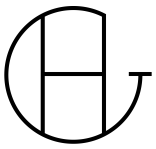


HoGent

Faculteit Bedrijf en Organisatie

Unity 5 versus Unreal Engine 4: Artificiële intelligentie van 3D vijanden voor een HTML5 project

Matthias Caryn

Scriptie voorgedragen tot het bekomen van de graad van
Bachelor in de toegepaste informatica

Promotor:
Joeri Van Herreweghe
Co-promotor:
Steven Delrue

Academiejaar: 2015-2016

Derde examenperiode

Faculteit Bedrijf en Organisatie

Unity 5 versus Unreal Engine 4: Artificiële intelligentie van 3D vijanden voor een HTML5 project

Matthias Caryn

Scriptie voorgedragen tot het bekomen van de graad van
Bachelor in de toegepaste informatica

Promotor:
Joeri Van Herreweghe
Co-promotor:
Steven Delrue

Academiejaar: 2015-2016

Derde examenperiode

Samenvatting

Rusty Bolt is een Belgische indie studio. Deze studio wilt een nieuw project starten voor een 3D spel in een HyperText Markup Language 5 (HTML5) browser die intensief gebruik zal maken van artificiële intelligentie (AI) en Web Graphics Library (WebGL). Na onderzoek via een requirements-analyse van verschillende mogelijkheden van game engines komen we terecht bij twee opties namelijk Unity 5, die Rusty Bolt al reeds gebruikt, of de Unreal Engine 4, wat voor hen onbekend terrein is. Qua features zijn ze enorm verschillend, maar ze voldoen elk niet aan één voorwaarde die Rusty Bolt verwacht van een game engine. Zo biedt Unity Technologies wel een mogelijkheid om software te bouwen in de cloud. De broncode van Unity wordt niet openbaar gesteld, tenzij men er extra voor betaalt. Deze game engine is dus niet volledig open source in tegenstelling tot Unreal Engine 4.

We vergelijken dan verder ook deze twee engines, namelijk Unity 5 en Unreal Engine 4. We tonen aan dat deze engines visueel verschillen van features, maar ook een andere implementatie van de AI hanteren. Zo beperkt Unity zich voornamelijk op *path finding* van de AI, terwijl Unreal Engine 4 daarbij ook *behavior trees* bij betrekt. Technisch zitten er ook verschillen in beide engines. Men kan de Unity engine op meer verschillende systemen draaien dan de Unreal Engine 4. En Unity 5 heeft ook lagere systeemvereisten dan Unreal Engine 4. Unity is verder ook enorm populair en heeft ook een groot marktaandeel, terwijl Unreal Engine 4 de grootste groei toont aan populariteit sinds 2015. Beide game engines gebruiken een verschillende objectgeoriënteerde programmeertaal, Unity 5 maakt gebruik van C# en JavaScript (JS), terwijl Unreal Engine 4 enkel geschreven kan worden in C++ maar hebben wel een eigen visuele scripttaal via *Blueprint Visual Scripting*. Maar als we echter kijken hoe ze hun geschreven code omzetten naar een werkend HTML5 geheel gebruiken beide hiervoor eenzelfde tool gemaakt door Mozilla genaamd emscripten.

In beide game engines maken we ook een basis AI prototype in vergelijkbare omstandigheden en omgeving. Op dat prototype voeren we een aantal tests en metingen uit. Zo komt duidelijk naar voor dat de ruimte die nodig is om een nieuw project aan te maken bij Unreal Engine meer dan vier keer groter is dan voor Unity 5.3. Het laden van de editor gaat bij Unity slechts één seconde sneller. Bij Unity 5 maakt het laden op verschillende browsers, zoals Firefox en Chrome, wel een verschil van ongeveer drie seconden langer wanneer men cache gebruikt. Als we gaan kijken bij Unreal Engine 4 is er nauwelijks tot geen verschil is tussen de snelheden van de verschillende browsers.

Uiteindelijk komen we tot de conclusie dat de engines enorm verschillen en toch hun taak goed volbrengen. De keuze voor een bepaalde game engine is dan ook iets persoonlijk en varieert van project tot project. Deze keuze wordt dan best gecombineerd met de verwachtingen van een game engine in functie van het bedrijf en het project.

Voorwoord

Om een student, in de professionele bachelor toegepaste informatica aan de Faculteit Bedrijf en Organisatie, voor te bereiden op het echte bedrijfsleven wordt men gevraagd een bachelorproef te schrijven. Hierbij leert men onderzoeksmatig informatie te verzamelen en te communiceren over onder andere nieuwe informatie- en communicatietechnologie (ICT) producten. Hierbij worden de informatiebronnen grondig geanalyseerd en aan de hand daarvan wordt een proof-of-concept uitgewerkt.

De student staat vrij om zelf een onderwerp te zoeken en voor te stellen. Als gamer en programmeur was ik dan ook laaiend enthousiast om mij verder te verdiepen in de ontwikkelomgeving voor games. Als potentiële beginnende toekomstige zelfstandige ontwikkelaar is het van belang om de start-up-kosten zo laag mogelijk te houden en dan toch gebruik te maken van gerenommeerde software.

Bij het onderzoek naar game engines bleek dat er heel wat mogelijkheden beschikbaar waren, daarom werden er enkele extra voorwaarden opgesteld in samenspraak met Rusty Bolt om het onderzoek beter te kunnen specificeren. Zo werd de lange lijst toch een deel ingekort. Desondanks de extra voorwaarden bekwam ik toch enkele tientallen game engines. Bij deze engines bekeek ik of deze voldoen aan de must have requirements. De engines die hadden voldaan aan al deze requirements werden verder vergeleken of deze de resterende eisen van Rusty Bolt bevatten. Hierdoor wordt er een korte lijst gecreëerd met uiteindelijk Unity 5 en Unreal Engine 4.

De scope van deze game engines leek me dan ook iets te grotesk, dus besloot ik me samen met Rusty Bolt te focussen op een onderdeel van 3D spelontwikkeling dat me ook enorm interesseert en dat is AI van vijanden. Verder zijn er ook verschillende platformen waarop men een spel kan uitbrengen en vind ik dat ontwikkeling voor het web een heuse vooruitgang heeft geboekt met de komst van HTML5 . Dus kom ik uiteindelijk aan de titel van deze bachelorproef.

Over de gehele lijn stond ik er gelukkig niet alleen voor. Ik werd bijgestaan door het bedrijf Rusty Bolt die tijd voor mij maakte tijdens de ontwikkeling en afwerking van het spel REVOLVE. Deze bachelorproef wordt natuurlijk ook ondersteund door verscheidene docenten van de bachelor toegepaste informatica van Hogeschool Gent. Ik wil hen dan ook allemaal bedanken. Meer specifiek wil ik de promotor Joeri Van Herreweghe en de co-promotor Steven Delrue bedanken.

Inhoudsopgave

Hoofdstuk 1

Inleiding

1.1 Probleemstelling en Onderzoeksvragen

1.1.1 Probleemstelling

Tegenwoordig worden er alom games ontwikkeld, er komen meer en meer start-ups in België die hun steentje daarbij toedragen. Verder zijn games makkelijk te distribueren door de opkomst van smart devices en casual games al dan niet in de browser. Die distributie van spellen naar potentiële spelers is ook verbeterd over de voorbije jaren dankzij online platformen zoals Steam (Matulef, 2016). Belgische studio's kunnen zelfs steun krijgen van het Vlaams Audiovisueel Fonds (VAF) (Tuffin, 2016). Gaming is big business. Er wordt dit jaar zelfs een eerste Belgisch eSports-toernooi gelanceerd (Cludts, 2015) en de inkomsten van de gaming industrie overstijgen zelfs Hollywood (Bloomberg Business, 2015).

Om game development toegankelijker te maken worden er gaming engines ontwikkeld. Één bekende die daar gebruikt van maakt is de Unity Development Engine en is aan zijn vijfde versie toe met nieuwe features (Unity Technologies, 2014). Veel indie developers maken dan ook volop gebruik van deze engine (Coyote, 2013).

Echter maken grote studio's eerder gebruik van een meer geavanceerde engine, namelijk Unreal Engine 4. Sinds 2015 is deze engine ook verkrijgbaar onder een gratis licentie (Sweeney, 2015).

Indie developers zijn ontwikkelaars die onafhankelijk zijn en bestaan meestal uit een klein team die werken met een klein budget, in vergelijking met grote ontwikkelaar studio's. Door de wegvallende start-upkosten is de engine dus interessanter geworden voor indie developers om er gebruik van te maken. Ook wordt 3D gaming in de browser belangrijker met de komst van HTML5 (Pettit, 2013).

1.1.2 Onderzoeksvragen

Unity 5 is de nieuwste versie van de Unity 3D game engine die hedendaags veel wordt toegepast om spellen te ontwikkelen en te deployen naar allerlei platformen (Unity Technologies, 2016k). Unreal Engine 4 is een gevorderde engine die zich ook richt op verschillende technologieën (Epic Games, 2016c). Unity maakt gebruik van HTML5 en WebGL (Unity Technologies, 2015b). Ook zit WebGL verwerkt in Unreal Engine 4 van Epic Games (Epic Games, 2014).

Rusty Bolt is een besloten vennootschap met beperkte aansprakelijkheid (BVBA). En als Belgische ontwikkelingsstudio gevestigd in Beernem (Rusty Bolt, g.d.), zou in de toekomst graag een spel ontwikkelen voor het web die vooral leunt op AI van vijanden. Men overweegt graag overschakelen naar een andere gaming engine zoals Unreal Engine 4.

Deze bachelorproef omvat volgende onderzoeksvragen:

- Welke eigenschappen verwacht Rusty Bolt van een game engine?
- Welke alternatieven zijn er?
- Hoe verschillen de game engines?
- Hoe wordt geschreven code omgezet naar HTML5 en WebGL?
- Welke features bevatten deze engines om AI in een 3D omgeving te implementeren?
- Hoe presteren deze engines verschillend met een zelfgemaakt prototype?

Hoofdstuk 2

Methodologie

Om een beeld te kunnen verkrijgen wat Rusty Bolt verwacht van een game engine werd met hen een requirements-analyse opgesteld. Hierdoor kwam er een lijst met kenmerken naar boven die de studio wel en niet waardeert in een game engine. Daarna werden deze eisen opgedeeld volgens prioriteit aan de hand van een MoSCoW-methode. Via deze informatie onderzocht ik verschillende alternatieven waardoor ik een lange lijst bekwam.

Bij het onderzoek naar game engines bleek dat er heel wat mogelijkheden beschikbaar waren, daarom werden er enkele extra voorwaarden opgesteld in samenspraak met Rusty Bolt om het onderzoek beter te kunnen specificeren. Zo werd de lange lijst deels ingekort. Desondanks de extra voorwaarden bekwam ik toch enkele tientallen game engines. Bij deze engines bekeek ik of deze voldoen aan de *must have* requirements. De engines die hadden voldaan aan al deze requirements werden verder vergeleken of deze de resterende eisen van Rusty Bolt bevatten. Hierdoor wordt er een korte lijst gecreëerd met uiteindelijk Unity 5 en Unreal Engine 4.

Beide engines zijn gewaardeerd in de gaming industrie (Batchelor, 2015) en bevatten veel verschillende features dus een goede literatuurstudie is noodzakelijk. Hiervoor verdiep ik me in de boeken, maar eerst zocht ik natuurlijk enkele vaktermen op online om gerichter te kunnen zoeken. En dat vind je makkelijk op de officiële websites. Via mijn promotor kreeg ik ook wat titels van interessante boeken, maar echte Unity 5 boeken blijken op het eerste zicht geen makkelijke opgave om deze te vinden.

Een boek met algemene informatie bestaat niet echt. De boeken specificeren zich meestal op één bepaald onderwerp gelinkt aan de engine. Dus eerst zoeken we wat algemene informatie op over Unity 5 en Unreal Engine 4. Het onderwerp is vrij specifiek, dus gerichte informatie zou makkelijk moeten zijn. Echter is niets minder waar. De zoektocht is moeizaam. Heel véél informatie is te vinden online en in boeken, maar weinig die de focus leggen op het HTML5 gedeelte, iets wat ik hier kan onderzoeken. De grootste tijd gebruik ik dan ook uitvoerig naar het achterhalen van bronnen en informatie.

Veel informatie die ik ook vond, was jammer genoeg snel gedateerd. De recentste informatie verloopt dan ook snel doordat de engines verschillende updates kregen doorheen de bachelorproef. Zo kreeg in de loop van de zoektocht, de Unity 5 engine een officiële ondersteuning van de WebGL standaard zoals vermeld in sectie 4.2. Toch heb ik redelijk wat algemene informatie kunnen vergaren en kan ik daarmee aan de slag om mijn bevindingen neer te pennen.

Via al deze informatie werd ook een basis AI scenario bedacht die de vijanden AI zal weergeven in het spel. Ik maakte dan ook een versie van verschillende scenario's in Unity 5 en Unreal Engine 4. Dit prototype werd zowel in de Unity engine als in Unreal Engine gemaakt met zoveel mogelijk basismateriaal en duidelijke voorwaarden om zo een gelijkend geheel te bekomen. Hierbij werden ook enkele metingen uitgevoerd om de verschillen in performantie te kunnen aantonen bij zowel de game engines alsook bij verschillende browsers. Ook werden meting uitgevoerd die meer nadruk leggen op de AI zelf. Uiteindelijk bekomen we dan het geheel dat antwoord geeft aan alle vragen die dit onderzoek stelde.

Hoofdstuk 3

Bedrijfscase

Een kleine indiestudio, Rusty Bolt, twijfelt om de overstap te nemen van Unity naar Unreal Engine. Hiervoor gaan we op onderzoek uit welke eigenschappen voor Rusty Bolt belangrijk zijn in een softwarepakket om games te ontwikkelen. Rusty Bolt bestaat uit één persoon die full time programmeert en ontwikkelt aan het spel. Verder zijn er nog andere personen die helpen met het design en geluid van het spel.

3.1 Analyse

3.1.1 Requirements-analyse

De eisen werden besproken met Rusty Bolt en werden alvast opgedeeld in functionele eisen en niet-functionele eisen. Na elke eis wordt er wat extra informatie toegelicht, eventueel gevolgd door een voorbeeld.

- Functionele eisen
  - Ondersteuning voor HTML5 : De engine ondersteunt officieel HTML5 in een versie dat in productie is.
  - Ondersteuning voor ontwikkeling in 3D: De engine ondersteunt officieel 3D omgevingen in een versie dat in productie is.
  - Ondersteuning voor WebGL : De engine ondersteunt officieel WebGL in een versie dat in productie is.
  - Ondersteuning van objectgeoriënteerde taal zoals C++ of C#: Er is een mogelijkheid om games te programmeren met behulp van C++ of C#.
  - Integratie mogelijk van advertenties: Een tool om advertenties van bijvoorbeeld Google te implementeren.

  - Mogelijkheid om te debuggen: Een tool die toegewijd is om een applicatie te debuggen.
  - Een eigen winkel om artefacten (gratis) aan te kopen: Er is een winkel aanwezig waarbij men bijvoorbeeld scripts, afbeeldingen of andere bronnen (gratis) kan kopen. Eventueel kan men deze ook verkopen tegen een prijs.
  - Mogelijkheid om te werken in een cloudomgeving: Een tool die toegewijd is om bijvoorbeeld een *build* uit te voeren in de cloud.
  - Tool om de AI op te volgen: Een tool die toegewijd is om te zien in welke staat een zich een AI bevindt.
  - Tool voor toegang met de AI : Een tool die toegewijd is om variabelen van de AI aan te passen.
  - Integratie met source management systemen: Integratie van bijvoorbeeld Git.
  - Importeren van artefacten, tools en formaten zoals een Photoshop Document (PSD): Een manier om bijvoorbeeld texturen in te laden die PSD ondersteunen, maar ook bijvoorbeeld Joint Photographic Experts Group (JPEG) of Portable Network Graphics (PNG).
  - Aanpassen van artefacten: Een tool om de artefacten rechtstreeks in de engine aan te passen.
  - Ondersteuning van eigen scripts: Eigen scripts kunnen (her)gebruiken om functies toe te voegen aan de engine.
  - Bevat een eigen *physics* implementatie: De engine heeft een eigen implementatie van *physics* zoals bijvoorbeeld wanneer twee objecten tegen elkaar botsen.
  - Bevat een *path finding* implementatie: De engine heeft een eigen implementatie van *path finding* zoals bijvoorbeeld een manier om de sneltste route van punt A naar B te berekenen.
  - Bevat een manier om te playtesten: Een manier om het spel te spelen en zo test resultaten te bemachtigen tijdens het spelen.

- Niet-functionele eisen
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200: Daarmee bedoelen we de prijs van het complete software pakket om een basis te implementeren. Dit bevat ook de support gegeven door ontwikkelaar, maar zonder een extra snellere en meestal betalende responstijd.
  - Moet open source zijn: De broncode is kosteloos te verkrijgen.

  - Heeft een actieve community: Een actieve community is een community die wekelijks vragen heeft en antwoorden verzorgt aan andere community-leden op de officiële verzorgde kanalen zoals bijvoorbeeld een forum.
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten: Een documentatie waarbij er voorbeeldprojecten en eventuele testen kosteloos te bekijken zijn via een officieel kanaal van de ontwikkelaar.
  - Support door ontwikkelaar: Dit kan rechtstreeks bekomen worden via een officieel kanaal van de ontwikkelaar, mag extra kosten. Dit kan via incident management, telefoon, e-mail of andere communicatiekanalen. Enkel een contactformulier op een website wordt niet gezien als support door de ontwikkelaar, tenzij expliciet vermeld.
  - Moet instelbaar zijn naar eigen wensen: Men kan de engine aanpassen naar eigen wensen zoals bijvoorbeeld de kleuren van de applicatie, grootte van het lettertype.

3.1.2 MoSCoW-methode

De opgestelde eisen werden dan ingedeeld volgens prioriteit via de MoSCoW-methode. Daarin komt duidelijk naar boven welke features er moeten inzitten (*must have*), welke er zouden moeten inzitten (*should have*) en welke leuk zouden zijn om te hebben (*could have*) en welke features er niet moeten inzitten (*won't have*) (Waters, 2009).

- Moeten er zeker inzitten (*must have*):
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- Zou er moeten inzitten (*should have*):
  - Integratie met source management systemen
  - Tool om de AI op te volgen

  - Tool voor toegang met de AI
  - Importeren van artefacten, tools en formaten zoals een PSD-bestand
  - Aanpassen van artefacten
  - Ondersteuning van eigen scripts
  - Bevat een eigen *physics* implementatie
  - Bevat een *path finding* implementatie
  - Bevat een manier om te playtesten
  - Heeft een actieve community
- Leuk om te hebben (*could have*):
  - Mogelijkheid om te werken in een cloudomgeving
  - Integratie mogelijk van advertenties
  - Moet open source zijn
  - Een eigen winkel om artefacten (gratis) aan te kopen
  - Moet instelbaar zijn naar eigen wensen

3.2 Alternatieven

Via de MoSCoW-methode van sectie 3.1.2 hebben we alvast de *must have* eisen op een rijtje gezet. Dit resulteert in een lange lijst in sectie 3.2.1. Deze wordt later korter in sectie 3.2.2 door te kijken naar de *should have* eisen van de engines, die in de lange lijst voldaan zijn aan alle *must have* eisen.

3.2.1 Lange lijst

De lange lijst bevat alle game engines die werden gevonden en voldoen aan de volgende voorwaarden vastgelegd in samenspraak met Rusty Bolt. Zo moet het spel gemaakt kunnen worden op besturingssysteem Windows 10 of hoger. De prijs wordt gerekend inclusief support door de ontwikkelaar. De ontwikkeling van de engine is reeds actief en heeft in 2015 nog een update doorgevoerd. De engine is verkrijgbaar in een stabiele release voor consumenten en voor datum 1 juli 2016. Indien mogelijk wordt een release nummer toegevoegd. Via deze extra voorwaarden wordt de zoektocht en lijst ingekort. Om te voldoen aan deze eisen wordt er enkel gekeken naar de informatie die beschikbaar is op de officiële website van de game engine zonder een registratie.

Voldoen aan alle *must have* eisen

Deze game engines voldoen aan alle eisen die er zeker moeten inzitten alsook aan de extra voorwaarden. Deze engines worden alfabetisch gerangschikt.

- Blend4web 16.07
  - Officiële website: https://www.blend4web.com
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- Unity 5.4
  - Officiële website: https://unity3d.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- Unreal Engine 4.12
  - Officiële website: https://www.unrealengine.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL

  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen

De minder bekende game engine van de drie is Blend4web 16.07. Deze game engine is een suite die naast Unity 5.4 en Unreal Engine 4.12 voldoet aan alle eisen die in een game engine moeten zitten volgens Rusty Bolt.

Voldoen aan bijna alle *must have* eisen

Deze game engines voldoen aan bijna alle eisen die er zeker moeten inzitten. Enkel 1 of 2 eisen worden niet voldaan.

- Unigine 2
  - Officiële website: https://unigine.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- GameMaker: Studio 1.4.1757
  - Officiële website: http://www.yoyogames.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar

  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen

- Amazon Lumberyard v1.3.0.1
  - Officiële website: https://aws.amazon.com/lumberyard/
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen

- Turbulenz
  - Officiële website: http://www.turbulenz.biz/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen

De prijs van de Unigine 2 is te duur en voldoet dus niet aan de maximale kostprijs. GameMaker ondersteunt geen 3D omgevingen. Turbulenz heeft geen support van de ontwikkelaar en ondersteunt geen C++. Amazon Lumberyard ondersteunt dan weer geen HTML5 en WebGL.

Voldoen niet aan alle *must have* eisen

Er werden nog een pak andere engines onderzocht. Deze voldoen niet aan 3 of meer *must have* eisen. Deze lijst is te vinden in bijlage A.

3.2.2 Korte lijst

Zoals gezien in sectie 3.2.1 zijn er slechts drie alternatieven die voldoen aan de *must have* eisen. We gaan dan ook verder met deze drie alternatieven en bekijken of ze voldoen aan de *should have* requirements.

Voldoet aan alle *should have* eisen

Deze game engines voldoen aan alle eisen die er moeten en zouden moeten inzitten alsook aan de extra voorwaarden. Deze engines worden alfabetisch gerangschikt.

- Unity 5.4
  - Integratie met source management systemen
  - Tool om de AI op te volgen
  - Tool voor toegang met de AI
  - Importeren van artefacten, tools en formaten zoals een PSD
  - Aanpassen van artefacten
  - Ondersteuning van eigen scripts
  - Bevat een eigen *physics* implementatie
  - Bevat een *path finding* implementatie
  - Bevat een manier om te playtesten
  - Heeft een actieve community
- Unreal Engine 4.12
  - Integratie met source management systemen
  - Tool om de AI op te volgen
  - Tool voor toegang met de AI
  - Importeren van artefacten, tools en formaten zoals een PSD
  - Aanpassen van artefacten
  - Ondersteuning van eigen scripts
  - Bevat een eigen *physics* implementatie
  - Bevat een *path finding* implementatie
  - Bevat een manier om te playtesten
  - Heeft een actieve community

Zowel Unity 5.4 als Unreal Engine 4.12 voldoen aan alle *should have* eisen.

Voldoet niet aan alle *should have* eisen

Deze game engines voldoen aan alle eisen die er moeten alsook aan de extra voorwaarden maar niet aan alle requirements die er zouden moeten inzitten.

- Blend4web 16.07
  - Integratie met source management systemen
  - Ondersteuning van eigen scripts
  - Bevat een eigen *physics* implementatie
  - Heeft een actieve community

Blend4web 16.07 bevat eigenlijk weinig features die er volgens Rusty Bolt zouden moeten inzitten.

