een knoop kan werken. Standaard zijn knopen passief (geen symbool). Interactieve (dubbele lijn) knopen duiden gebruikersacties aan die tijdens een stap een knoop kunnen activeren. Automatische (*) knopen werken vanzelf.
& Kom	Knopen werken (act) ofwel door knikkers langs hun ingangen te trekken (standaard, geen symbool) of door knikkers langs hun uitgangen te duwen (p). Ze werken op twee manieren (how). Knopen die de *any*-modifier hebben (standaard, geen symbool), interpreteren de hoeveelheden van hun bronverbindingen als bovengrenzen, en verplaatsen zoveel mogelijk knikkers. Bij knopen die in plaats daarvan de *all*-modifier (&) hebben, zijn het strikte vereisten, en de bijbehorende stromen gebeuren allemaal of helemaal niet.
△	Een *source*, een knoop die eruitziet als een driehoek die naar boven wijst, is het enige element dat knikkers kan genereren. Sources kan je zien als een pool met een oneindige hoeveelheid knikkers. Ze kunnen daarom altijd voldoende knikkers leveren.
▽	Een *drain*, een knoop die eruitziet als een driehoek die naar beneden wijst, is het enige element waarin knikkers kunnen verdwijnen. Drains kun je zien als pools met een oneindige negatieve hoeveelheid knikkers. Ze kunnen altijd meer knikkers aantrekken.
* →	Een *trigger* is een verbinding die wordt weergegeven als een lijn met een vermenigvuldigingsteken (*). De oorsprongsknoop van een trigger activeert de doelknoop wanneer voor elke bronverbinding waarop de bron werkt, er een stroom is die groter of gelijk is aan die van de bijbehorende hoeveelheid van de bronverbinding.
▷	Converters zijn knopen, weergegeven als een driehoek die naar rechts wijst met een verticale lijn door het midden, die de ene soort knikkers verbruiken en een andere produceren. Converters werken alleen als alle knikkers op de ingangen beschikbaar zijn.