De *could have* eisen

Zoals te zien in sectie 3.2.2 voldoen enkel Unreal Engine 4.12 en Unity 5.4 aan alle eisen die er zouden moeten inzitten volgens Rusty Bolt. We bekijken dan ook enkel van deze twee game engines de eisen die leuken zouden zijn mochten die er inzitten en bekomen volgende resultaten:

- Unity 5.4
  - Mogelijkheid om te werken in een cloudomgeving
  - Integratie mogelijk van advertenties
  - Een eigen winkel om artefacten (gratis) aan te kopen
  - Moet instelbaar zijn naar eigen wensen
- Unreal Engine 4.12
  - Integratie mogelijk van advertenties
  - Moet open source zijn
  - Een eigen winkel om artefacten (gratis) aan te kopen
  - Moet instelbaar zijn naar eigen wensen

Ook hier gaat het gelijk op en voldoen ze beiden niet aan alle eisen. Zo moet men betalen om de source code van Unity 5.4 te achterhalen. Unreal Engine 4.12 heeft dan echter geen mogelijkheid om te werken in een cloud omgeving.

Hoofdstuk 4

Unity

Unity, ook gekend onder de naam Unity3D, is een software-ontwikkelomgeving geproduceerd door Unity Technologies voor interactieve media zoals 2D en 3D games (Unity Technologies, 2016j). Het ondersteunt verscheidene platformen op smartphones, desktopcomputers, virtual realiteit (VR), consoles, televisie platformen en web (Unity Technologies, 2016k). Unity wordt gebruikt door indie developers tot grote studio's zoals Blizzard Entertainment voor het produceren van Hearthstone: Heroes of Warcraft (Chayes, 2014), een succesvolle populaire online collectable card game (CCG) (Crecente, 2015). Verder kan een goed spel ook een boost aan inkomsten geven aan Unity Technologies, zoals bijvoorbeeld recent werd ondervonden met de zomerhype van Pokémon Go op mobiele apparaten (Wingfield, 2016).

4.1 Geschiedenis

Na 4 jaar in ontwikkeling door Unity Technologies werd op 6 juni 2005 (Helgason, 2005) Unity 1 gelanceerd op de Apple Worldwide Developers Conference (WWDC) (Unity Technologies, 2016i). Twee jaar later werd een opvolger aangekondigd onder de naam Unity 2.0 op 10 okotober 2007 (Unity Technologies, 2016i). In 2008 werd Unity iPhone gelanceerd voor iPhone en kondigde Unity Technologies aan dat ze een geautoriseerde middleware provider worden van de Wii (Unity Technologies, 2016i). Een jaar later werd een gratis licentie voor Unity aangeboden aan het publiek (Unity Technologies, 2016i). Op 27 september 2010 werd de 3e editie aangekondigd onder de naam Unity 3 (Marketwired L.P., 2010a), vijf jaar na de eerste versie. Ook werd dat jaar Unity voor Android gelanceerd (Unity Technologies, 2016i) en kwam er een *native client* voor Chrome, onder de naam Unity Native Client (NaCL) (Seyler, 2010). Tegen het einde van 2010 werd ook de kaap van 250 duizend ontwikkelaars overschreden (Marketwired L.P., 2010b).

In 2011 werd de 100ste werknemer aangenomen (Unity Technologies, 2016i). Op 18 juni 20112 kondigde Unity Technologies een nieuwe en vierde generatie van de engine aan (Marketwired L.P., 2012a) samen met de mogelijkheid om games op Linux en Adobe Flash te publiceren (Unity Technologies, 2016i) en steeg het aantal ontwikkelaars naar één miljoen (Marketwired L.P., 2012b). Een 2D tool werd pas in 2013 toegevoegd aan de engine (Marketwired L.P., 2013). De recentste versie, genaamd Unity 5, verscheen in 2014 (Unity Technologies, 2014).

4.1.1 Populariteit

Unity wordt hoog gewaardeerd in de industrie. De engine stond op de derde plaats in 2015 op *The Tech List*. Verder heeft ook Unity een groot aandeel op de markt, zo heeft het marktaandeel van 45 procent wereldwijd (Tnw Deals, 2016). Dit is 28 procent meer dan de dichtstbijzijnde competitie (Tnw Deals, 2016). Ook werd in 2014 Unity verkozen als populairste ontwikkelingssuite in het Verenigd Koninkrijk met maar liefst 62 procent (Tnw Deals, 2016). In juli 2014 werden dan meer dan 10000 ontwikkelaars ondervraagd welke tools men gebruikt (Tnw Deals, 2016). Daarbij koos 47 procent voor de game engine van Unity Technologies (Tnw Deals, 2016). Sinds 2015 zijn er meer dan 4 miljoen geregistreerde waaronder bekende studio's, van onder andere Disney, Electronic Arts en Microsoft, die gebruik maken Unity (Unity Technologies, 2016i). De grootste inkomsten van mobiele 3D spellen op de Chinese, Japanse, Koreaanse en Amerikaanse markt komen van spellen gebouwd door middel van de Unity engine (Unity Technologies, 2016i).

4.2 Unity 5

Unity 5 kwam op 3 maart 2015 op de markt onder twee edities, een professionele en persoonlijke editie (Unity Technologies, 2015a). In Unity 5 kan je gebruik maken van verschillende programmeertalen waaronder C#, JS en Boo (Haas, 2014), al wordt deze laatste programmeertaal minder en minder ondersteund door het weinige gebruik ervan (Aleksandr, 2014). Momenteel zit men aan versie 5.4 en heeft men hiervoor officieel ondersteuning voor WebGL, die in de eerste versie van Unity 5 nog maar in preview was (Jarvis, 2015).

4.2.1 Edities

Er zijn vier edities gericht van Unity 5, gericht op zowel kleine en grote ontwikkelaarsstudio's, waarvan één gratis, versie, genaamd Unity Personal Edition waarbij je tot $100.000 aan jaarlijkse inkomsten kan verdienen op 25 verschillende platformen (Takahashi, 2015). Hierbij moet men geen auteursrechten op betalen (Takahashi,

2015). Als men toch meer dan $100.000 verdient, maar minder dan $200.000, kan men overschakelen op de Unity Plus Edition voor een bedrag van $35 per maand per ontwikkelaar (Unity Technologies, 2016m). Hierbij krijgt men enkele extra voordelen zoals betere analytische informatie (Unity Technologies, 2016m). Indien er meer dan $200.000 jaarlijks wordt verdient of wanneer men meer features nodig heeft, kan men dan kiezen voor een licentie op de Unity Professional Edition, die verschillende versies aanbiedt (Takahashi, 2015). De prijs voor de professionelere versie start vanaf $125 per maand (Unity Technologies, 2016m). Deze editie bevat extra features en extra hulpmiddelen zoals Unity Cloud Build Pro, Unity Analystics Pro, Unity Assets Store Level 11 Deals en nog veel meer (Nutt, 2016). Via deze versie krijgt men ook toegang tot de broncode van Unity 5 (Unity Technologies, 2016m). Als laatste is er ook nog een Unity Enterprise Editie waarbij men iets samenstelt naar de wensen van het bedrijf zelf (Unity Technologies, 2016m). Naast deze vier edities zijn er ook nog minder bekende versies voor ondernemingen, onderwijs of een specifieke industriesector (Unity Technologies, 2016c).

4.2.2 Features

Een belangrijk onderdeel van de 5e versie is, zoals eerder genoemd in het begin van deze sectie 4.2, de toevoeging van WebGL waarbij het mogelijk wordt om games uit te brengen op browsers, zoals Chrome en Firefox, zonder een invoegtoepassing (Masters, 2015). Een andere grote en gewaardeerde toevoeging is het aanleveren van een 64-bit editor om games met te ontwikkelen, zodat ontwikkelaars nu ook meer Random Access Memory (RAM) geheugen kunnen gebruiken (Masters, 2015). Verder zijn er natuurlijk nog andere grote veranderingen gebeurd voor zowel 2D als 3D omgevingen, zoals verbetering in de schaduwen, verlichtingspunten, audio, *physics* en nog zoveel meer (Masters, 2015).

4.2.3 Systeemvereisten

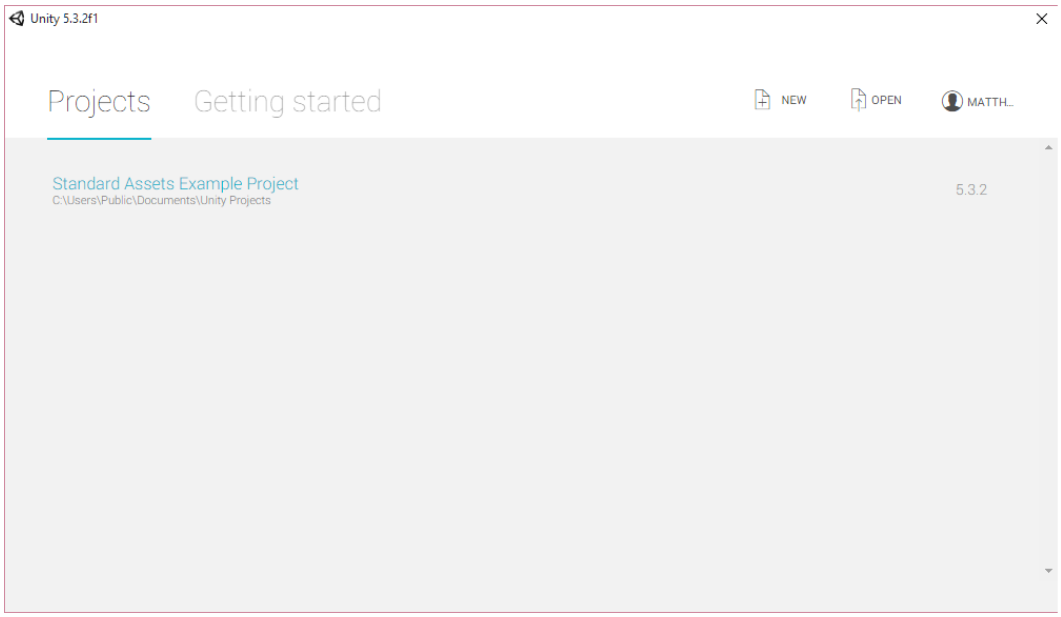
Om te kunnen ontwikkelingen op Unity 5 met een computersysteem dat men gebruikt, moet dit systeem natuurlijk compatibel zijn met de systeemvereisten. Zo worden Windows 7 Service Pack 1 of hoger, Windows 8 en Windows 10 ondersteunt, alsook kan men gebruik maken van het besturingssysteem Mac operating system (OS) X 10.8 of hoger (Unity Technologies, 2016h). Het spel spelen kan wel op andere en oudere besturingssystemen zoals Windows XP Service Pack 2 of hoger, Mac OS X 10.8 of hoger, Ubuntu 12.04 of hoger en SteamOS (Unity Technologies, 2016h). De graphics processing unit (GPU) moet dan gebruik kunnen maken van DirectX9 (shader model 3.0) of DirectX11 met feature level 9.3 (Unity Technologies, 2016h). Voor WebGL is een recente browser die HTML5 ondersteunt nodig zoals Chrome, Firefox, Edge of Safari (Unity Technologies, 2016h).

4.2.4 Omgeving

In dit project maken we gebruik van Unity 5.4.0f3 en starten we de Unity 5 Personal Edition applicatie op het Windows 10 besturingssysteem. We maken gebruik van de 64-bits editor van Unity 5 waarop we reeds zijn ingelogd.

Starten

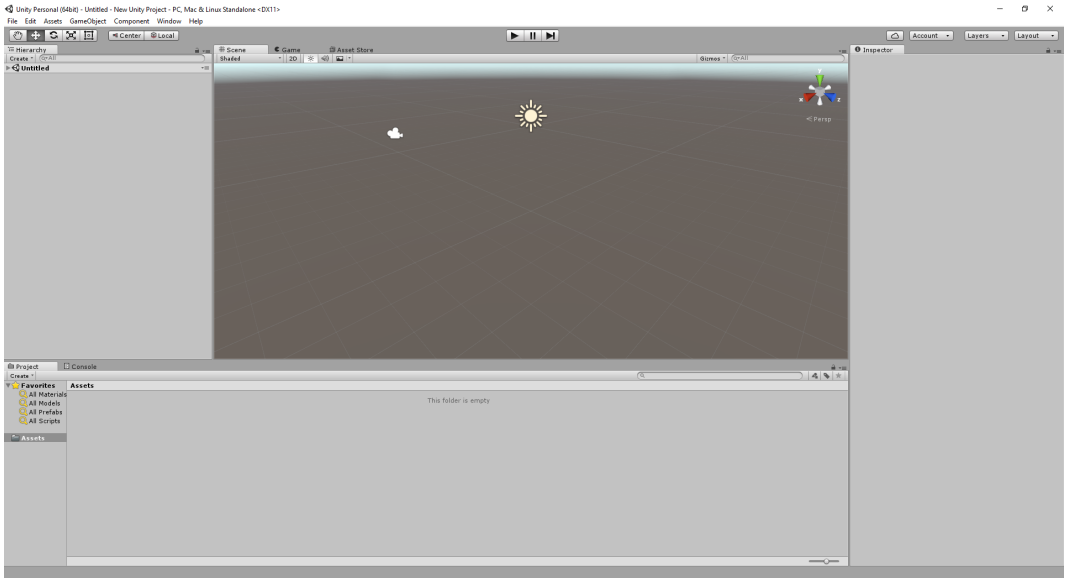
Tijdens het starten van de applicatie krijg je een overzicht van uw projecten en hun titels, zoals te zien in figuur 4.1, samen met de locatie van de mappen. Ook wordt vermeld op welke Unity versie deze projecten zitten. Je kan bovenaan kiezen om een nieuw project aan te maken of een bestaand project te openen. Verder zie je rechtsboven ook uw accountnaam. Wanneer je erop klikt heb je verschillende mogelijkheden om aanpassingen te doen. Via *Getting Started* krijg je een video te zien met algemene informatie. We kiezen hier voor het voorbeeldproject.



Figuur 4.1: Starten van Unity 5 op Windows 10

Editor

Nadat de editor is ingeladen, krijgen we het beginscherm van de editor beschikbaar, zoals weergeven in figuur 4.2. Bovenaan de Unity 5 editor bevindt zich de *Toolbar* die allerhande sneltoetsen bevat naar functies voor onder andere het transformeren van de *Scene View* dat zich in het midden bevindt. Links ervan is het *Hierarchy Window* waar alle game objecten zich bevinden in de huidige *Scene*. Helemaal rechts is dan het *Inspector View* waarbij je eigenschappen van de objecten kunt beïnvloeden. Onderaan bevindt zich het *Project Windows* waar je objecten en materialen kunt beherend die toebehoren tot het project.



Figuur 4.2: Editor van Unity 5 op Windows 10

AI hulpmiddelen

De AI hulpmiddelen van Unity zijn voornamelijk gebaseerd op navigatie en *path finding*. Als we bij Unity 5 kijken om AI te implementeren zien we voornamelijk dat ze oplossingen geven op twee problemen (Unity Technologies, 2016d):

- Hoe redeneert een agent zich om een plaatst te vinden (Unity Technologies, 2016d)
- Hoe verplaatst een agent zich naar deze plaats (Unity Technologies, 2016d)

Een agent is hierbij een karakter dat zichzelf verplaatst op een intelligente manier (Unity Technologies, 2016d). De beloopbare ruimtes waar een agent (*NavMesh Agent*) zich kan begeven, wordt in Unity een *NavMesh* genoemd (Unity Technologies, 2016d).

De ruimtes worden in verschillende convexe polygonen verdeeld en samen met de buren opgeslagen (Unity Technologies, 2016d). Men gebruikt deze voorstelling omdat er geen obstructie is tussen twee punten in een polygoon (Unity Technologies, 2016d). Om de beste manier te vinden van startpositie naar eindpositie gebruikt Unity het *A** algoritme (Unity Technologies, 2016d). Dit wordt berekend aan de hand buren van de polygonen (Unity Technologies, 2016d). De opeenvolgende polygonen die een pad beschrijven worden een gang genoemd (Unity Technologies, 2016d). De agent zal steeds de eerstvolgende zichtbare gang nemen (Unity Technologies, 2016d). Per frame zal één enkele agent slechts weinig bewegen en kan de gang steeds opnieuw berekend worden wanneer er zich bijvoorbeeld een andere agent dichtbij bevindt (Unity Technologies, 2016d). Obstakels (*NavMesh Obstacles*) kunnen beweegbare objecten zijn maar ook andere agenten (Unity Technologies, 2016d). Om obstakels te ontwijken zal de snelheid en richting bijgestuurd worden (Unity Technologies, 2016d). Unity zal dit herberekenen om zo eventuele botsingen te voorspellen of om de botsingen te voorkomen (Unity Technologies, 2016d). Na al deze calculaties zal Unity een ideale eindsnelheid hebben voor het navigatiesysteem van de agent waardoor de agent verplaatst zal worden binnenin de *NavMesh* (Unity Technologies, 2016d).

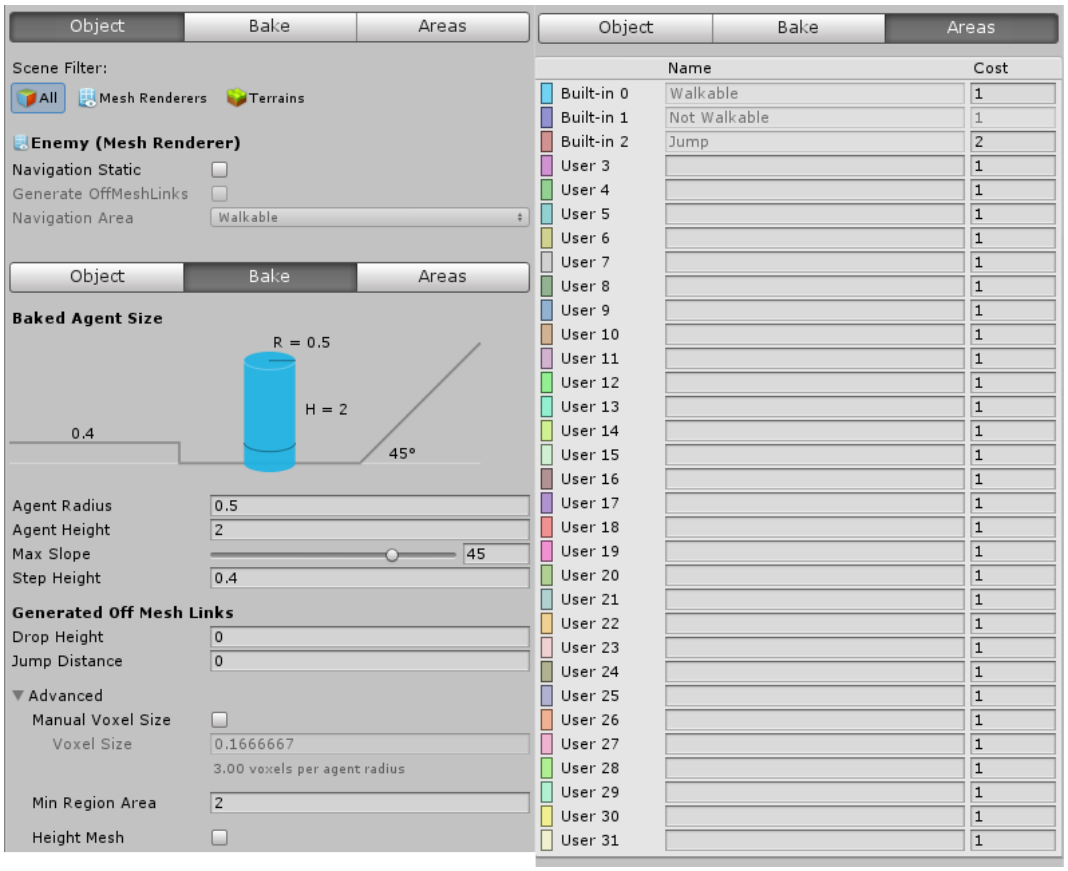
De *NavMesh* zoekt globaal het pad via gangen aan de hand van locaties op de map (Unity Technologies, 2016d). Zo berekent men de mogelijkheid om zich rond statische objecten te begeven (Unity Technologies, 2016d). Deze globale navigatie vraagt veel rekenkracht en geheugen (Unity Technologies, 2016d). Lokaal wordt dan gekeken hoe men zich efficiënt kan verplaatsten door een gang (Unity Technologies, 2016d). Eventueel wordt de agent bijgestuurd om een bewegend obstakel te vermijden (Unity Technologies, 2016d). Erna zal de agent dan ook effectief verplaatst worden en de eerstvolgende beschikbare gangen opnieuw te bereken (Unity Technologies, 2016d). Soms is het echter nodig om een agent over ruimtes te verplaatsen die niet beloopbaar (Unity Technologies, 2016d). Hiervoor gebruikt men dan een *Off-Mesh Link* (Unity Technologies, 2016d). Om een NavMesh te maken, gebruikt Unity 5 een methode genaamd *NavMesh Baking* (Unity Technologies, 2016a). Deze methode maakt de *NavMesh* aan van de beloopbare ruimte door gebruik te maken van de gerenderde meshes en omgevingen van alle statische objecten (Unity Technologies, 2016a). Via *Navigation Areas* kan men bepalen hoe moeilijk het is om een ruimte te belopen door er een bepaalde kost aan vast te hangen. Zo is het moeilijker om water te doorlopen dan de begane grond (Unity Technologies, 2016f).

Verder kan men ook de positie van de *NavMesh* accurater bepalen door gebruik te maken van een *Height Mesh*, wat bijvoorbeeld van pas komt bij de positie an een agent op een trap (Unity Technologies, 2016b). Ook is het mogelijk om een *NavMesh* te generen over verschillende *Scenes* via *Additive Loading* met behulp van een *Off-Mesh Link* (Unity Technologies, 2016e). Het gebruik van een *NavMesh Agent* heeft ook voordelen om gebruik te maken met andere componenten van Unity 5 zoals *Physics* en

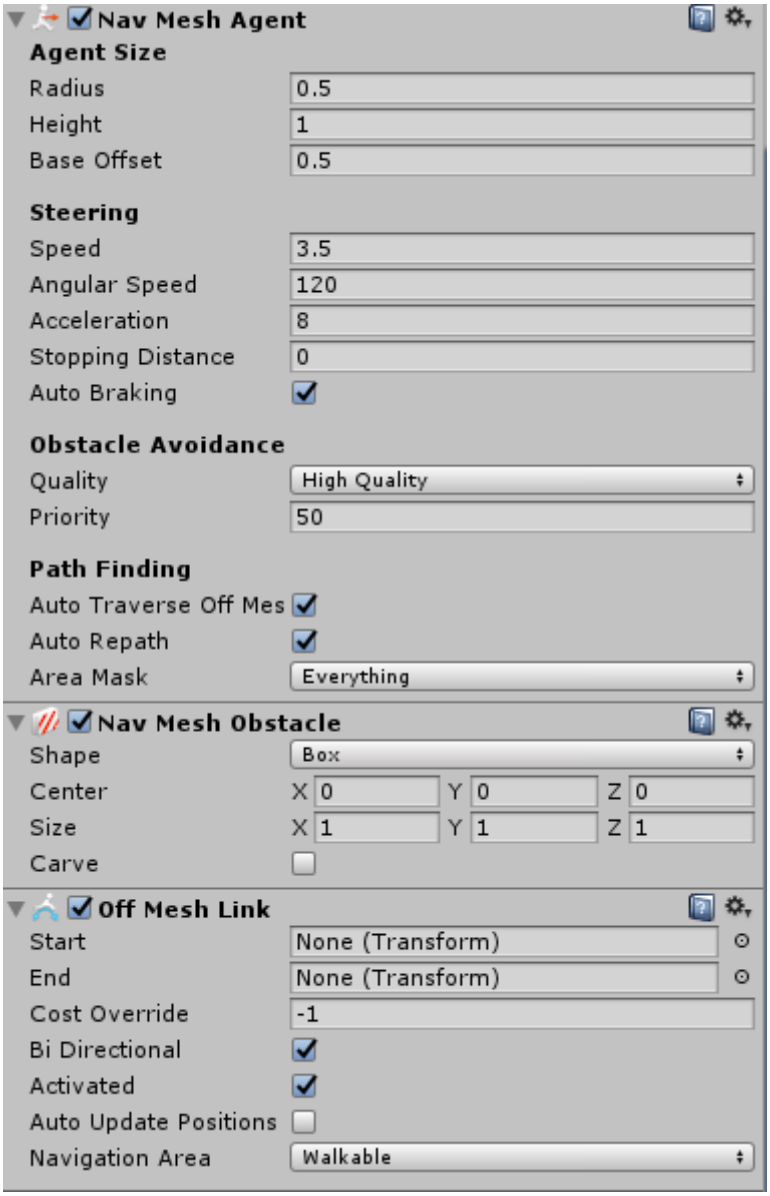
Animator (Unity Technologies, 2016l). Men kan de snelheid van de *NavMesh Agent* doorgeven aan de *Animator* om de animaties van het karakter er beter op af te stellen (Unity Technologies, 2016l).

AI interface

Via het *Navigation* venster kan men in Unity 5.4 artificiële intelligentie voor navigatie toevoegen aan de objecten (Unity Technologies, 2016g). Via een *Navigation Mesh*, te zien op figuur 4.3, kan men de oppervlakte beschrijven die men kan gebruiken. Het object wordt aangemaakt met behulp van een *Nav Mesh Agent* en men kan een *Off Mesh Link* gebruiken om de agent kortere wegen te laten gebruiken (Unity Technologies, 2016g). Er zijn ook nog *Nav Mesh Obstacles* die de agent zal ontwijken (Unity Technologies, 2016g). Deze drie componenten worden weergegeven in figuur 4.4.



Figuur 4.3: *Navigation Window* met 3 submenu's: *Object*, *Bake* en *Areas*

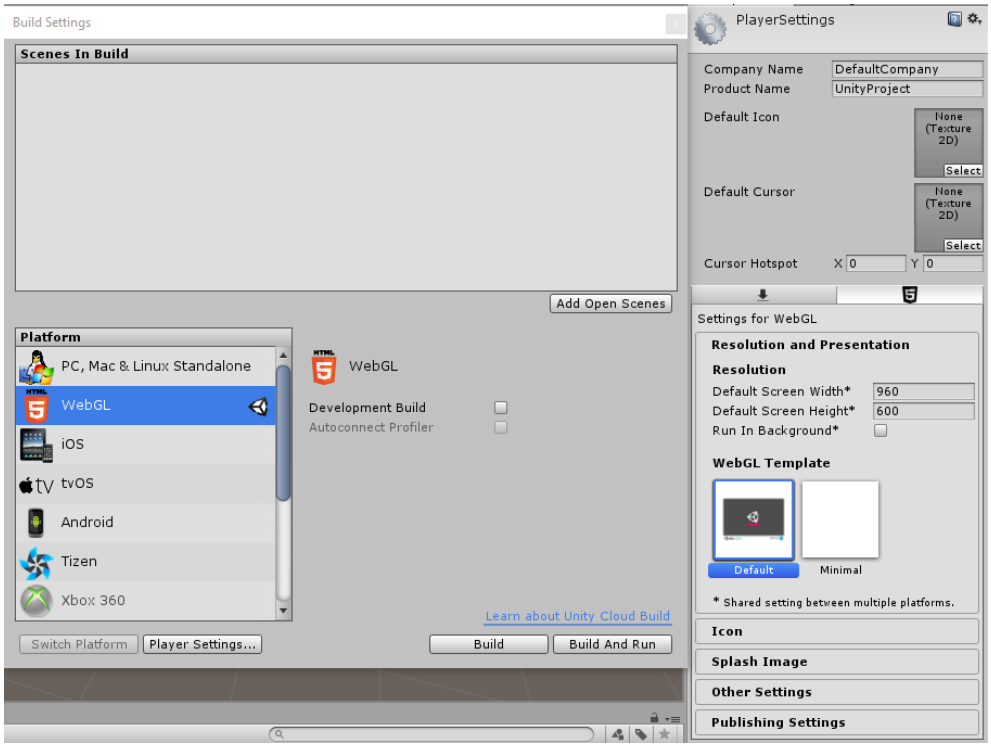


Figuur 4.4: De drie navigatiecomponenten: *Nav Mesh Agent*, *Nav Mesh Obstacle* en *Off Mesh Link*

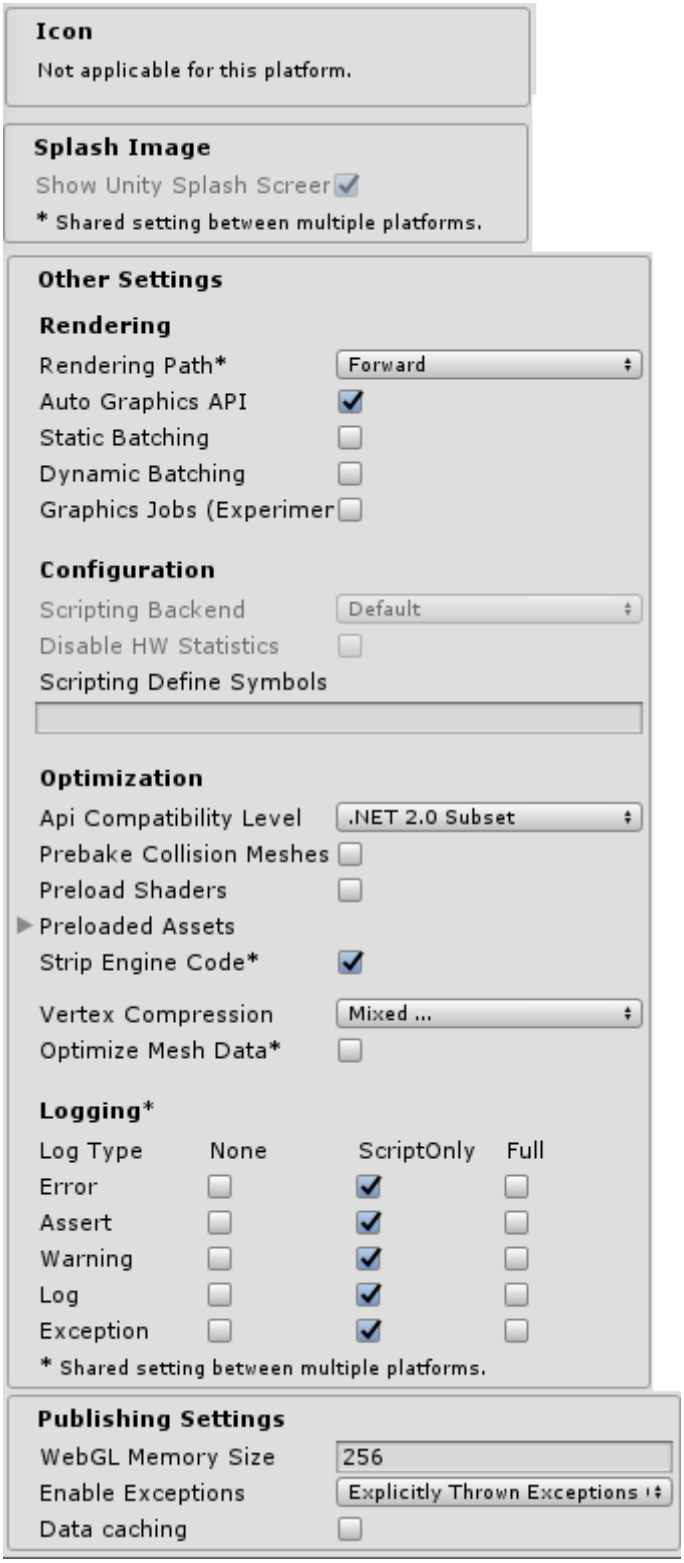
Van C# code naar HTML5 en WebGL

Om code over te zetten naar WebGL wordt er gebruik gemaakt van de emscripten compiler van Mozilla (Echterhoff, 2014). In de versie van Unity 5.4.0f3 is dit versie 1.34.1. Het compileert de geschreven code van C++ naar JS om dan een HTML5 template te gebruiken om de JS in te laden. Echter heeft Unity Technologies een interne tool ontwikkeld die de geschreven C# omzet naar de C++ code die emscripten gebruikt (Echterhoff, 2014). Deze tool wordt Intermediate Language to C++ (IL2CPP) genoemd (Peterson, 2015). Het bestaat uit twee verschillende delen waaronder een *runtime* bibliotheek die de virtuele machine van .NET ondersteunt (Peterson, 2015). Deze bibliotheek zorgt onder andere voor de implementatie van interne aanroepen zoals het uitvoeren van *garbage collection* en het gebruik maken van bronnen via een Application programming interfaces (API) (Peterson, 2015). Het andere onderdeel is de ahead-of-time (AOT) compiler (Peterson, 2015). Dit onderdeel vertaalt de low-level output, de *intermediate language* (Peterson, 2015). Deze objectgeoriënteerde taal wordt gebruikt door compilers van .NET compilers (Peterson, 2015). De AOT compiler vertaalt dus deze low-level output van .NET naar de C++ broncode, deze C++ code wordt dan door de emscripten compiler gebruikt (Peterson, 2015).

Om in Unity 5 dan een project te bouwen krijg je verschillende opties zoals te zien in figuur 4.5. We kiezen natuurlijk voor een HTML5 *build*. We hebben dan onderaan ook de *Player Settings* die dan rechts te zien zijn. Hierin kun je verschillende componenten definiëren waaronder de resolutie en presentatie door bijvoorbeeld de Unity WebGL template te gebruiken. Verder zijn er ook nog instellingen voor eventueel een icoon of *splash* foto. Er zijn ook meer geavanceerde instellingen voor onder andere rendering, publicatie en optimalisatie zoals te zien in figuur 4.6



Figuur 4.5: De *build* opties van Unity 5: deel 1



Figuur 4.6: De *build* opties van Unity 5: deel 2

Hoofdstuk 5

Unreal Engine

Unreal Enigne is een game development suite die ontwikkelaars, zowel van indie developers als grote ontwikkelaarsstudio's, helpt bij het maken van 2D en 3D omgevingen (Epic Games, 2016c). De suite, gemaakt door Epic Games, is bruikbaar op verschillende platformen zoals desktopcomputers, smartphones, consoles, VR en HTML5 (Epic Games, 2016c). Een van de eerste WebGL spellen die gebruik maakt van Unreal Engine 4 is Tappy Chicken (Noland, 2014). Het spel werd ontwikkeld door één persoon zonder gebruik te maken van C++ maar met behulp van *Blueprint Visual Scripting* (Noland, 2014). Ook de recente games, zoals Batman: Return to Arkham, maken gebruik van Unreal Engine 4 (Futter, 2016). Voor een toekomstige VR implementatie van Batman gaat de ontwikkelaarsstudio, Rocksteady, gebruik maken van Unreal Engine 4 (Jarvis, 2016).

5.1 Geschiedenis

Door populariteit van eerste-persoon schietspellen begon Epic Games, onder de naam Epic MegaGames (Plante, 2012), in 1998 aan hun eigen creatieve visie van dit genre onder de naam Unreal (Bleszinski, 2010). Het spel, Unreal, was beschikbaar vanaf 22 mei 1998 (Plante, 2012). In datzelfde jaar besloot Epic dan ook om de tools die ze maakte en gebruikte voor Unreal te verkopen via licenties aan bereidwillige partners, zoals Legend Entertainment en Microprose (Bleszinski, 2010). De focus voor de eerste Unreal Engine lag op desktopspellen en dan voornamelijk voor eerste eerste-persoon schietspellen (Plante, 2012). De engine ondersteunde naast C++ ook UnrealScript, een eigen scriptingtaal (Busby, Parrish & Wilson, 2009). Met de komst van Unreal Engine 2 werd dit uitgebreid en kwam ook support voor consoles zoals PlayStation 2, Xbox en Gamecube (Bleszinski, 2010). In 2001 kwam het Unreal Developer Network dat diende als hub voor alle game developers die gebruik maakte van Unreal Engine (Busby e.a., 2009). Unreal Engine 2 bracht ook nieuwe mogelijkheden zoals nieuwe

physics (Busby e.a., 2009), maar ook een deeltjessysteem en werd het systeem voor het renderen herwerkt (Bleszinski, 2010). In een later stadium hebben andere studio's de engine naar andere apparaten gebracht zoals smartphones en draagbare systemen (Parrish, 2011). Zo bracht Ubisoft een Unreal Engine 2 port naar de Nintendo 3DS (Parrish, 2011). In november 2016 kwam dan de derde grote versie van Unreal Engine uit op de markt (Busby e.a., 2009). De focus lag op hoge definitie (HD) systemen en de grafische kwaliteiten van Unreal Engine (Bleszinski, 2010). Dit bracht een hoop meer complexiteit met zich mee voor ontwikkelaars om de nieuwe tools te gebruiken (Bleszinski, 2010). Na verloop van tijd kwam Unreal Engine 3 dan toch in het goede daglicht te staan en kondigde Epic Games iPhone support aan voor de engine (Bleszinski, 2010) en later ook Android (Epic Games, 2013).

In 2005 verscheen het eerste nieuws over Unreal Engine 4 die al reeds enkele jaren in ontwikkeling was (Epic Games, 2005). Pas in 2012 werd de engine getoond aan ontwikkelaars achter gesloten deuren op de Game Developer Conference (Shaw, 2012). Support voor de engine werd uitgebreid naar verschillende apparaten (Epic Games, 2016c), echter was support voor UnrealScript verwijderd (Nutt, 2014). Een van de verbetering was een manier om via *Blueprint Visual Scripting* visueel te coderen zonder gebruik te maken van C++ (Thier, 2012). Het bracht ook nog andere features zoals *live debugging* (Thier, 2012). Op 2 maart 2015 werd aangekondigd dat er een gratis versie beschikbaar komt (Sweeney, 2015). Momenteel bevat Unreal Engine 4, sinds 4.7, ook support voor HTML5 (Grubb, 2015). Momenteel zit de engine aan versie 4.12 en bracht deze enkele nieuwigheden op VR gebied (James, 2016), maar is er nog geen uitgebreide informatie want de officiële documentatie geeft aan dat dit nog steeds in experimentele fase is (Epic Games, g.d.-e). Het is ook de eerste game engine die gebruik maakt van Vulkan API (Batchelor, 2016).

5.1.1 Populariteit

Epic Games en hun Unreal Engine wordt enorm hoog gewaardeerd in de industrie, zo stond deze engine bovenaan op de eerste plaats in 2015 op *The Tech List*. Als we echter kijken naar het marktaandeel bevindt Unreal Engine zich minder bovenaan. Zo gebruikte in 2014 slechts 12 procent Unreal Engine als video game ontwikkelaarssuite in het Verenigd Koninkrijk (Tnw Deals, 2016). Het blijft dus wel een van de meer populaire tool, maar Unity wordt blijkbaar wel meer gebruikt zoals te lezen in sectie 4.1.1. Sinds de aankondiging van een gratis versie van Unreal Engine 4 steeg het aantal gebruikers tot 1,5 miljoen waardoor het de snelste stijger is qua gebruikers in de categorie game engines (Batchelor, 2016). Ook hebben er zeker zeven gameseries elk al meer dan 1 biljoen dollar opgebracht waaronder Batman (Batchelor, 2016).

5.2 Unreal Engine 4

Unreal Engine 4 werd op 19 maart 2014 op de markt gebracht via een abonnement (Dyer, 2014) en heeft sindsdien al een paar significante updates gekregen tot versie 4.12 (Dyer, 2016). Deze versie verscheen op 1 juni 2016 (Dyer, 2016). Unreal Engine 4 ondersteunt C++ als programmeertaal, maar heeft ook tools voor visueel games te implementeren zonder code door gebruik te maken van *Blueprint Visual Scripting* (Epic Games, 2016a). In versie 4.7 kwam er support voor HTML5 (Grubb, 2015), maar is nog steeds in experimetele fase (Epic Games, g.d.-e).

5.2.1 Edities

Unreal Engine 4 is gratis voor niet commerciële doeleinden of voor educatief gebruik (Epic Games, 2016b). Als je de Unreal Development Kit (UDK) wilt gebruiken binnenin het bedrijf, is men verplicht een licentie te tekenen (Epic Games, 2016b). Voor intern gebruik bedraagt het bedrag 2500 dollars per ontwikkelaar per jaar (Epic Games, 2016b). Echter voor commerciële doeleinden betaalt men 99 dollars vanaf het begin waarbij men 0 procent van de auteursrechten afstaat bij de eerste 50.000 dollars (Epic Games, 2016b). Dit percentage stijgt wel naar 25 procent na de eerste 50.000 dollars (Epic Games, 2016b). Slechts één licentie is nodig voor het bedrijf om alle applicaties en/of spellen te ontwikkelen (Epic Games, 2016b).

5.2.2 Features

Op 24 februari 2015 kwam support uit voor HTML5 en WebGL voor Unreal Engine 4 (Sieprawski, 2015). Later werden ook andere platformen ondersteund zoals VR platformen (Cowley, 2015). Het meest recentste platform is Universal Windows Platform (UWP) (Jones, 2016). Ook één van de grote voordelen van Unreal Engine 4 is *Blueprint Visual Scripting* waarbij men een compleet spel kan spelen zonder maar één lijn code in C++ te schrijven (Epic Games, 2016a). Ook ondersteunt deze scriptingtaal *live debuggen* (Epic Games, 2016a). Verder werd de AI verbetert (Epic Games, 2016a). De karakters hebben nu beter inzicht in de ruimte waarin ze zich bevinden en kan men sneller en slimmer bewegen (Epic Games, 2016a). Niet alleen wordt de Unreal Engine gebruikt voor games, ook in films worden nieuwe features gebruikt zoals real-time cinematografie (Hagedoorn, 2016).

5.2.3 Systeemvereisten

Om de Unreal Engine 4 te gebruiken raadt men aan om een desktopcomputer te gebruiken met Windows 7 64-bit of hoger of een Mac OS met Mac OS X 10.9.2 of hoger, met minimum 8 Gigabyte (GB) RAM (Epic Games, g.d.-h). Qua processor

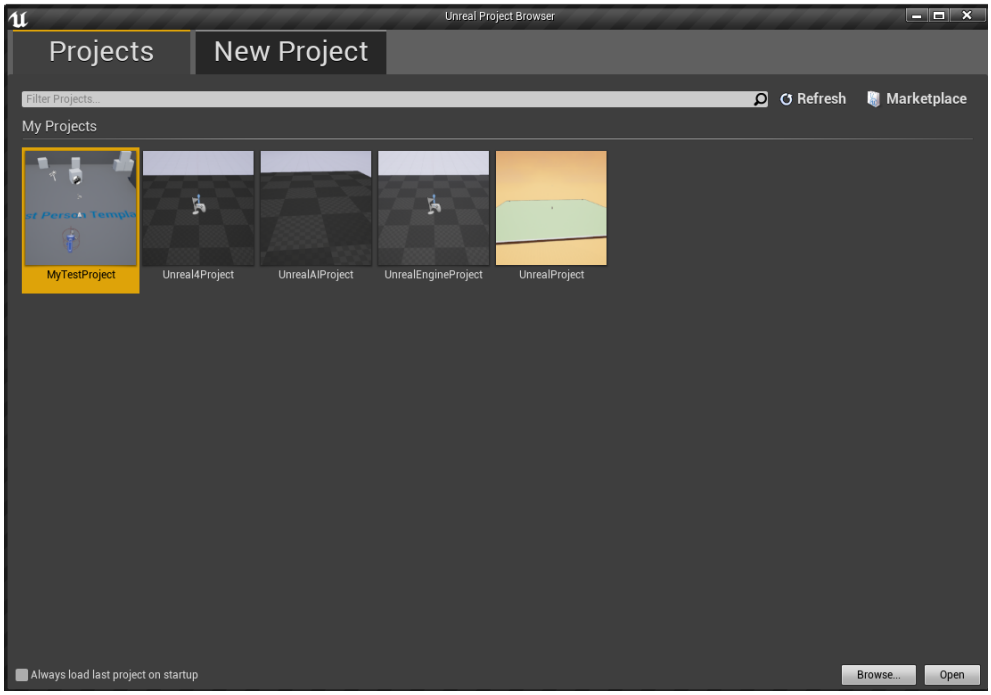
heeft men een quad-care Intel of Advanced Micro Devices (AMD) processor nodig met minstens 2.5 Gigahertz (GHz) (Epic Games, g.d.-h). Een grafische kaart moet voldoen aan een Nvidia GeForce 470 GTX kaart of AMD 6870 HD series kaart of hoger (Epic Games, g.d.-h). Voor het ontwikkelen van HTML5 projecten is het nodig om een 64-bits browser te gebruiken die HTML5 en WebGL ondersteunt zoals Chrome en Firefox (Epic Games, g.d.-e).

5.2.4 Omgeving

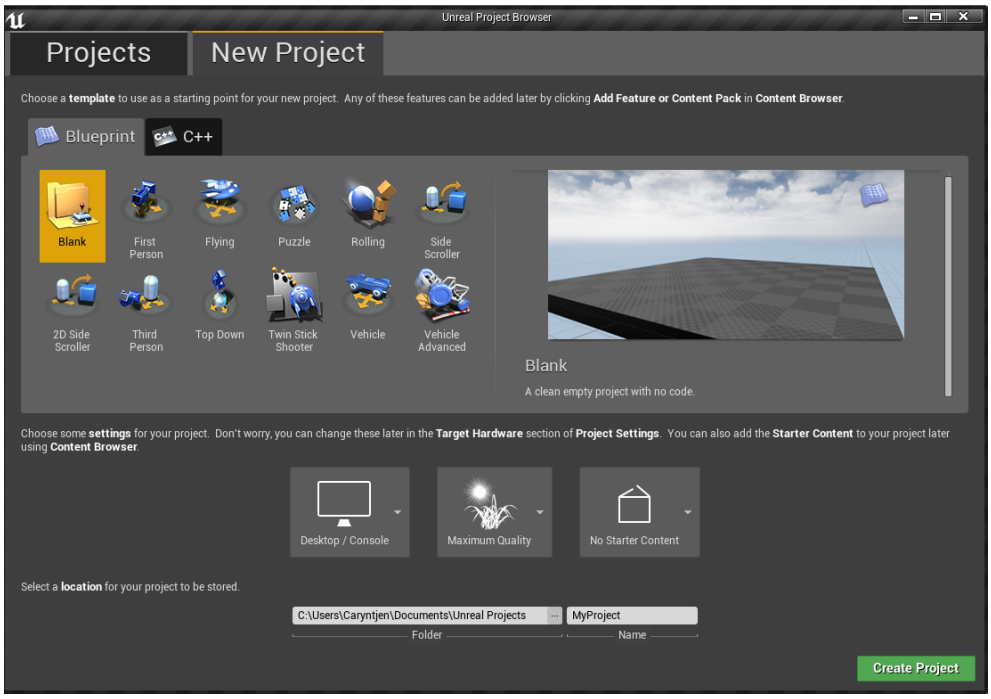
Voor dit prototype maken we gebruik van de gratis versie van Unreal Engine 4.12.5-3039270. We starten de applicatie op het Windows 10 besturingssysteem. We maken gebruik van de 64-bits editor van Unreal Engine 4 waarop we zijn ingelogd.

Starten

Voor we een project kunnen starten komen we terecht in de *Unreal Project Browser*, te bekijken in figuur 5.1, waarbij men een overzicht krijgt van huidige projecten. Men kan bovenaan kiezen om te filteren op projecten. Helemaal bovenaan kan men een nieuw project starten. Hierbij kan men kiezen voor een *Blueprint* project of een *cplusplus* project en bijhorend enkele instellingen aanpassen zoals hardware, kwaliteit en eventueel extra inhoud. Voor een nieuw project kan men de folder en de naam instellen zoals te zien op figuur 5.2. Rechtsboven kan men rechtstreeks gaan naar de *Marketplace* waar men artefacten kan aankopen om zich verder te helpen in een project. Links onderaan kan men kiezen om steeds het laatste project te laden, terwijl we rechts via *Browse...* een projectfolder kunnen toevoegen of openen. We kiezen hier voor een nieuw leeg *Blueprint* project op desktop met maximum kwaliteit en extra inhoud voor starters.



Figuur 5.1: Starten van *Unreal Project Browser* in Unreal Engine 4 op Windows 10

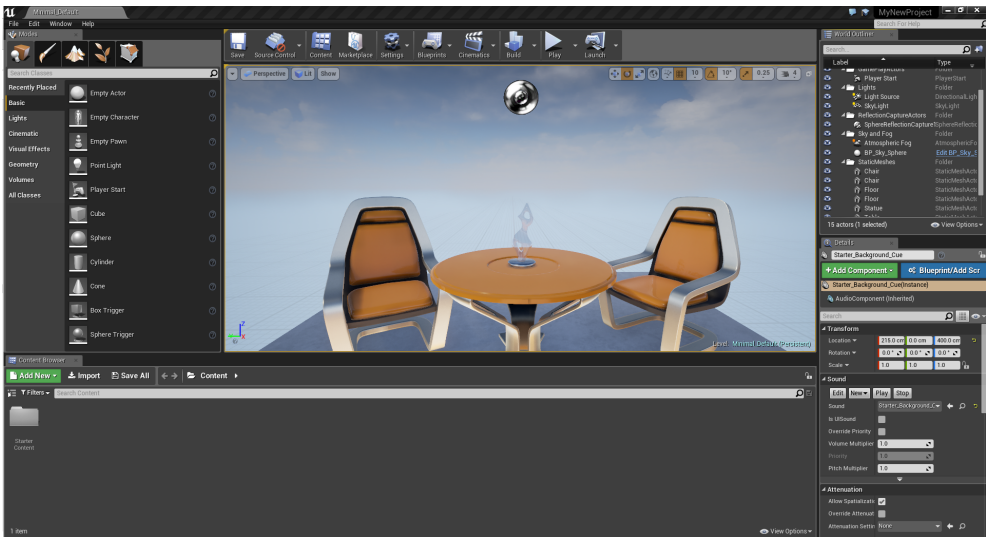


Figuur 5.2: Starten van nieuw project in Unreal Engine 4 op Windows 10

Editor

De editor wordt ingeladen en getoond zoals weergeven in figuur 5.3. Helemaal bovenaan rechts bevindt zich de naam van het project, met daarnaast opties om feedback te verzenden of voor instructies te tonen. Daaronder bevindt zich een *Main Menu Bar* met verschillende functies voor onder andere bestanden en vensters. Als we dan bovenaan centraal kijken bevinden we ons op de *Toolbar* waarbij er zich de meestgebruikte functies bevinden voor een *Scene* op te slaan of om de *Scene* te bouwen en te spelen. Helemaal rechts op de *Toolbar* kan de *Scene* ingeladen worden in bijvoorbeeld een browser zoals Chrome. De *Scene* is eronder zichtbaar in de *Viewport*. De *Viewport* bevat ook enkele opties om onder andere te navigeren in de *Scene* of om een anders perspectief te bekomen.

Links van de *Toolbar* bevindt zich het *Modes* paneel, waarin we beheren in welke modus de editor zich bevindt. Op figuur 5.3 bevinden we ons in de *Place* mode waarin we objecten kunnen plaatsen in de *Scene*, zoals basisobjecten, lichten, enzovoort. Er zijn dus nog andere modi zoals *Paint*, *Landscape*, *Foliage* en *Geometry Editing*. In de *Place* modus bevindt zich links een categorie zoals *Basic* en rechts ernaast de effectieve objecten zoals *Cube*. Deze objecten kunnen dan in de *Scene* geplaatst worden. Rechts bevinden we ons in het *World Outliner* venster waarin de objecten die zich momenteel in de *Scene* bevinden worden aangeduid met een label en een type. Wanneer we op een object in het *World Outliner* venster klikken, krijgen we, onderaan in het *Details* paneel, uitgebreide informatie over dat object alsook de mogelijkheid om opties toe te voegen en aan te passen. Helemaal onderaan bevindt zich dan ook een *Content Bowser* waarin men de folderstructuur kan doorzoeken. Hier kan men ook nieuwe inhoud, zoals een *Blueprint* of *cplusplus* klasse, aanmaken of aanpassen.



Figuur 5.3: Editor van Unreal Engine 4 op Windows 10

AI hulpmiddelen

De Unreal Engine heeft een uitgebreidere manier om AI te implementeren dan Unity 5. De nadruk ligt niet alleen op navigatie en *path finding* zoals bij Unity 5 te lezen in sectie 4.2.4. *Behavior Trees* is één van de tools die Unreal Engine 4 voorziet om AI te implementeren (Epic Games, g.d.-a). Het is een combinatie van twee verschillende soorten types (Epic Games, g.d.-a). Zo wordt het *Blackboard* gebruikt om het geheugen van de AI te definiëren (Epic Games, g.d.-a). Dit type slaat de belangrijke waarden op die de *Behavior Trees* gebruiken (Epic Games, g.d.-a). Dan hebben we nog de *Behavior Tree* zelf die de processor is van de AI (Epic Games, g.d.-a). De *Behavior Tree* maakt dus beslissingen en reageert daarop (Epic Games, g.d.-a).

De *Behavior Tree* kan je zien als een datastructuur in de vorm van een algoritmische boom (Simpson, 2014). De boom heeft een wortel die zich bovenaan bevindt waarmee knooppunten hiërarchisch verbonden zijn (Simpson, 2014). De knooppunten beheren dan steeds de manier waarop beslissingen gevolgd worden (Simpson, 2014). Als men dan bij één van de bladeren of eindpunten van de boom komt, bevat deze eigenlijk de effectieve beslissingen die de AI dan uitvoert (Simpson, 2014). Om te bepalen hoe een AI best reageert, zal men de boom doorlopen en zo de beste mogelijke oplossing zoeken (Simpson, 2014). Unreal Engine 4 implementeert deze bomen zodat ze reageren op events (Epic Games, g.d.-f). Deze events kunnen zorgen voor veranderingen in de boom (Epic Games, g.d.-f). Hierdoor luistert de *Behavior Tree* passief naar events (Epic Games, g.d.-f).

De bomen definiëren enkel wat een AI kan doen, maar niet hoe of waarom (Epic Games, g.d.-g). Deze bomen gebruikt men dan ook in combinatie van een *navigation mesh* (*Nav Mesh*) waarbij men de beloopbare ruimte definieert (Epic Games, g.d.-g). Men kan deze ruimte van de *Nav Mesh* aanpassen via een *Nav Modifier Volume* (Epic Games, g.d.-g). Moeilijkere plaatsen zonder pad zijn te bereiken via een *Nav Link Proxy* (Epic Games, g.d.-g). Met behulp van *Blueprints* implementeert men hoe en waarom een AI zich moet gedragen (Epic Games, g.d.-b).

Naast de *Behavior Tree* is er ook een ander systeem, namelijk het *Environment Query System (EQS)* (Epic Games, g.d.-d). Dit systeem is een manier om data te halen uit de omgeving. Dit gebeurt tests om de beste mogelijke data te bekomen aan de hand van vragen die het systeem stelt (Epic Games, g.d.-d). Zo kan men bijvoorbeeld de vijand met de hoogste dreiging bepalen of bepalen of deze schuilt beter schuilt achter een object (Epic Games, g.d.-d). Men kan dit systeem ook gebruiken in combinatie met een *Behavior Tree* (Epic Games, g.d.-d).

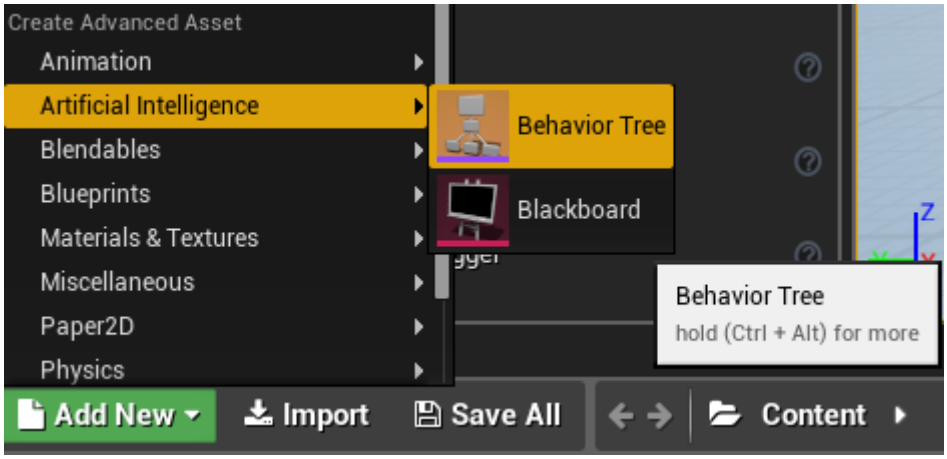
AI interface

Twee grote features komen naar boven wanneer we AI zouden willen implementeren. De eerste is *Behavior Trees*, die meestal ook een *Blackboard* bezit, zoals te zien op

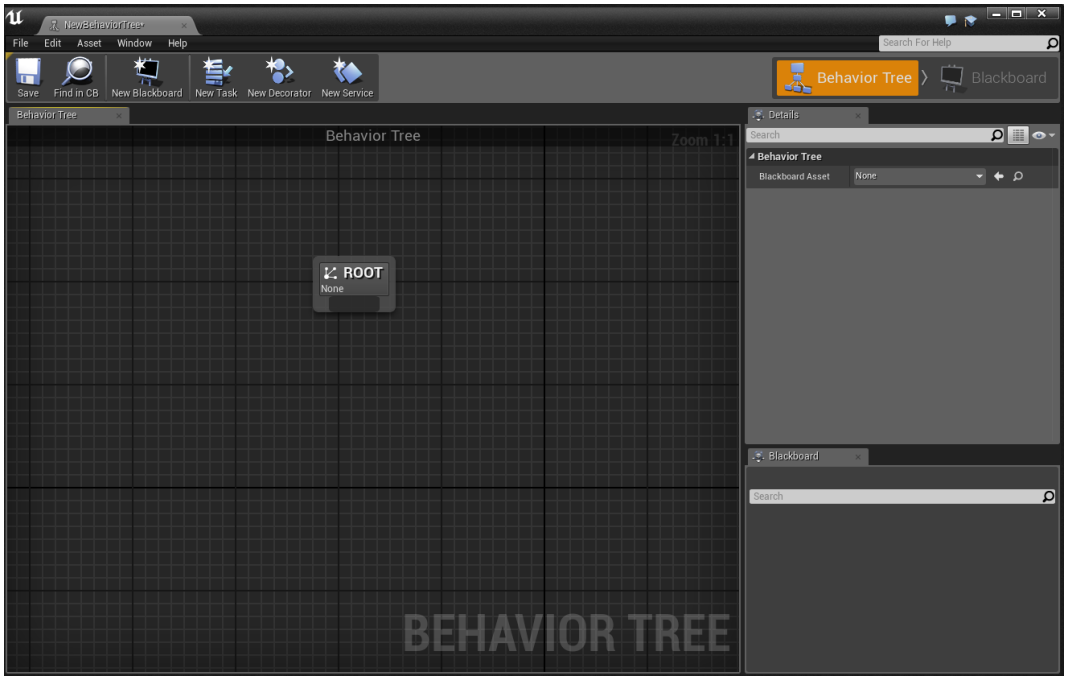
figuur 5.4. Deze bomen definiëren wat een AI allemaal kan doen. Met behulp van een *Blackboard* kan data geschreven en gelezen worden om beslissingen van dit gedrag te definiëren. We tonen respectievelijk de vensters van *Behavior Trees* en *Blackboard* in figuren 5.5 en 5.6.

Om te navigeren kan men gebruik maken van verschillende objecten. Zo is er bijvoorbeeld, te bekijken op figuur 5.6, dat men een *Nav Mesh Bounds Volume* kan definiëren als een volume waarin een AI kan navigeren en dit bewerken door middel van een *Nav Modifier Volume*. Om bepaalde moeilijke plaatsten te bereiken, zoals bijvoorbeeld een plaats waarop men kan springen, kan men gebruik maken van een *Nav Link Proxy*.

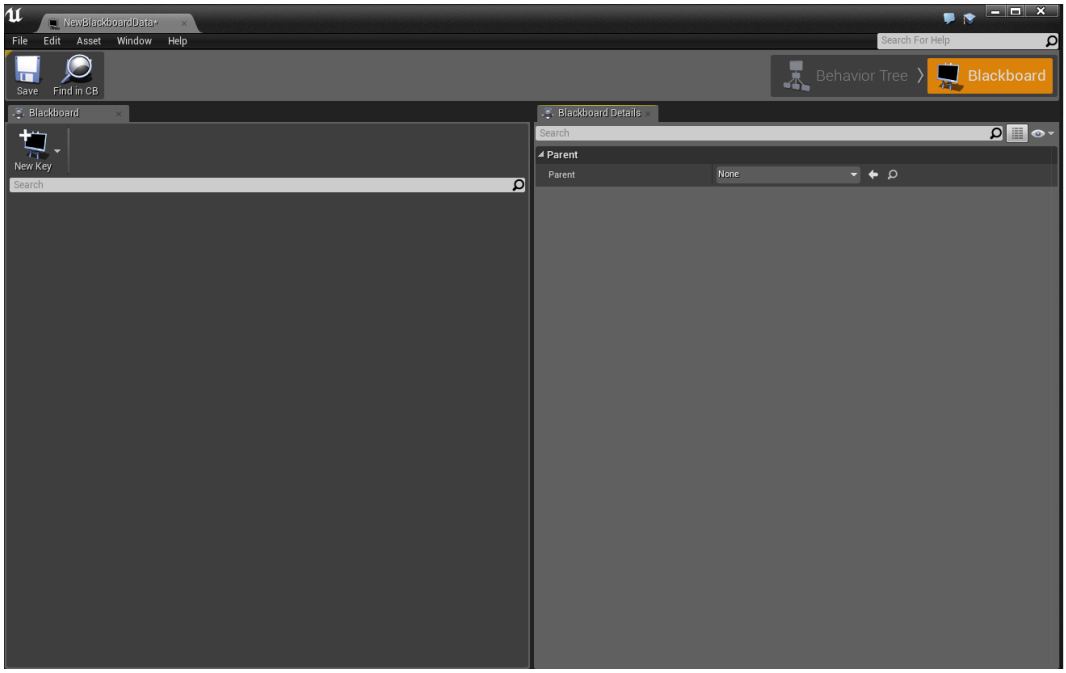
Een ander grote feature van Unreal Engine 4 is het gebruik van een *EQS*. Het is een manier om data uit de omgeving te halen en hierover vragen te stellen om dan een oplossing te bekomen. Dit wordt via een experimentele feature in de instellingen van de editor geactiveerd waardoor er een nieuwe optie verschijnt bij AI, zoals te zien op figuur 5.8. Het *EQS* venster is hieronder te bekijken op figuur 5.9.



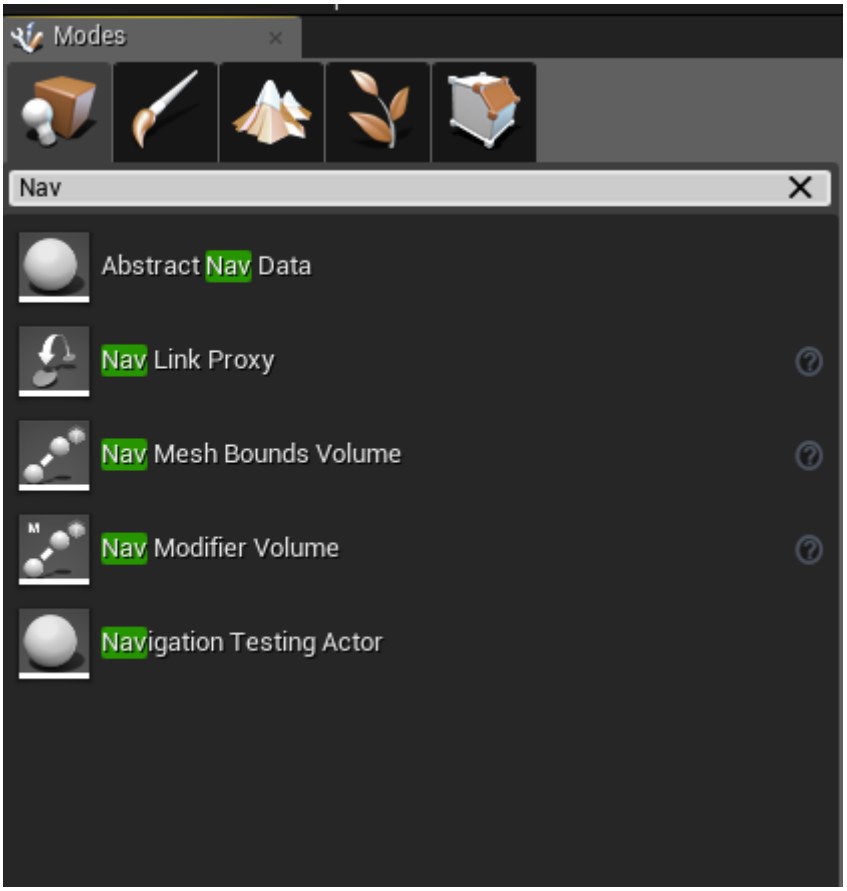
Figuur 5.4: *Content Browser* paneel met 2 opties om AI toe te voegen: *Behavior Tree* en *Blackboard*



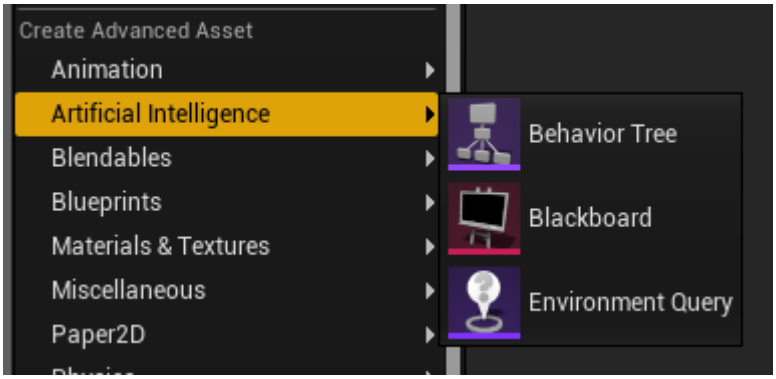
Figuur 5.5: *Behavior Tree* venster



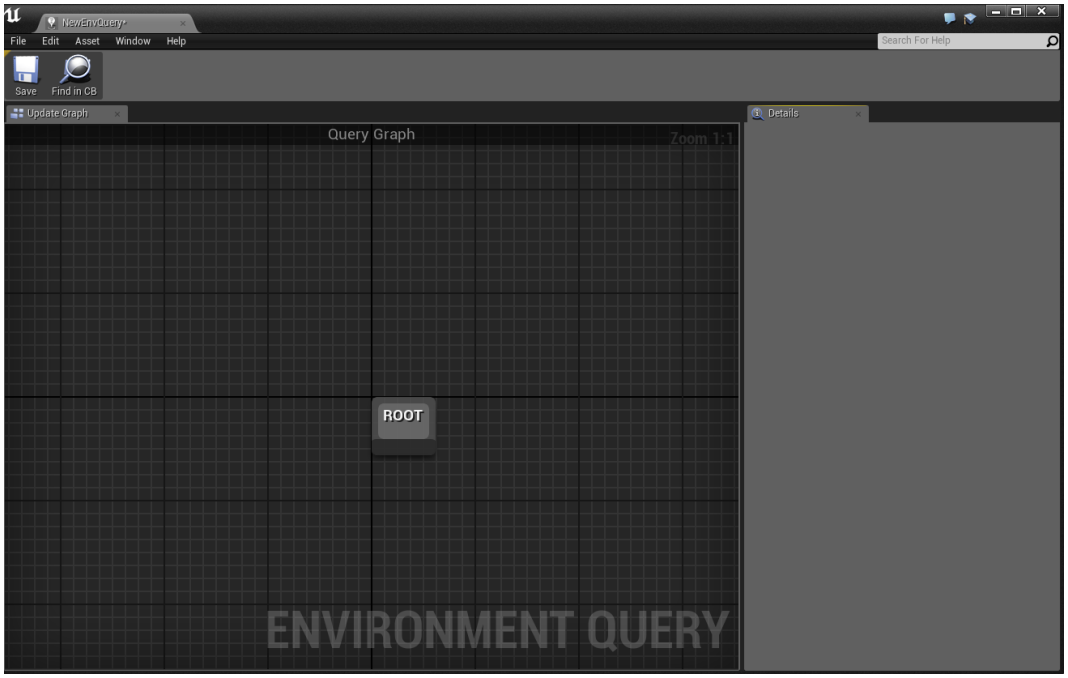
Figuur 5.6: *Blackboard* venster



Figuur 5.7: Objecten die *Nav* bevatten in het *Modes* paneel



Figuur 5.8: *Content Browser* paneel met 3 opties om AI toe te voegen: *Behavior Tree*, *Blackboard* en *Environment Query*

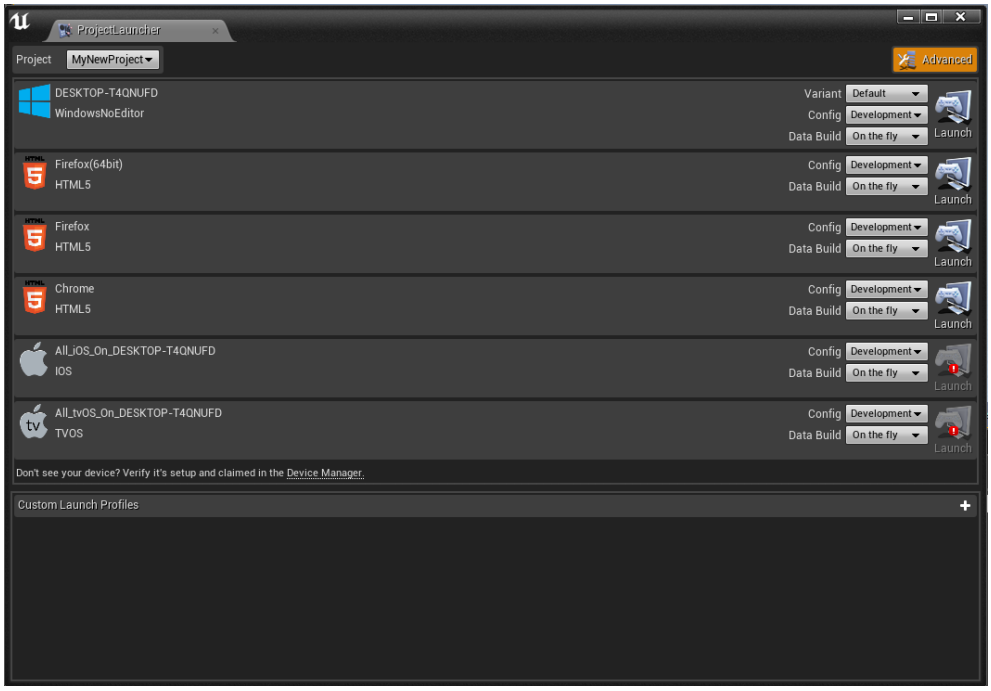


Figuur 5.9: Het *EQS* venster

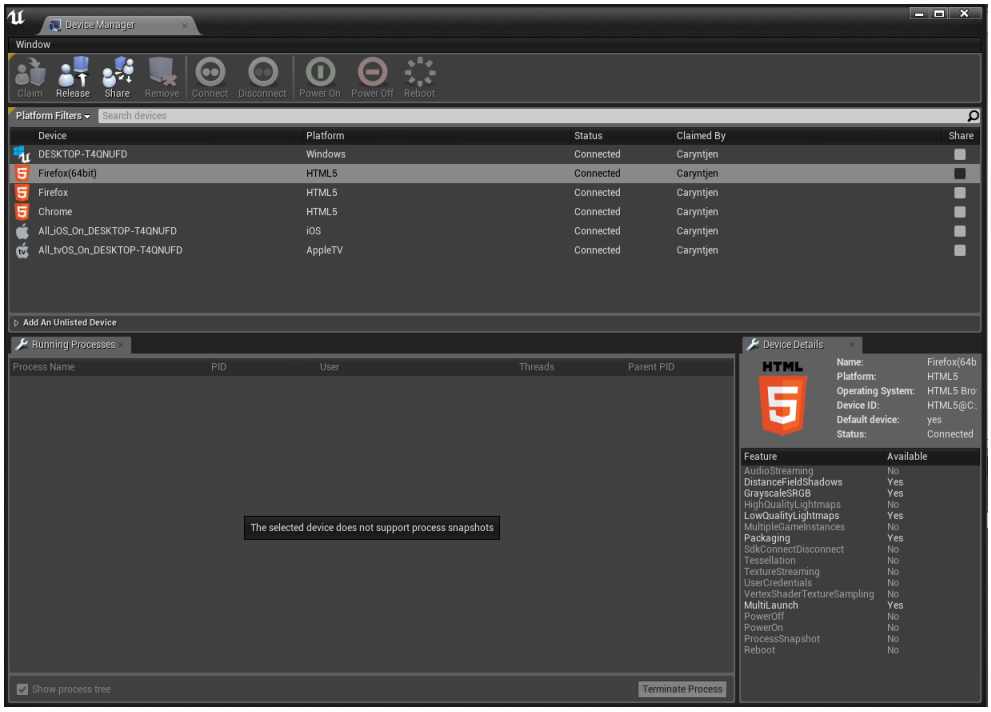
Van C++ code naar HTML5 en WebGL

Net zoals bij Unity in sectie 4.2.4 gebruikt Unreal Engine 4 de emscripten compiler van Mozilla (Epic Games, g.d.-e). Als we kijken in de folderstructuur van een project, zien we dat dit een hogere versie is voor Unreal Engine 4.12.5-3039270 in vergelijking met Unity 5. In Unreal Engine 4 gebruikt men versie 1.35.0 van emscripten, deze versie bevat verschillende nieuwigheden alsook *bugfixes* in vergelijking met versie 1.34.1 die in Unity 5 wordt gebruikt (Emscripten Contributors, 2016). De meest recentste versie is 1.36.5. Deze versie werd vrijgegeven op 7 juli 2016 (Emscripten Contributors, 2016).

Om een *build* te doen bij Unreal Engine 4 gaan wie hiervoor naar de *Project Launcher*, getoond in figuur 5.10. Deze toont de verschillende mogelijkheden. Via de geavanceerde knop krijgen we enkele extra opties om de *build* te personaliseren voor bijvoorbeeld een ontwikkelomgeving. Via de *Device Manager* kan men verschillende systemen aanmaken of toevoegen. De *Device Manager* toont ook de details van een systeem zoals te zien op figuur 5.11



Figuur 5.10: De Unreal Engine 4 *Project Launcher*



Figuur 5.11: De Unreal Engine 5 *Device Manager*

Hoofdstuk 6

Prototype

Om een duidelijker beeld te creëren wordt op beide engines een prototype ontwikkeld die gebruik zal maken van basis AI. Er worden meerdere scenario's ontwikkeld op zowel Unity 5 als Unreal Engine 4, gebruik maken van een desktopcomputer zoals beschreven in sectie 6.1. Met deze desktopcomputer zullen ook metingen worden uitgevoerd zoals geformuleerd in onder andere sectie 6.2.1.

6.1 De desktop

Om het scenario, beschreven in sectie 6.2, te programmeren en om de applicatie en metingen, getoond in sectie 6.2.1, uit te voeren zal er gebruikt worden van een desktopcomputer. Deze heeft volgende eigenschappen:

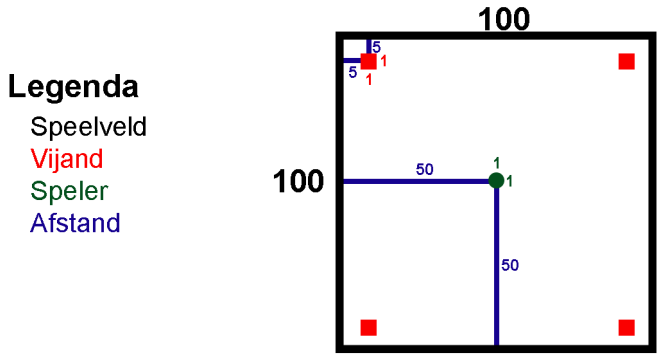
- 64-bits besturingssysteem: Windows 10.0.10586 Build 10586 Education
- Processor: Intel Core i5-2500K met 3,30 GHz
- RAM: 12GB
- GPU: 2x AMD Radeon HD 6900 Series met elkaar verbonden via CrossFireX technologie van AMD
- Browser: Chrome 52.0.2743.82 m (64-bit) en Firefox 47.0.1 (64-bit)
- Programmeeromgeving: Visual Studio Community 2015 versie 14.025424.00 Update 3
- Game engines: Unreal Engine 4.12.5-3039270 en Unity 5.4.0f3

6.2 Het eerste scenario

Hieronder wordt het scenario van het prototype omschreven waarbij we vooral de performantie meten van de editor en de invloeden van verschillende browsers. In figuur 6.1 wordt het spelbord gevisualiseerd van boven af om een visueel beeld van de opstelling van het project te verkrijgen. Dit scenario omvat volgende eigenschappen:

- Het project
  - Er wordt een nieuw leeg project opgestart.
  - Er wordt zoveel mogelijk gebruik gemaakt van basis objecten.
  - Er worden minstens minstens twee klassen zelf gecodeerd in C++ of C#.
  - Er worden geen extra texturen toegevoegd of artefacten gebruikt van de beschikbare winkels.
- De speler
  - De speler is een eerste-persoon speler.
  - De speler kan links en rechts bewegen met de pijltjestoetsen met een snelheid van 10 eenheden.
  - De speler kan rond zichzelf kijken door middel van de muis te bewegen.
  - De speler bevindt zich in het midden van het speelbord.
  - De speler heeft een hoogte van 2 eenheden, een breedte 1 eenheid en een straal van 0.5 eenheden.
- De vijanden
  - De vijand ziet de speler direct en beweegt in richting van de speler met snelheid van 10 eenheden. De vijand kleur hier momenteel blauw.
  - Indien een vijand zich op een afstand van 2 eenheden bevindt, kleurt die rood en wordt de snelheid verlaagt tot 8 eenheden.
  - Wanneer de vijand zich bij de speler bevindt kleurt die groen en stopt deze met bewegen.
  - De vijand is een kubus en bevat de standaard *physics* met een zijde van elk 1 eenheid.
  - Er zijn 4 vijanden op het speelbord aanwezig, elk aanwezig in één van de vier hoeken van het speelbord. De indeling wordt getoond op figuur 6.1.
- De camera
  - De camera bevindt zich op de locatie van de speler.

- De verlichting
  - Er wordt gebruik gemaakt van directe verlichting in combinatie met de standaard openlucht verlichting.
  - De directionele verlichting bevindt zich in het midden op een hoogte van 20 eenheden, onder een verticale hoek van -30 graden en een horizontale hoek van 50 graden met een sterkte van 1 eenheid.
- Het spelbord
  - Het spelbord is 100 eenheden lang en breed en heeft een hoogte van 2 eenheden.
  - Het spelbord maakt gebruik van een balk.
- De frames
  - De frames van het spel worden gemaximeerd tot 60 frames per seconde (FPS).



Figuur 6.1: Het bovenperspectief van het speelbord van het eerste scenario

6.2.1 De metingen

Er worden verschillende metingen gedaan met het scenario, te lezen in sectie 6.2, om de verschillen in omgevingen en hun performantie in browsers toe te lichten. Hiervoor wordt gebruik gemaakt van de desktopcomputer beschreven in sectie 6.1. Indien er een meting afhankelijk is van tijd, zal deze vijf keer uitgevoerd worden om een gemiddelde te berekenen. Geschiedenis van de browser en de cache wordt steeds leeg gemaakt bij elke meting, tenzij anders vermeld. Project cache wordt steeds leeggemaakt, tenzij anders vermeld. Bij elke meting worden dezelfde omstandigheden van de desktop-computer nagebootst. De browsers Chrome en Firefox werden geselecteerd op basis

van marktaandeel en ondersteuning door de game engine. Deze twee browsers zijn de meest gebruikte browsers wereldwijd in juni 2016 die HTML5 en WebGL ondersteunen (Calunod, 2016). Voor elke game engine worden volgende metingen uitgevoerd met dit scenario:

- De grootte van het project met het afgewerkte scenario.
- Het laden van het afgewerkte scenario in de game engine.
- Het bouwen en laden in de game engine van de productie *build* voor een WebGL omgeving van het afgewerkte scenario. Er worden meting in Firefox uitgevoerd voor de eerste *build*, zodat er geen cache aanwezig is. Er worden ook meting gedaan zonder de cache leeg te maken.
- De grootte van de productie *build* voor een WebGL omgeving van het afgewerkte scenario.
- Het laden van de productie *build* voor een WebGL omgeving van het afgewerkte scenario in twee verschillende browsers, Chrome en Firefox. De metingen worden zowel met browser caching als zonder browser caching uitgevoerd.

6.2.2 De resultaten

Unity 5

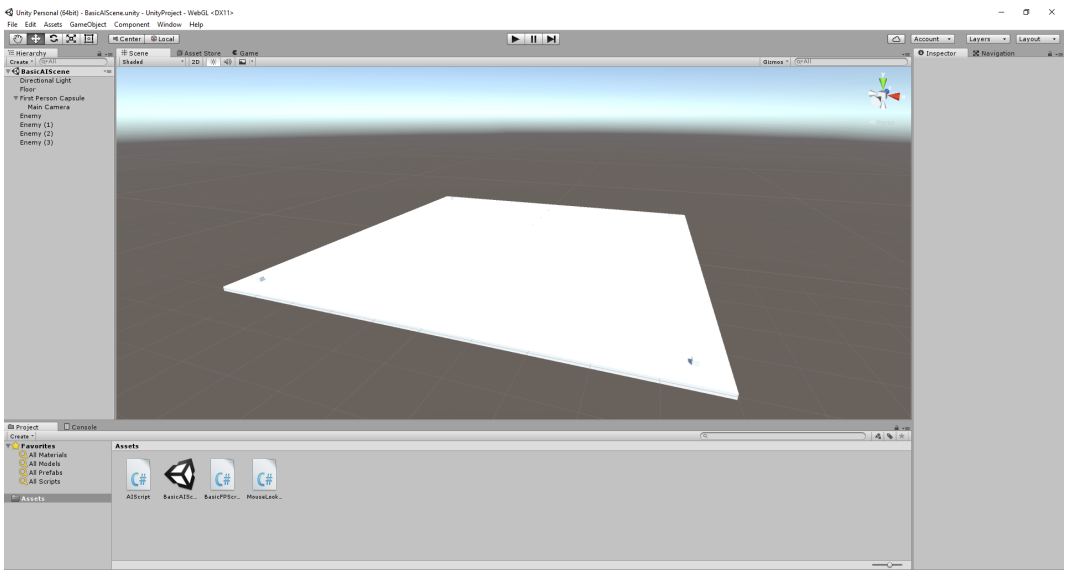
Voor Unity 5 worden er verschillende objecten aangemaakt voor de *Scene*, zoals beschreven in het scenario van sectie 6.2. Zo bestaat de speler uit een 3D object, een *Capsule*, waaraan een *Main Camera* wordt toegevoegd om het eerste persoon perspectief te genereren. De speler krijgt ook een *Rigidbody* component toegevoegd die automatisch de *physics* op de speler zal toepassen. De *rendering* van de *Capsule* wordt verwijderd. Aan de speler wordt ook nog een C# klasse toegevoegd om de AI van de speler te beschrijven, te zien in bijlage B.1. Hierbij erft de klasse af van *MonoBehaviour* en worden de snelheden gedefinieerd. *MonoBehaviour* is een klasse waarvan elk script moet overerven. Het object zal met dat script ook zoeken naar de speler en zo nodig zijn positie en snelheid wijzigen naarmate de afstand wijzigt. De camera krijgt ook een C# script, te bekijken in bijlage B.2, om de bewegingen van de muis te registreren. Deze klasse erft dan ook af van *MonoBehaviour*.

De vijanden bestaan ook uit 3D objecten, genaamd *Cube*, in de vorm van kubussen. De vijanden krijgen ook een *Rigidbody* voor de *physics* te berekenen. Door een *Rigidbody* te gebruiken wordt het object geactiveerd om gebruik te maken van de *physics* engine. Hierdoor zal het object dus reageren op de zwaartekracht. De vijanden krijgen ook elk een *Nav Mesh Agent* component die het mogelijk maakt om zelf het beste pad te zoeken, met als doeleind de speler. Ook hier krijgt men een C# klasse die erft af

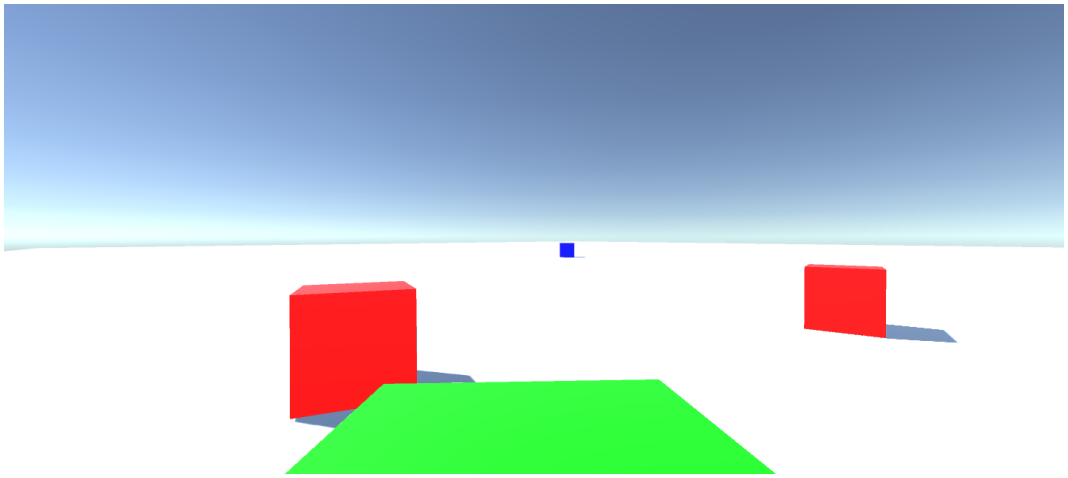
van *MonoBehaviour* van toegekend om de AI in te stellen, zoals te zien in bijlage B.3. In dit script wordt via code de kleur van de vijanden en de snelheid verandert volgens het scenario. De *Target* van dat script werd via de *Inspector* ingesteld op de speler.

De vloer is ook een 3D object van het type *Cube* waarbij er statische navigatie wordt toegevoegd waarop de vijanden dus kunnen lopen. Verder werd ook directionele verlichting toegevoegd. De openlucht verlichting is standaard toegevoegd, te zien in het *Lighting Window*. De *Nav Mesh* van het scenario wordt creëert door op *Bake* te klikken in het *Navigation Window*.

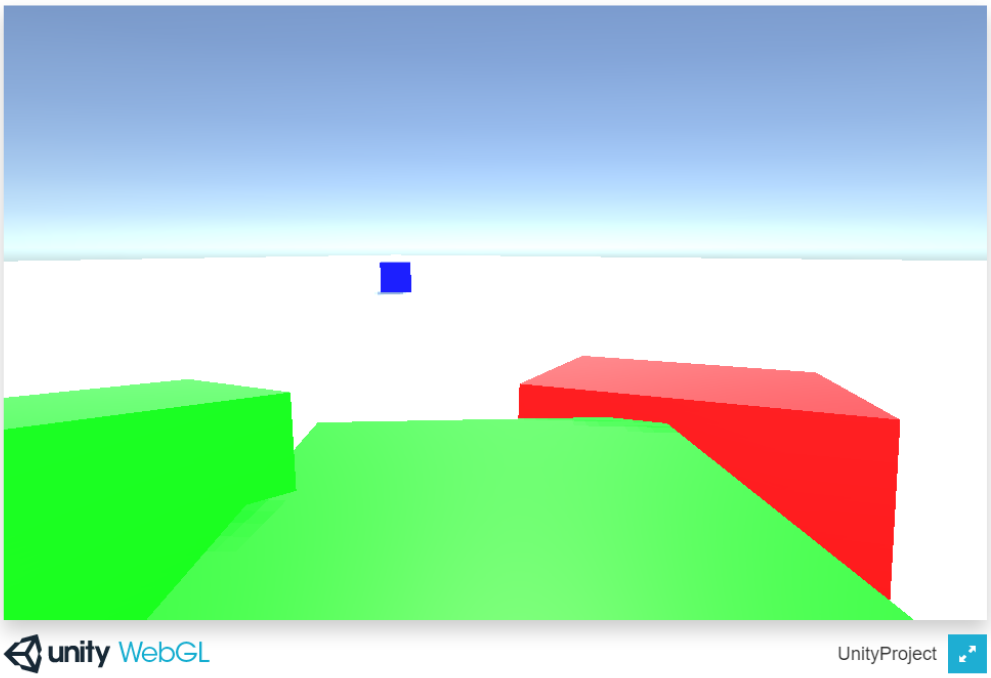
Het volledige resultaat van de editor is te zien in figuur 6.2. Ook is er een schermafbeelding te zien van het spel op figuur 6.3. Bij het laden van de *build* in een browser is de resolutie van het spel steeds hetzelfde als de standaard resolutie, namelijk 960 pixels breedte bij 600 pixels hoogte. Wanneer het spel is geladen in de browser verschijnt er onderaan een knop om het spel te vergroten naar de volledige schermgrootte. Figuur 6.4 toont een productie *build* geopend in de Chrome.



Figuur 6.2: Editor zicht van het afgewerkte scenario in Unity



Figuur 6.3: Schermafbeelding van het prototype in Unity



Figuur 6.4: Prototype geladen in Chrome browser in Unity

Meting 1: Laden van het project: Bij het laden van het project wordt een chronometer gestart wanneer het project wordt geopend bij de *launcher* van Unity. De tijd stopt wanneer de scène volledig is geladen. Gemiddeld komt dit neer op 6,06 seconden zoals te zien in tabel 6.1.

Meting	**Tijd in seconden**
Meting 1	6,21
Meting 2	6,03
Meting 3	5,58
Meting 4	6,34
Meting 5	6,14
Gemiddelde	**6,06**

Tabel 6.1: Unity Resultaten: Laden van het project

Meting 2: Grootte van het project: Het grootte van het project wordt achterhaald door het Windows eigenschappen venster van de folder. Het resultaat, getoond in tabel 6.2, is 44,1 Megabyte (MB).

Grootte in MB	**44,1**
Grootte in bytes	**46285725**

Tabel 6.2: Unity Resultaten: Grootte van het project

Meting 3: Bouwen en laden productie *build* in Firefox met cache: In tabel 6.3 is te zien dat het gemiddeld 2 minuten en 15,43 seconden duurt om een productie *build* te bouwen en te laden op , de 64-bits versie, zonder de geschiedenis te verwijderen.

Meting	**Tijd**
Meting 1	2m 15,39s
Meting 2	2m 13,48s
Meting 3	2m 17,01s
Meting 4	2m 15,23s
Meting 5	2m 16,04s
Gemiddelde	**2m 15,43s**

Tabel 6.3: Unity Resultaten: Bouwen en laden productie *build* in Firefox met cache

Meting 4: Bouwen en laden productie *build* in Firefox zonder cache: Bij het bouwen en laden van de productie *build* wordt het project gebouwd op de 64-bits versie van Firefox. Hierbij wordt telkens de recente geschiedenis van de browser verwijdert. Er worden vijf keer meting uitgevoerd. De meting start wanneer de *build* begint en tot het de scène van het scenario heeft ingeladen. Het gehele proces duurt gemiddeld, berekend in tabel 6.4, ongeveer 2 minuten en 40 seconden.

Meting	**Tijd**
Meting 1	2m 39,01s
Meting 2	2m 42,23s
Meting 3	2m 37,57s
Meting 4	2m 38,10s
Meting 5	2m 43,16s
Gemiddelde	**2m 40,014s**

Tabel 6.4: Unity Resultaten: Bouwen en laden productie *build* in Firefox zonder cache

Meting 5: Grootte van de productie *build*: Het generen van de productie *build* in Unity moet plaatsvinden op een andere locatie dan de project folder. Deze *build* is amper 4,70MB zoals te zien in tabel 6.5.

Grootte in MB	**4,70**
Grootte in bytes	**4935680**

Tabel 6.5: Unity Resultaten: Grootte van de productie *build*

Meting 6: Laden productie *build* in Firefox zonder cache: Het duurt slechts 9,104 seconden, volgens het gemiddelde van vij metingen in tabel 6.6, om de scène van Unity te laden in Firefox. Hierbij verwijdert men steeds de browsergeschiedenis zodat er geen cache meer aanwezig is.

Meting	**Tijd in seconden**
Meting 1	9,25
Meting 2	8,93
Meting 3	9,34
Meting 4	9,03
Meting 5	8,97
Gemiddelde	**9,104**

Tabel 6.6: Unity Resultaten: Laden productie *build* in Firefox zonder cache

Meting 7: Laden productie *build* in Firefox met cache: Hier gaat het natuurlijk een deel sneller om de scène van het scenario te laden in de 64-bits versie van Firefox in vergelijking met de meting in sectie 6.2.2. Na vijf metingen, te bezichtigen in tabel 6.7, bekomen we een gemiddelde van 6,21 seconden.

Meting	**Tijd in seconden**
Meting 1	6,87
Meting 2	5,96
Meting 3	6,13
Meting 4	6,21
Meting 5	5,88
Gemiddelde	**6,21**

Tabel 6.7: Unity Resultaten: Laden productie *build* in Firefox met cache

Meting 8: Laden productie *build* in Chrome zonder cache: Voor de 64-bits versie van Chrome worden er ook vijf metingen uitgevoerd om een productie *build* van de scène te laden. Tussen elke meting wordt de geschiedenis gewist zodat er geen cache aanwezig is. Het gemiddelde bedraagt 10,866 seconden, te zien in tabel 6.8.

Meting	Tijd in seconden
Meting 1	10,54
Meting 2	10,89
Meting 3	11,02
Meting 4	10,75
Meting 5	11,13
Gemiddelde	**10,866**

Tabel 6.8: Unity Resultaten: Laden productie *build* in Chrome zonder cache

Meting 9: Laden productie *build* in Chrome met cache: Als we het gemiddelde vergelijken van het laden van de *build* met of zonder cache, te zien in sectie 6.2.2, dan is er weinig verschil aangezien tabel 6.9 een gemiddelde van 9,24 seconden toont.

Meting	Tijd in seconden
Meting 1	9,46
Meting 2	8,95
Meting 3	9,36
Meting 4	9,56
Meting 5	8,87
Gemiddelde	**9,24**

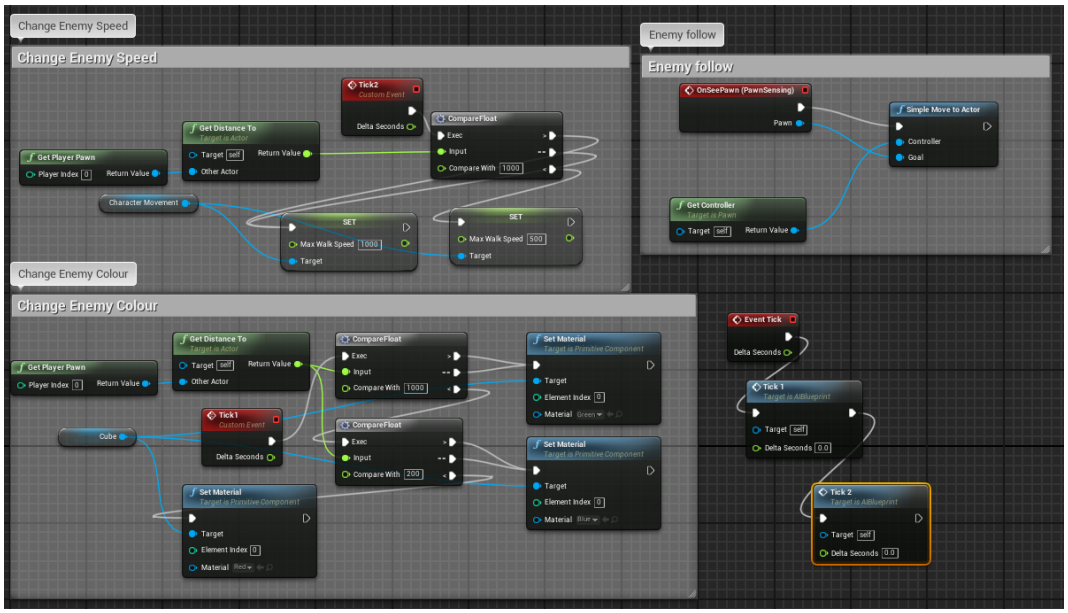
Tabel 6.9: Unity Resultaten: Laden productie *build* in Chrome met cache

Unreal Engine 4

Met behulp van Unreal Engine 4 maken we een gelijkaardig scène van het scenario zoals beschreven in sectie 6.2. De speler bestaat volledig uit een zelfgeschreven type *AIBlueprint* die opgemaakt is uit een C++ klasse en header, zoals te zien in bijlage C.1 en C.2. De header erft af van een klasse *ACharacter* die de karaktriestieken van een karakter kan beïnvloeden. De code zorgt ervoor dat de speler via de pijltjestoetsen en muis kan bewegen. Om de pijltjestoetsen en controle van de computermuis te definiëren is het wel nodig om in de *Project Settings* de nodige aanpassingen door te voeren. Zo worden er vier *Axis Mapping* toegevoegd aan de input van de engine, samen met hun verschillende toetsten en schaal. Deze *Axis Mapping* worden dan in de klasse gebruikt aan de hand van hun naam. De positie van de speler bevindt zich in het midden van het bord. Op de speler worden ook *physics* gesimuleerd. Een basis *PlayerStart* object geeft aan waar de speler begint. Deze bevindt zich ook midden op het bord.

De vijanden bestaan uit basis *Cube* objecten. De AI van deze vijanden worden opgesteld door middel van een *Blueprint*. Hierin wordt de snelheid van de vijanden aangepast, alsook de kleuren. Om de kleuren te kunnen definiëren werden er voor elke kleur, in totaal dus drie, *Materials* aangemaakt en gebruikt in de *Blueprint*. Verder wordt er in de *Blueprint* gedefinieerd dat men de speler moet volgen. De *Blueprints* worden weergegeven in figuur 6.5.

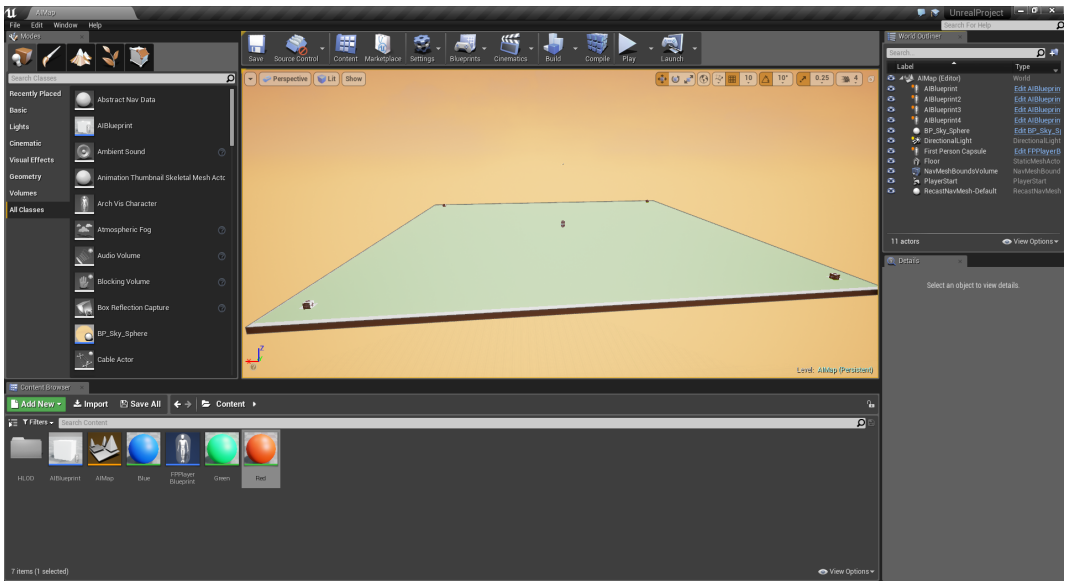
Via een NavMeshBoundsVolume wordt het volume gedefinieerd en berekend door Unreal Engine welke een *ai* kan belopen. Dat komt omdat de *RecastNavMesh* telkens de *NavMesh* berekent en dit op elk bepaald moment kan doorgeven aan de *NavMesh*.



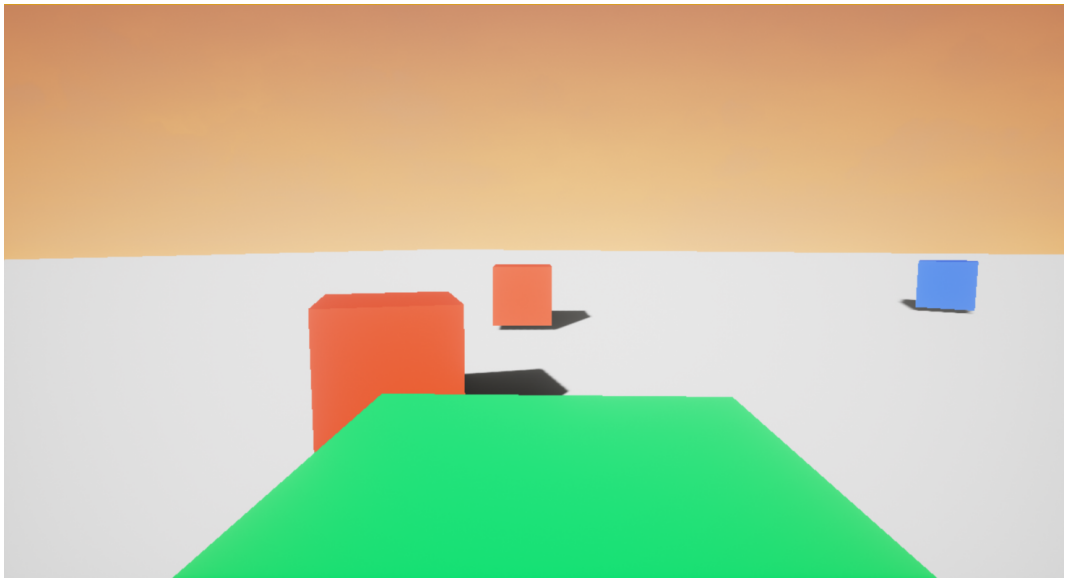
Figuur 6.5: *Blueprint* van om een vijand in Unreal Engine 4

Voor het level zelf wordt er een klasse en header gedefinieerd zoals respectievelijk te bekijken in bijlage C.3 en C.4. De klasse beschrijft onder andere de speler. De header erft hier van *AGameMode* die de spelregels beheert. Het spelbord bestaat uit een basis *Cube* object. Naast de standaard directe verlichting is er ook een standaard *Blueprint* van een *Sky Sphere* aanwezig om te zorgen voor open lucht verlichting.

Het eindresultaat van het scenario in de Unreal Editor voor Unreal Engine 4 is te bezichtigen op figuur 6.6. Ook is er een schermafbeelding te zien van het spel op figuur 6.7.



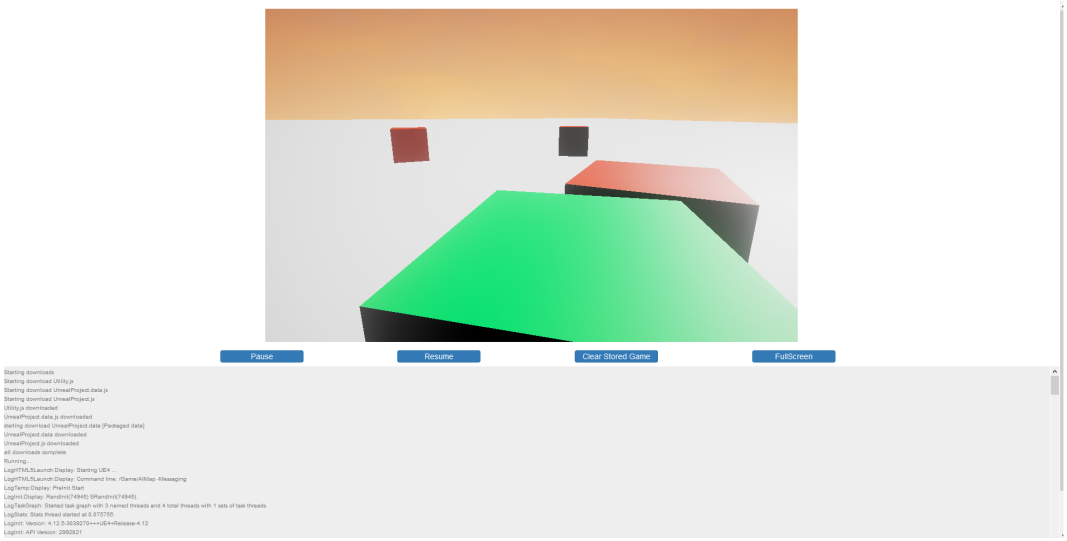
Figuur 6.6: Editor zicht van het afgewerkte scenario in Unreal Engine 4



Figuur 6.7: Schermafbeelding van het prototype in Unreal Engine 4

Bij het laden van de *build* in een browser gebruikt zullen we de resolutie manueel in de HTML5 code aanpassen naar een resolutie van 960 pixels breedte bij 600 pixels hoogte. Onderaan het spel krijgen we extra informatie over de *logging* van het spel. Verder zijn er ook toesten beschikbaaar om het spel te pauzeren of om het spel te starten. Verder kan men ook onderaan een opgeslagen spel verwijderen of volledig

overgaan tot een volledig scherm, zoals te zien in figuur 6.8 toont een productie *build* geopend in de Chrome.



Figuur 6.8: Prototype geladen in de Chrome browser in Unreal Engine 4

Meting 1: Laden van het project: De tijd wordt opgenomen bij de start van het openen van het project bij de *launcher* van Unreal Engine. Wanneer de scène is geladen, stop de tijd. Gemiddeld duurt dit 7,268 seconden zoals te zien in tabel 6.10.

Meting	**Tijd in seconden**
Meting 1	7.09
Meting 2	7,21
Meting 3	7,30
Meting 4	7,59
Meting 5	7,15
Gemiddelde	**7,268**

Tabel 6.10: Unreal Engine Resultaten: Laden van het project

Meting 2: Grootte van het project: Via het eigenschappen venster van Windows komen we de grootte te weten van het project. Het resultaat is 1,92GB. In vergelijking met de grootte van het project bij Unity, gemeten in sectie 6.2.2, zien we een enorme toename. Echter zit bij Unreal Engine de project folder al reeds ingerekend, daarom is de grootte getoond zonder de productie *build* folder om een vergelijkbare meting weer te geven, zoals getoond in tabel 6.11. Desondanks het uitsluiten van de *build* folder, wordt er hier nog steeds gesproken over ongeveer 1,48GB, terwijl het bij Unity slechts over ging over 44,1MB.

Grootte in GB	**1,48**
Grootte in bytes	**1476680354**

Tabel 6.11: Unreal Engine Resultaten: Grootte van het project zonder *build* folder

Meting 3: Bouwen en laden productie *build* in Firefox met cache: Om een productie *build* te bouwen en te laden op , de 64-bits versie zonder de geschiedenis te verwijderen, duurt het gemiddeld 34,306 seconden. Het resultaat is te zien in tabel 6.12.

Meting	**Tijd in seconden**
Meting 1	34,76
Meting 2	32,19
Meting 3	33,22
Meting 4	35,39
Meting 5	35,97
Gemiddelde	**34,306**

Tabel 6.12: Unreal Engine Resultaten: Bouwen en laden productie *build* in Firefox met cache

Meting 4: Bouwen en laden productie *build* in Firefox zonder cache: De recente geschiedenis wordt verwijderd bij elke meting. We noteren de tijd vijf keren bij het bouwen en laden van de productie *build* op de 64-bits versie van Firefox. De meting start wanneer de *build* begint en tot het de scène van het scenario is geladen. Het gehele proces duurt gemiddeld, berekend in tabel 6.13, ongeveer 8 minuten en 27,238 seconden. Vooral het aanmaken van de cache die door het project zelf wordt aangemaakt, is hier enorm tijdrovend in vergelijk met bovenstaande sectie 6.2.2. Unreal Engine 4 maakt voor elke nieuwe *build* steeds nieuwe bestanden aan door middel van *Content Cooking* (Epic Games, g.d.-c). Via dit proces worden er steeds intern enkele bestanden aangemaakt zoals bijvoorbeeld PNGs voor *texture data* (Epic Games, g.d.-c). Echter wordt deze content steeds omgezet naar een formaat dat afhankelijk is aan het platform zelf (Epic Games, g.d.-c). Zoals je dus kan zien in onderstaande tabel 6.13, vraagt dit proces steeds enorm veel tijd in beslag.

Meting	**Tijd**
Meting 1	8m 22,91s
Meting 2	8m 29,79s
Meting 3	8m 27,98s
Meting 4	8m 25,17s
Meting 5	8m 30,34s
Gemiddelde	**8m 27,238s**

Tabel 6.13: Unreal Engine Resultaten: Bouwen en laden productie *build* in Firefox zonder cache

Meting 5: Grootte van de productie *build*: Voor Unreal Engine bedraagt de grootte van de productie *build* alvast stukken meer dan degene van Unity zoals te zien in sectie 6.2.2. Het generen van de productie *build* bevindt zich bij Unreal Engine op dezelfde locatie als de project folder. Deze *build* bedraagt 566MB zoals te zien in tabel 6.14.

Grootte in MB	**566**
Grootte in bytes	**594474310**

Tabel 6.14: Unreal Engine Resultaten: Grootte van de productie *build*

Meting 6: Laden productie *build* in Firefox zonder cache: Gemiddeld duurt het slechts 21,832 seconden, te bezien in tabel 6.15, om de scène van Unreal Engine te laden in Firefox waarbij men steeds de browsergeschiedenis heeft verwijdert zodat er geen cache meer aanwezig is.

Meting	**Tijd in seconden**
Meting 1	21,85
Meting 2	22,39
Meting 3	20,97
Meting 4	21,07
Meting 5	22,88
Gemiddelde	**21,832**

Tabel 6.15: Unreal Engine Resultaten: Laden productie *build* in Firefox zonder cache

Meting 7: Laden productie *build* in Firefox met cache: In deze meting gaat het natuurlijk een deel sneller om de scène van het scenario te laden in de 64-bits versie van Firefox als we de vergelijking maken met de meting in sectie 6.2.2. Na vijf metingen, te bekijken in tabel 6.16, bekomen we een gemiddelde van 6,218 seconden.

Meting	Tijd in seconden
Meting 1	6,44
Meting 2	5,87
Meting 3	6,63
Meting 4	6,17
Meting 5	5,98
Gemiddelde	**6,218**

Tabel 6.16: Unreal Engine Resultaten: Laden productie *build* in Firefox met cache

Meting 8: Laden productie *build* in Chrome zonder cache: Voor de 64-bits versie van Chrome voeren we ook vijf metingen uit om een productie *build* van de scène van het scenario te laden. Hiervoor berekenen we het gemiddelde van deze metingen, te zien in tabel 6.17. Tussen elke meting wordt de geschiedenis gewist zodat er geen cache aanwezig is. Het gemiddelde bedraagt 23,186 seconden.

Meting	Tijd in seconden
Meting 1	22,56
Meting 2	24,30
Meting 3	23,82
Meting 4	23,53
Meting 5	21,72
Gemiddelde	**23,186**

Tabel 6.17: Unreal Engine Resultaten: Laden productie *build* in Chrome zonder cache

Meting 9: Laden productie *build* in Chrome met cache: In tabel 6.18 tonen we de vijf metingen van het laden van de *build* met cache. Het gemiddelde bedraagt 6,596 seconden en toont 16,59 seconden verschil in vergelijking met de sectie 6.2.2.

Meting	Tijd in seconden
Meting 1	6,69
Meting 2	6,25
Meting 3	7,03
Meting 4	6,39
Meting 5	6,62
Gemiddelde	**6,596**

Tabel 6.18: Unreal Engine Resultaten: Laden productie *build* in Chrome met cache

6.3 Het tweede scenario

Het tweede scenario bouwt verder op het eerste scenario zoals beschreven in sectie 6.2. Echter zullen we hiervoor de vijanden die in dat scenario werden aangemaakt vermeerderen tot 100. Deze vijanden worden op willekeurige posities op de grond geplaatst. De vijanden bevinden zich naast elkaar en naast de speler.

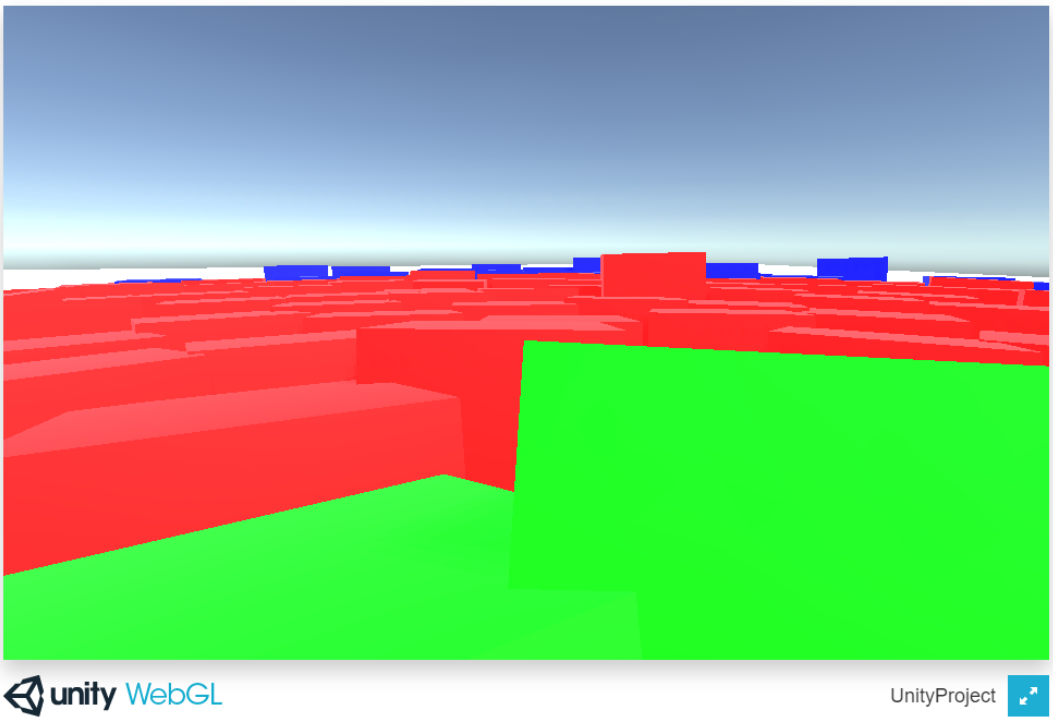
6.3.1 De metingen

Voor elke engine wordt er één meting gedaan om te zien hoe goed de performantie blijft wanneer men vele vijanden op één scherm heeft. Elke meting wordt steeds vijf maal herhaalt op de desktopcomputer beschreven in sectie 6.1. Er wordt enkel getest op Chrome en de cache wordt niet verwijderd. De meting stopt wanneer een eerste vijand groen kleurt. Er wordt ook gerenderd op éénzelfde schermgrootte namelijk 960 pixel bij 600 pixels.

6.3.2 De resultaten

Unity 5

Het kopiëren van de vijanden ging zonder problemen en de editor reageerde normaal. Het resultaat van de browser is te zien op figuur 6.9. Bij het begin van het laden vertoont de browser wat vertraging. De eerste 2 seconden zijn er minder dan 60 FPS.



Figuur 6.9: Het laden van 100 AI vijanden door Unity 5 in Chrome

Meting 10: 100 vijanden: In Unity gaat het laden enorm snel en bedraagt dit slechts 10,802 seconden voor de browser Chrome zoals te zien in tabel 6.19. Er is dus slechts een verschil van 1,562 seconden in vergelijking met meting van het eerste scenario in tabel 6.8.

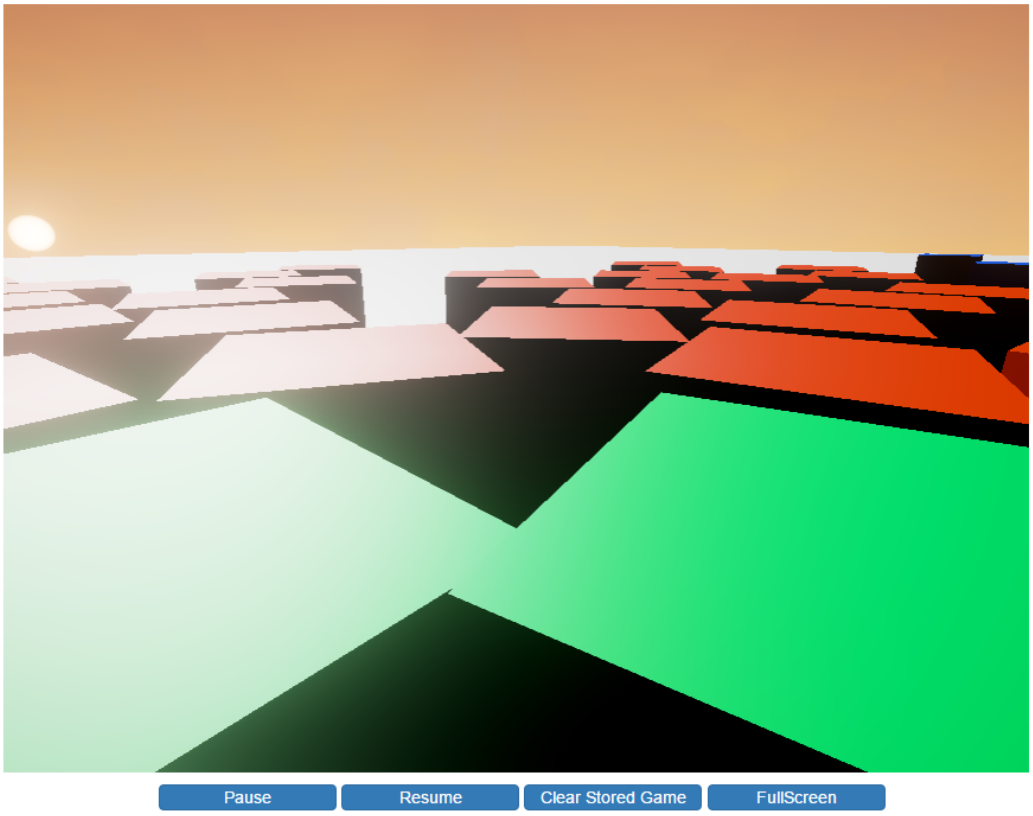
Meting	**Tijd in seconden**
Meting 1	10,35
Meting 2	10,86
Meting 3	11,12
Meting 4	11,04
Meting 5	10,64
Gemiddelde	**10,802**

Tabel 6.19: Unity Resultaten: Laden van het project in Chrome met 100 AI vijanden

Unreal Engine 4

Het kopiëren van de vijanden ging was gepaard met enkele problemen. Zo besloot de editor om af en toe te bevriezen doordat steeds onder andere de omgeving en verlichting werden herberekend. Het eindresultaat in Chrome wordt getoond op figuur

6.10. Tijdens het laden van de omgeving en het bewegen van de vijanden was tijdens de eerste 3 seconden een FPS van minder dan 60.



Figuur 6.10: Het laden van 100 AI vijanden door Unreal Engine 4 in Chrome

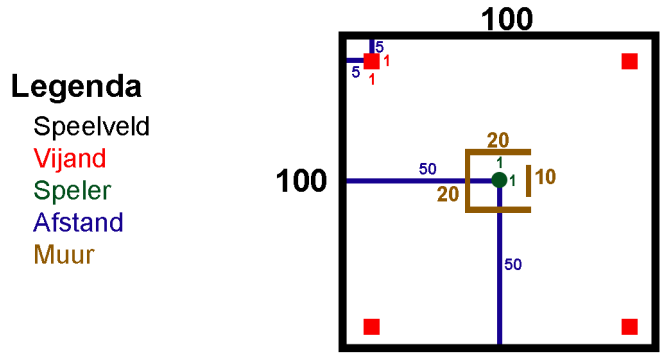
Meting 10: 100 vijanden: In Unreal Engine 4 gaat het laden een stuk trager in vergelijking met Unity 5 zoals te bezichtigen in tabel 6.19. Het duurt dan ook 25,406 seconden langer om het geheel te laden. Het is ook een pak trager dan de meting van het eerste scenario te zien in tabel 6.17, een verschil van 29,612 seconden. Gemiddeld duurt het dus 36,208 seconden zoals getoond in tabel 6.20.

Meting	Tijd in seconden
Meting 1	35,02
Meting 2	36,56
Meting 3	35,89
Meting 4	37,11
Meting 5	36,46
Gemiddelde	**36,208**

Tabel 6.20: Unreal Engine Resultaten: Laden van het project in Chrome met 100 AI vijanden

6.4 Het derde scenario

Voor dit scenario vertrekken we wederom van het eerste scenario, te doornemen in sectie 6.2. Hierbij voegen we enkele muren. Deze muren bevinden zich rondom de speler. De muren hebben steeds een hoogte van 1 eenheid bovenop de grond en langs één zijde van de muur zijn er twee doorgangen. Drie muren staan loodrecht op elkaar met elk een lengte van 20 eenheden, terwijl de vierde muur slechts 10 eenheden heeft. Het speelveld wordt geschetst in figuur 6.11



Figuur 6.11: Het bovenperspectief van het speelbord van het derde scenario

6.4.1 De metingen

We stellen wederom voor beide engines een vergelijkbare omgeving op gebaseerd op het scenario beschreven in sectie 6.4. We testen dit uit met dezelfde desktopcomputer vermeld in sectie 6.1. Via deze metingen willen we te weten komen hoe het *path finding* systeem van beide engines presteert in de Chrome browser met cache. De meting start wanneer het schem in de browser is geladen en stopt wanneer alle vijanden groen kleuren

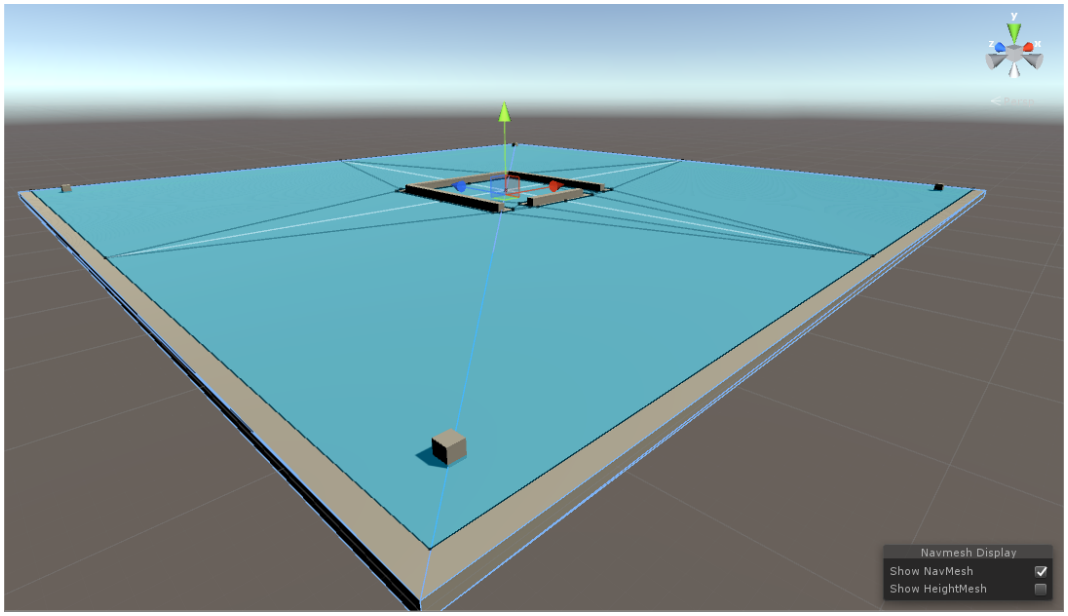
en zich dus bij de speler bevinden. Of indien ze de speler niet vinden of vastzitten tegen de muren. Er wordt ook gerenderd op éénzelfde schermgrootte namelijk 960 pixel bij 600 pixels. De muren hebben volgende posities:

- Muur 1: Breedte is 1 eenheid. Lengte is 20 eenheden en hoogte is 1 eenheid. De muur bevindt zich op 10 eenheden van de speler.
- Muur 2: Breedte is 1 eenheid. Lengte is 20 eenheden en hoogte is 1 eenheid. De muur bevindt zich op 10 eenheden van de speler. Deze muur loopt evenwijdig met muur 1.
- Muur 3: Breedte is 1 eenheid. Lengte is 20 eenheden en hoogte is 1 eenheid. De muur bevindt zich op 10 eenheden van de speler. De muur wordt 90 graden gedraaid en staat loodrecht op muur 1 en muur 2.
- Muur 4: Breedte is 1 eenheid. Lengte is 10 eenheden en hoogte is 1 eenheid. De muur bevindt zich op 10 eenheden van de speler. De muur wordt 90 graden gedraaid en loopt evenwijzig met muur 3.

6.4.2 De resultaten

Unity 5

Naast de objecten vermeld in sectie 6.2.2 worden hier dus vier extra muren bijgeplaatst rond de speler van het type *Cube*. De muren zijn gepositioneerd zoals beschreven in sectie 6.4.1. Aan de muren werd ook een *Nav Mesh Obstacle* toegevoegd, zodat de *Nav Mesh Agent* weet dat men deze muur niet kan bewandelen, dus dat men een andere route zal moeten vinden. Figuur 6.12 geeft de deel van de editor weer waar het derde scenario zichtbaar is. Voor het maken van het project wordt een *Bake* uitgevoerd zodat de *Nav Mesh* correct is berekend.



Figuur 6.12: Het derde scenario in de editor van Unity 5

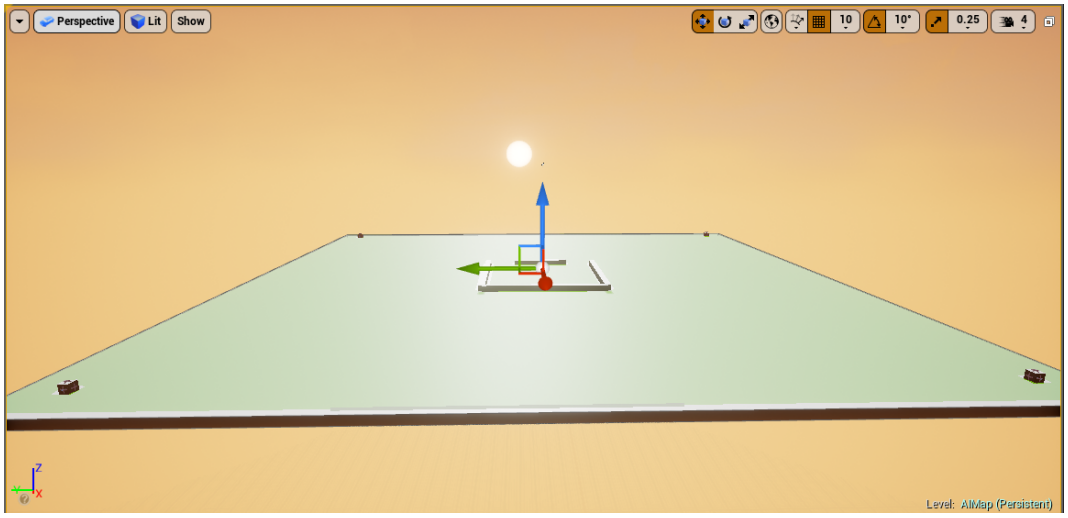
Meting 11: Het *path finding* systeem: Het project wordt gestart in de Chrome browser en de chronometer start wanneer de omgeving is ingeladen. De tijd stopt wanneer alle vijanden groen kleuren of niet meer bewegen. Na vijf metingen bekomen we een gemiddelde van 12,926 seconden zoals te zien in tabel 6.21.

Meting	**Tijd in seconden**
Meting 1	12,51
Meting 2	12,81
Meting 3	13,06
Meting 4	13,27
Meting 5	12,98
Gemiddelde	**12,926**

Tabel 6.21: Unity Resultaten: Het *path finding* systeem

Unreal Engine 4

Ook bij Unreal Engine 4 komen er vier extra muren bij zoals beschreven in op figuur 6.11. De vier muren zijn gepositioneerd zoals vermeld in sectie 6.4.1. Doordat deze muren statische *Cube* objecten zijn, zal automatisch de *Nav Mesh* worden geüpdatet doordat het *Nav Mesh Bound Volume* ook deze muren bevat. Het derde scenario nagemaakt in de editor wordt getoond op figuur 6.13.



Figuur 6.13: Het derde scenario in de editor van Unreal Engine 4

Meting 11: Het *path finding* systeem: Wanneer we de Chrome browser starten, wachten we tot de omgeving is geladen om de tijd te meten. We stoppen de tijd wanneer alle vier de vijanden de speler hebben gevonden en herhalen dit vijf keer. We bekomen hier een gemiddelde van 9,89 seconden zoals te lezen in tabel 6.22.

Meting	**Tijd in seconden**
Meting 1	9,58
Meting 2	10,17
Meting 3	9,89
Meting 4	10,08
Meting 5	9,73
Gemiddelde	**9,89**

Tabel 6.22: Unreal Engine Resultaten: Het *path finding* systeem

6.5 Overzicht

Hieronder in tabel 6.23 het complete overzicht van de metingen zowel in Unity 5, als in Unreal Engine 4.

Soort meting	**Unity 5**	**Unreal Engine 4**
Laden van het project	6,06s	7,268s
Grootte van het project	44,1MB	1477MB
Bouwen en laden productie *build* 3 (met cache)	2m 15,43s	34,306s
Bouwen en laden productie *build* 3 (zonder cache)	2m 40,014s	8m 27,238s
Grootte van de productie *build*	4,7MB	566MB
Laden productie *build* (Firefox met cache)	6,21s	6,218s
Laden productie *build* (Firefox zonder cache)	9,104s	21,832s
Laden productie *build* (Chrome met cache)	9,24s	6,596s
Laden productie *build* (Chrome zonder cache)	10,866s	23,186s
100 vijanden (Chrome met cache)	10,802s	36,208s
Het *path finding* systeem (Chrome met cache)	12,926s	9,89s

Tabel 6.23: Overzicht van de resultaten van de metingen voor Unity en Unreal Engine

Hoofdstuk 7

Conclusie

Als men met een game engine wilt werken zijn er héél véél verschillende mogelijkheden. Wanneer we kijken naar de eigenschappen die Rusty Bolt verwacht, zullen deze anders zijn dan verwachtingen van een ander bedrijf met andere personeelsleden en een verschillend project. De grootste eisen zijn natuurlijk deze die moeten voldoen aan het project. De engines moeten HTML5 en WebGL ondersteunen, maar ook eisen waarin men ervaring heeft zoals programmeren talen C++ of C#. Natuurlijk als men wil overschakelen naar een nieuwe engine, verwacht men ook dat de documentatie up-to-date is. Deze documentatie moet vooral duidelijk zijn in gebruik. Wanneer men toch in de problemen zit, moet men ook terecht kunnen bij de ontwikkelaar van de software. Andere eigenschappen zijn natuurlijk om de ontwikkeling van het spel zelf te behelpen zoals speciale tools voor AI en *physics*. Ondanks dat we eigenlijk op twee alternatieven zijn gekomen, zijn er toch nog andere alternatieven zoals Blend4web of Unigine 2 voor 3D ontwikkeling indien men meer budget had of wanneer men bepaalde features zou laten vallen. Maar Unity 5 en Unreal Engine 4 zijn wel de enige twee die aan alle eisen behalve één voldoen.

Unreal Engine 4 heeft geen support voor cloud omgevingen. Terwijl Unity 5 hun broncode niet gratis ter beschikking stelt. Beide engines hebben wel gratis versies. Bij Unity 5 ontwikkelingen kun je tot 100.000 dollar verdienen zonder iets te moeten investeren in de game engine, terwijl bij Unreal Engine 4 dat al vanaf de helft is. Qua flexibiliteit voor een licentie heeft Unity 5 vier mogelijkheden, terwijl Unreal Engine 4 er slechts twee aanbiedt. Visueel verschillen ze ook enorm en de uitwerking om AI via tools aan de ontwikkelaars te brengen is ook verschillend. Zo ligt de focus van Unity 5 meer op het *path finding* systeem, terwijl Unreal Engine 4 dit uitbreidt met *Behavior Trees* en *EQS*. Unity 5 kan op meerdere besturingssystemen gedraaid worden en heeft ook lagere systeemvereisten dan deze van Unreal Engine 4. Unity 5 ondersteunt ook meer dan één programmeertaal, namelijk C# en JS. Terwijl Unreal Engine 4 enkel C++ ondersteunt. Unreal Engine 4 zorgt wel voor een tool om zonder enige kennis van programmeren een project te maken via hun *Blueprint Visual Scripting*.

Unreal Engine 4 mag misschien al langer in ontwikkeling zijn, qua marktaandeel staat Unity wel bovenaan. Unreal Engine 4 wint wel aan populariteit sinds deze gratis werd aangeboden. Unreal Engine 4 daarentegen wordt wel meer gewaardeerd in de industrie als we *The Tech List* van 2014 mogen geloven. Beide engines gebruiken wel dezelfde Low Level Virtual Machine (LLVM) naar JS compiler, namelijk emscripten. Echter gebruikt de Unreal Engine 4 een iets recentere versie.

Wanneer we het prototype bekijken, zien we duidelijk enige verschillen in de grootte voor de projecten. Zo is de plaats die nodig is om het geheel op te slaan wel aanzienlijk groter bij Unreal Engine 4 dan bij Unity 5. Het laden van het project van het prototype verschilt slechts ongeveer 1 seconde met een voorsprong voor Unity 5. Het prototype van de Unreal Engine 4 laadt dan weer ongeveer gelijk op Chrome als op Firefox wanneer men cache toe laat, zonder cache duurt Chrome bijna 2 seconden langer. Wanneer we eigenlijk de scène laden vanuit de editor, duurt het bij Unity gemiddeld tussen de 2 à 3 minuten. Bij Unreal Engine duurt het slechts 34 seconden wanneer we cache gebruiken, maar wel meer dan 8 minuten zonder cache. Wanneer we het prototype uitbreiden naar meerdere vijanden zien we toch een groter verschil in performantie. Zo duurt het bij Unreal Engine 4 al meer dan 29 seconden langer wanneer we van 4 naar 100 vijanden gaan. Bij Unity 5 duurt het slechts iets meer dan 1 seconde voor Chrome die gebruik maakt van de cache. Als we dan dieper ingaan op het *path finding* systeem zien we dat de vijanden in *Unreal Engine 4* sneller hun pad vinden naar de speler dan in Unity 5. Het verschil bedraagt iets meer dan 3 seconden.

Als we ons nu de vraag stellen welke game engine Rusty Bolt zelf gebruikt, dan antwoorden we democratisch dat het eigenlijk een persoonlijke keuze is. Men kijkt beter naar de ervaring die men heeft met de game engine en eventueel met de ondersteunende programmeertalen. Unreal Engine 4 heeft wel een uitgebreider gamma van tools om te helpen met het implementeren van de AI. Het lijkt op het eerste zicht wel iets sneller dan Unity 5. Maar de keuze is ook afhankelijk van project. Het is misschien commercieel gunstiger om te investeren in één welbepaalde game engine. Indien men enkele verkoopverwachtingen heeft, houdt men best ook rekening met de auteursrechten die betaald moeten worden. Ook zal men best kijken waarvoor men het project wil realiseren, misschien wil men naast Chrome en Firefox nog andere browsers ondersteunen. Het laden van een prototype zal ook afhankelijk zijn van de computers die men beschikbaar stelt in het bedrijf. De ervaring van een realistisch project zal ook anders lopen dan het gegeven prototype. Sowieso hebben beide game engines genoeg features en hulpmiddelen om te voldoen aan de eisen die Rusty Bolt verwacht. Ze zullen de taak uiteindelijk kunnen uitvoeren door de ontwikkelaars van het bedrijf, dus beide zijn zeker geen slechte keuze.

Bibliografie

Aleksandr. (2014, september 3). *Documentation, Unity scripting languages and you*. Geraadpleegd op 6 augustus 2016, via http://blogs.unity3d.com/2014/09/03/documentation-unity-scripting-languages-and-you/

Batchelor, J. (2015). *Unreal Engine 4 « 2. The Tech List*, 8.

Batchelor, J. (2016, maart 17). *Tim Sweeney reveals that seven Unreal-powered have grossed more than $1bn*. Geraadpleegd op 7 augustus 2016, via http://www.develop-online.net/news/2015-was-the-best-year-ever-for-unreal/0218044

Bleszinski, C. (2010, februari 23). *History of the Unreal Engine*. Geraadpleegd op 6 augustus 2016, via http://www.ign.com/articles/2010/02/23/history-of-the-unreal-engine

Bloomberg Business. (2015, augustus 5). *Bigger Than Hollywood: The Numbers Behind Video Gaming*. doi:http://bloom.bg/1SMpmiR

Busby, J., Parrish, Z. & Wilson, J. (2009). *Mastering Unreal Technology, Volume I*. Sams Publishing.

Calunod, J. (2016, juli 13). *Web Browser Market Share Wars for June 2016: Google Chrome vs Microsoft Internet Explorer vs Mozilla Firefox vs Edge vs Safari*. Geraadpleegd op 1 augustus 2016, via http://www.christianpost.com/news/web-browser-market-share-wars-for-june-2016-google-chrome-vs-microsoft-internet-explorer-vs-mozilla-firefox-vs-edge-vs-safari-166387/

Chayes, J. (2014, juni 19). *Card Life*. Geraadpleegd op 3 april 2016, via http://unity3d.com/showcase/case-stories/hearthstone

Cludts, D. (2015, juni 1). *Game Mania lanceert eerste Belgische eSports-toernooi*. Geraadpleegd op 21 december 2015, via http://www.zdnet.be/nieuws/167606/game-mania-lanceert-eerste-belgische-esports-toernooi/

Cowley, D. (2015, juni 10). *Unreal Engine 4.8 Released!* Geraadpleegd op 7 augustus 2016, via https://www.unrealengine.com/blog/unreal-engine-48-released

Coyote, R. (2013, mei 16). *Why Are Most Indie Games 2D Instead of 3D?* Geraadpleegd op 23 december 2015, via http://rampantgames.com/blog/?p=5934

Crecente, B. (2015, augustus 11). *Superdata: Hearthstone pulls in $20 million a month as it disrupts the card game industry*. Geraadpleegd op 3 april 2016, via http:

//www.polygon.com/2015/8/11/9130779/superdata-hearthstone-pulls-in-20-million-a-month-as-it-disrupts-the

Dyer, M. (2014, maart 19). *GDC: Epic Games' Unreal Engine 4 adopts subcription model*. Geraadpleegd op 7 augustus 2016, via http://www.ign.com/articles/2014/03/19/gdc-epic-games-unreal-engine-4-adopts-subscription-model

Dyer, M. (2016, juni 1). *Unreal Engine 4.12 Released!* Geraadpleegd op 7 augustus 2016, via https://www.unrealengine.com/blog/unreal-engine-4-12-released

Echterhoff, J. (2014, april 29). *On the future of Web publishing in Unity*. Geraadpleegd op 29 juli 2016, via http://blogs.unity3d.com/2014/04/29/on-the-future-of-web-publishing-in-unity/

Emscripten Contributors. (2016, juli 7). *If you love something, set it free*. Geraadpleegd op 6 augustus 2016, via http://kripken.github.io/emscripten-site/docs/introducing_emscripten/release_notes.html

Epic Games. (g.d.-a). *Behavior Trees*. Geraadpleegd op 14 augustus 2016, via https://docs.unrealengine.com/latest/INT/Engine/AI/BehaviorTrees/

Epic Games. (g.d.-b). *Blueprints Visual Scripting*. Geraadpleegd op 14 augustus 2016, via https://docs.unrealengine.com/latest/INT/Engine/Blueprints/

Epic Games. (g.d.-c). *Content Cooking*. Geraadpleegd op 21 augustus 2016, via https://docs.unrealengine.com/latest/INT/Engine/Deployment/Cooking/

Epic Games. (g.d.-d). *Environment Query System*. Geraadpleegd op 14 augustus 2016, via https://docs.unrealengine.com/latest/INT/Engine/AI/EnvironmentQuerySystem/

Epic Games. (g.d.-e). *Getting Started: Developing HTML5 Projects*. Geraadpleegd op 6 augustus 2016, via https://docs.unrealengine.com/latest/INT/Platforms/HTML5/GettingStarted/

Epic Games. (g.d.-f). *How Unreal Engine 4 Behavior Trees Differ*. Geraadpleegd op 14 augustus 2016, via https://docs.unrealengine.com/latest/INT/Engine/AI/BehaviorTrees/HowUE4BehaviorTreesDiffer/index.html

Epic Games. (g.d.-g). *Navmesh Content Examples*. Geraadpleegd op 14 augustus 2016, via https://docs.unrealengine.com/latest/INT/Resources/ContentExamples/NavMesh/index.html

Epic Games. (g.d.-h). *UDK Licensing*. Geraadpleegd op 8 augustus 2016, via https://wiki.unrealengine.com/Recommended_Hardware

Epic Games. (2005, augustus 18). *Rein: 'We've been working on Unreal Engine 4 for two years'*. Geraadpleegd op 6 augustus 2016, via https://web.archive.org/web/20140110101539/http://www.computerandvideogames.com/123639/rein-weve-been-working-on-unreal-engine-4-for-two-years/

Epic Games. (2013, januari 29). *Epic Games Releases "Epic Citadel" for Android*. Geraadpleegd op 6 augustus 2016, via https://www.epicgames.com/news/epic-games-releases-epic-citadel-for-android/

Epic Games. (2014). *Tappy Chicken*. Geraadpleegd op 29 juli 2016, via https://www.unrealengine.com/html5/

Epic Games. (2016a). *About Unreal Engine 4*. Geraadpleegd op 7 augustus 2016, via https://www.unrealengine.com/unreal-engine-4

Epic Games. (2016b). *UDK Licensing*. Geraadpleegd op 7 augustus 2016, via https://www.unrealengine.com/previous-versions/udk-licensing-resources

Epic Games. (2016c). *What is Unreal Engine 4?* Geraadpleegd op 29 juli 2016, via https://www.unrealengine.com/what-is-unreal-engine-4

Futter, M. (2016, mei 18). *Batman: Return to Arkham*. Geraadpleegd op 6 augustus 2016, via http://www.gameinformer.com/games/batman_return_to_arkham/b/playstation4/archive/2016/05/18/batman-return-to-arkham-is-real-with-both-games-using-unreal-engine-4.aspx

Grubb, J. (2015, februari 24). *Unreal 4 gets native HTML5 exporting and support for giant worlds*. Geraadpleegd op 25 januari 2015, via http://venturebeat.com/2015/02/24/unreal-4-gets-native-html5-exporting-and-support-for-giant-worlds/

Haas, J. (2014). *A History of the Unity Game Engine* (masterscriptie, Worcester Polytechnic Institute, 100 Institute Rd, Worcester, MA 01609, United States).

Hagedoorn, H. (2016, februari 8). *Video: Real-Time Cinematography in Unreal Engine 4*. Geraadpleegd op 8 augustus 2016, via http://www.guru3d.com/news-story/video-real-time-cinematography-in-unreal-engine-4.html

Helgason, D. (2005, juni 6). *Unity Technologies Delivers Unity 3*. Geraadpleegd op 3 april 2016, via http://www.marketwired.com/press-release/unity-technologies-delivers-unity-3-1325564.htm

James, P. (2016, juni 6). *Unreal Engine 4.12 is Out Now, Gets 'VR Editor', OSVR and Google VR Support*. Geraadpleegd op 25 januari 2015, via http://www.roadtovr.com/unreal-engine-4-12-is-out-now-gets-vr-editor-osvr-and-google-vr-support/

Jarvis, M. (2015, december 8). *Unity 5 makes WebGL support official*. Geraadpleegd op 16 april 2016, via http://deve1op.net/1Nfuynw

Jarvis, M. (2016, juni 17). *Rocksteady moves to Unreal Engine 4 for Batman Arkham VR*. Geraadpleegd op 6 augustus 2016, via http://www.develop-online.net/news/rocksteady-moves-to-unreal-engine-4-for-batman-arkham-vr/0221876

Jones, B. (2016, juli 28). *Microsoft releases Unreal Engine 4 fork with Universal Windows*. Geraadpleegd op 7 augustus 2016, via http://www.digitaltrends.com/gaming/unreal-engine-4-universal-windows-support-microsoft/

Marketwired L.P. (2010a, september 11). *Unity Technologies Delivers Unity 3*. Geraadpleegd op 3 april 2016, via http://www.marketwired.com/press-release/unity-technologies-delivers-unity-3-1325564.htm

Marketwired L.P. (2010b, november 1). *Unity Technologies Surpasses 250K Developers Milestone and 35M Installs of Free Unity Web Player*. Geraadpleegd op 10 april 2016, via http://www.marketwired.com/press-release/unity-technologies-surpasses-250k-developers-milestone-35m-installs-free-unity-web-player-1344670.htm

Marketwired L.P. (2012a, juni 12). *The Next Generation of the Unity Game Engine Unveiled*. Geraadpleegd op 10 april 2016, via http://www.marketwired.com/press-release/the-next-generation-of-the-unity-game-engine-unveiled-1670512.htm

Marketwired L.P. (2012b, april 9). *Unity Reaches One Million Registered Developers*. Geraadpleegd op 10 april 2016, via http://www.marketwired.com/press-release/unity-reaches-one-million-registered-developers-1641486.htm

Marketwired L.P. (2013, augustus 28). *Unity Reveals 2D Tools*. Geraadpleegd op 10 april 2016, via http://www.marketwired.com/press-release/Unity-Reveals-2D-Tools-1825422.htm

Masters, M. (2015, april 7). *Discover the New Unity 5 Features*. Geraadpleegd op 16 april 2016, via http://blog.digitaltutors.com/new-unity-5-features/

Matulef, J. (2016, augustus 12). *No Man's Sky isn't going over well on Steam*. Geraadpleegd op 14 augustus 2016, via http://www.eurogamer.net/articles/2016-08-12-no-mans-sky-isnt-going-over-well-on-steam

Noland, M. (2014, mei 22). *Shipping Tappy Chicken*. Geraadpleegd op 6 augustus 2016, via https://www.unrealengine.com/blog/shipping-tappy-chicken

Nutt, C. (2014, maart 21). *Unreal Engine 2*. Geraadpleegd op 6 augustus 2016, via http://www.gamasutra.com/view/news/213647/Epics_Tim_Sweeney_lays_out_the_case_for_Unreal_Engine_4.php

Nutt, C. (2016, maart 21). *How Unity is pushing VR and better 3D graphics with its game engine*. Geraadpleegd op 16 april 2016, via http://www.gamasutra.com/view/news/237853/Unity_5_released_with_upgrades_fullfeatured_free_version.php

Parrish, K. (2011, februari 25). *Ubisoft: 3DS Can Handle Unreal Engine 2*. Geraadpleegd op 6 augustus 2016, via http://www.tomsguide.com/us/Unreal-Engine-2-Splinter-Cell-3DS-Nitendo-3DS,news-10601.html

Peterson, J. (2015, mei 6). *An introduction to ILZCPP internals*. Geraadpleegd op 27 juli 2016, via http://blogs.unity3d.com/2015/05/06/an-introduction-to-ilcpp-internals/

Pettit, N. (2013, september 3). *3D in the Browser: WebGL versus CSS 3D Transforms*. Geraadpleegd op 18 april 2016, via http://blog.teamtreehouse.com/3d-in-the-browser-webgl-versus-css-3d-transforms

Plante, C. (2012, april 2). *Better with age: A history of Epic Games*. Geraadpleegd op 6 augustus 2016, via http://www.polygon.com/2012/10/1/3438196/better-with-age-a-history-of-epic-games

Rusty Bolt. (g.d.). *Rusty Bolt BVBA*. Geraadpleegd op 29 juli 2016, via http://rustybolt.be/

Seyler, B. (2010, mei 19). *Google, Android, and the futher of games on the web*. Geraadpleegd op 10 april 2016, via http://blogs.unity3d.com/2010/05/19/google-android-and-the-future-of-games-on-the-web/

Shaw, P. (2012, februari 27). *Unreal Engine 4 behind closed doors at GDC*. Geraadpleegd op 6 augustus 2016, via http://www.wired.com/2012/02/unreal-engine-4-gdc/

Sieprawski, B. (2015, februari 24). *Unreal Engine 4.7 Released!* Geraadpleegd op 7 augustus 2016, via https://www.unrealengine.com/blog/unreal-engine-47-released

Simpson, C. (2014, juli 17). *Behavior trees for AI: How they work*. Geraadpleegd op 14 augustus 2016, via http://www.gamasutra.com/blogs/ChrisSimpson/20140717/221339/Behavior_trees_for_AI_How_they_work.php

Sweeney, T. (2015, maart 2). *If you love something, set it free*. Geraadpleegd op 25 januari 2015, via https://www.unrealengine.com/blog/ue4-is-free

Takahashi, D. (2015, maart 3). *Unity 5 released with upgrades, full-featured free version*. Geraadpleegd op 16 april 2016, via http://venturebeat.com/2016/03/21/clive-downie-interview/

Thier, D. (2012, juni 29). *Epic's Tim Sweeney on How Unreal Engine 4 Will Change The Way Games Are Made, and Why You Care*. Geraadpleegd op 6 augustus 2016, via http://www.forbes.com/sites/davidthier/2012/06/29/epics-tim-sweeney-on-how-unreal-engine-4-will-change-the-way-games-are-made-and-why-you-care/

Tnw Deals. (2016, maart 24). *This engine is dominating the gaming industry right now*. Geraadpleegd op 31 juli 2016, via http://tnw.to/e4mGGl

Tuffin, S. (2016, april 27). *Recordaantal bezoekers Vlaamse film in 2015*. Geraadpleegd op 14 augustus 2016, via http://vertigoweb.be/recordaantal-bezoekers-vlaamse-film-2015/

Unity Technologies. (2014, maart 18). *Unity 5 announced at GDC 2014, pre-order begins*. Geraadpleegd op 10 april 2016, via https://unity3d.com/company/public-relations/news/unity-announces-unity5

Unity Technologies. (2015a, maart 3). *Unity 5 is here*. Geraadpleegd op 16 april 2016, via http://unity3d.com/company/public-relations/news/unity-5-here

Unity Technologies. (2015b, december 8). *Unity 5 makes WebGL support official*. Geraadpleegd op 16 april 2016, via http://deve1op.net/1Nfuynw

Unity Technologies. (2016a). *Building a NavMesh*. Geraadpleegd op 21 augustus 2016, via https://docs.unity3d.com/Manual/nav-BuildingNavMesh.html

Unity Technologies. (2016b). *Building Height Mesh for Accurate Character Placement*. Geraadpleegd op 21 augustus 2016, via https://docs.unity3d.com/Manual/nav-HeightMesh.html

Unity Technologies. (2016c). *Get Unity*. Geraadpleegd op 16 april 2016, via https://unity3d.com/get-unity

Unity Technologies. (2016d). *Inner Workings of the Navigation System*. Geraadpleegd op 14 augustus 2016, via https://docs.unity3d.com/Manual/nav-InnerWorkings.html

Unity Technologies. (2016e). *Loading Multiple NavMeshes using Additive Loading*. Geraadpleegd op 21 augustus 2016, via https://docs.unity3d.com/Manual/nav-AdditiveLoading.html

Unity Technologies. (2016f). *Navigation Areas and Costs*. Geraadpleegd op 21 augustus 2016, via https://docs.unity3d.com/Manual/nav-AreasAndCosts.html

Unity Technologies. (2016g). *Navigation System in Unity*. Geraadpleegd op 31 juli 2016, via https://docs.unity3d.com/Manual/nav-NavigationSystem.html

Unity Technologies. (2016h). *SYSTEM REQUIREMENTS FOR UNITY 5.4*. Geraadpleegd op 31 juli 2016, via https://unity3d.com/unity/system-requirements

Unity Technologies. (2016i). *The leading global game industry software*. Geraadpleegd op 10 april 2016, via https://unity3d.com/public-relations

Unity Technologies. (2016j). *Unity*. Geraadpleegd op 3 april 2016, via http://unity3d.com/unity

Unity Technologies. (2016k). *Unity Multiplatform*. Geraadpleegd op 3 april 2016, via http://unity3d.com/unity/multiplatform

Unity Technologies. (2016l). *Using NavMesh Agent with Other Components*. Geraadpleegd op 21 augustus 2016, via https://docs.unity3d.com/Manual/nav-MixingComponents.html

Unity Technologies. (2016m). *Welcome to Unity*. Geraadpleegd op 29 juli 2016, via https://store.unity.com/

Waters, K. (2009, januari 12). *Prioritization using MoSCoW*. Geraadpleegd op 27 juli 2016, via http://www.allaboutagile.com/prioritization-using-moscow/

Wingfield, N. (2016, juli 26). *Unity Technologies, Maker of Pokémon Go Engine, Swells in Value*. Geraadpleegd op 13 juli 2016, via http://www.nytimes.com/2016/07/14/technology/unity-technologies-maker-of-pokemon-go-engine-swells-in-value.html

Lijst van figuren

Lijst van tabellen

Woordenlijst

Adobe Flash Een multimediaplatform van Adobe Systems. 18

Android Het mobiele besturingssysteem van Google. 17, 29

API Een programmeerbare interface die een set van regels definieert waarmee een softwareproduct kan communiceren. 25

Boo Een objectgeoriënteerde programmeertaal geïnspireerd door de Python syntaxis. 18

C# Een objectgeoriënteerde programmeertaal ontwikkeld door Microsoft. 1, 8, 10, 12–14, 18, 25, 42, 44, 65, 81–90, 92

C++ Een objectgeoriënteerde en gestandaardiseerde programmeertaal ontwikkeld door Bell Labs. 1, 8, 10, 12–14, 25, 28–30, 42, 50, 65, 81–90, 92

CCG Een verzamelkaartspel. 17

Chrome Een browser van Google. 1, 17, 19, 31, 33, 41, 43–46, 49, 53, 56–58, 60, 62–64, 66, 73

DirectX API van Microsoft gericht op multimedia projecten. 19

Edge Een browser van Microsoft. 19

eSports Een toernooi dat zich online wereld afspeelt door middel van games. 4

Firefox Een browser van Mozilla. 1, 19, 31, 41, 43, 44, 47–49, 54, 55, 64, 66

GPU Een processor die verantwoordelijk is voor alles wat met video heeft te maken. 19

iPhone Een serie van smartphones ontwikkeld door Apple. 17, 29

JS Een objectgeoriënteerde programmeertaal geïnspireerd door de Java syntaxis. 1

Linux Een besturingssysteem gebaseerd op Unix. 18, 77

Mac OS Een besturingssysteem van Apple. 19, 30

Middleware Een systeem dat softwarecomponenten met elkaar laat communiceren. 17

MoSCoW Een methode die eisen opdeelt volgens prioriteit. 6, 10, 11

.NET Een framework ontwikkeld door Microsoft. 25

OS Een besturingssysteem. 19

Safari Een browser van Apple. 19

SteamOS Een besturingssysteem gebaseerd op Linux. 19

Ubuntu Een besturingssysteem gebaseerd op Linux. 19

Wii Een spelconsole van Nintendo. 17

Windows Een besturingssysteem van Microsoft. 11, 19, 20, 30, 31, 41, 47, 53, 79

WWDC Een jaarlijkse conferentie van Apple voor ontwikkelaars waar nieuwe ontwikkeling van Apple worden getoond. 17

Acroniemen

AI Artificiële intelligentie. 1, 5, 7, 9–11, 15, 21, 30, 34, 35, 41, 44, 45, 51, 65, 66

AMD Advanced Micro Devices. 31, 41

AOT Ahead-of-time. 25

API Application programming interface. 25, 29, 76, *Woordenlijst:* API

BVBA Besloten vennootschap met beperkte aansprakelijkheid. 5

CCG Collectable card game. 17, *Woordenlijst:* CCG

EQS Environment Query System. 34, 35, 65

FPS Frames per seconde. 43, 57, 59

GB Gigabyte. 30, 41, 53, 54

GHz Gigahertz. 31, 41

GPU Graphics processing unit. 19, 41, *Woordenlijst:* GPU

HD Hoge definitie. 29

HTML5 HyperText Markup Language 5. 1, 4–6, 8, 10, 12–14, 19, 25, 28–31, 44, 52, 65, 81–86, 88–93

ICT Informatie- en communicatietechnologie. 1

IL2CPP Intermediate Language to C++. 25

JPEG Joint Photographic Experts Group. 9

JS JavaScript. 1, 18, 25, 65, 66, *Woordenlijst:* JS

LLVM Low Level Virtual Machine. 66

MB Megabyte. 47, 48, 53, 55, 64

NaCL Native Client. 17

OS Operation system. 19, 30, 77, *Woordenlijst:* OS

PNG Portable Network Graphics. 9, 54

PSD Photoshop Document. 9, 11, 15

RAM Random Access Memory. 19, 30, 41

UDK Unreal Development Kit. 30

UWP Universal Windows Platform. 30

VAF Vlaams Audiovisueel Fonds. 4

VR Virtuele realiteit. 17, 28–30

WebGL Web Graphics Library. 1, 5, 7, 8, 10, 12–14, 18, 19, 25, 28, 30, 31, 44, 65, 81–86, 88, 89

WWDC Apple Worldwide Developers Conference. 17, *Woordenlijst:* WWDC

Appendices

Bijlage A

Lange Lijst (vervolg)

A.1 Voldoen niet aan alle *must have* eisen

Deze game engines voldoen niet aan alle eisen die er zeker moeten inzitten. Enkel wanneer er 3 of meer eisen niet worden aan voldaan worden deze opgelijst.

- Voldoet niet aan drie *must have* eisen
  - Antiryad Gx v3.5
    - Officiële website: http://www.arkham-development.com/
    - Ondersteuning voor ontwikkeling in 3D
    - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    - Support door ontwikkelaar
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    - Mogelijkheid om te debuggen
  - Construct 2 r227
    - Officiële website: https://www.scirra.com/construct2
    - Ondersteuning voor HTML5
    - Ondersteuning voor WebGL
    - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    - Support door ontwikkelaar
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    - Mogelijkheid om te debuggen

- CopperCube
  - Officiële website: http://www.ambiera.com/coppercube/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
- CryEngine
  - Officiële website: https://www.cryengine.com/
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- GameSalad Creator v. 1.0.0
  - Officiële website: http://gamesalad.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor WebGL
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
  - Support door ontwikkelaar
  - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Mogelijkheid om te debuggen
- Enchant.js v0.8.0
  - Officiële website: http://enchantjs.com/
  - Ondersteuning voor HTML5
  - Ondersteuning voor ontwikkeling in 3D
  - Ondersteuning voor WebGL
  - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200

    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Marmalade Platform 8.5
    * Officiële website: https://www.madewithmarmalade.com/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- Ogre 2.1
    * Officiële website: http://www.ogre3d.org/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning voor WebGL
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- PlayCanvas
    * Officiële website: https://playcanvas.com/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- ShiVa 1.9.2
    * Officiële website: http://www.shiva-engine.com/
    * Ondersteuning voor ontwikkeling in 3D

        * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
        * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
        * Support door ontwikkelaar
        * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
        * Mogelijkheid om te debuggen

- Voldoet niet aan vier *must have* eisen
    - Clickteam Fusion 2.5
        * Officiële website: http://www.clickteam.com/
        * Ondersteuning voor HTML5
        * Ondersteuning voor WebGL
        * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
        * Support door ontwikkelaar
        * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    - CocoonJS
        * Officiële website: https://www.ludei.com/cocoonjs/
        * Ondersteuning voor HTML5
        * Ondersteuning voor WebGL
        * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
        * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
        * Mogelijkheid om te debuggen
    - Cocos2D-x V3.12
        * Officiële website: http://www.cocos2d-x.org/
        * Ondersteuning voor HTML5
        * Ondersteuning voor WebGL
        * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
        * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
        * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    - Godot Engine 2.0.4.1

    * Officiële website: https://godotengine.org/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Leadwerks 4.1
    * Officiële website: http://www.leadwerks.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- LibGDX 1.9.3
    * Officiële website: https://libgdx.badlogicgames.com/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- MonoGame 3.5
    * Officiële website: http://www.monogame.net/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- RPG Maker
    * Officiële website: http://www.rpgmakerweb.com/

    * Ondersteuning voor HTML5
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- StepMania 5.0.11
    * Officiële website: http://www.stepmania.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- Three.js
    * Officiële website: http://threejs.org/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Voxel.js
    * Officiële website: http://voxeljs.com/
    * Ondersteuning voor HTML5
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- V-Play
    * Officiële website: https://v-play.net/
    * Ondersteuning voor HTML5

    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- Voldoet niet aan vijf *must have* eisen
  - Cafu
    * Officiële website: http://www.cafu.de/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - ClanLib
    * Officiële website: http://www.clanlib.org
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Flare3D v2.7.00
    * Officiële website: http://flare3d.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Gamebryo
    * Officiële website: http://www.gamebryo.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Support door ontwikkelaar

    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- GamePlay 2D/3D v3.0.0
    * Officiële website: http://www.gameplay3d.io/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- jMonkeyEngine 3.0.10
    * Officiële website: http://jmonkeyengine.org/
    * Ondersteuning voor ontwikkeling in 3D
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- ORX 1.7
    * Officiële website: http://orx-project.org/
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- Panda.js
    * Officiële website: http://www.pandajs.net/
    * Ondersteuning voor HTML5
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Panda3D 1.9.2
    * Officiële website: https://www.panda3d.org/

    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Phaser 2.6.1
    * Officiële website: http://phaser.io/
    * Ondersteuning voor HTML5
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Pixi.js v3
    * Officiële website: http://www.pixijs.com/
    * Ondersteuning voor HTML5
    * Ondersteuning voor WebGL
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Spring Engine 103.0
    * Officiële website: https://springrts.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Starling Framework 2.0.1
    * Officiële website: http://gamua.com/starling/
    * Ondersteuning voor ontwikkeling in 3D
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen

- Voldoet niet aan zes *must have* eisen
  - Adventure Game Studio 3.3.5
    - Officiële website: http://www.adventuregamestudio.co.uk/
    - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    - Support door ontwikkelaar
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Allegro 5.2.0
    - Officiële website: http://liballeg.org/
    - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    - Support door ontwikkelaar
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Crafty v. 0.7.1
    - Officiële website: http://craftyjs.com/
    - Ondersteuning voor HTML5
    - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Corona
    - Officiële website: https://coronalabs.com/
    - Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - EasyIJS 0.8.2.
    - Officiële website: http://createjs.com/
    - Ondersteuning voor HTML5
    - Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    - Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Game Closure

    * Officiële website: http://www.gameclosure.com/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Havok
    * Officiële website: http://www.havok.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Impact
    * Officiële website: http://impactjs.com/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Isogenic Game Engine v1.5.5
    * Officiële website: http://www.isogenicengine.com/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Kivy 1.9.1
    * Officiële website: https://kivy.org/
    * Ondersteuning voor ontwikkeling in 3D
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- LimeJS
    * Officiële website: http://www.limejs.com/
    * Ondersteuning voor HTML5

    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- LycheeJS
    * Officiële website: https://lychee.js.org
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Moai 1.7.5
    * Officiële website: http://www.arkham-development.com/
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- OpenClonk 7.0
    * Officiële website: http://www.openclonk.org/
    * Ondersteuning van een objectgeoriënteerde taal zoals C++ of C#
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- OpenSimulator
    * Officiële website: http://opensimulator.org/
    * Ondersteuning voor ontwikkeling in 3D
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- PlayN
    * Officiële website: http://playn.io/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

  - Quintus
    * Officiële website: http://www.html5quintus.com/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Stencyl 3.3
    * Officiële website: http://www.stencyl.com/
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
    * Mogelijkheid om te debuggen
  - Wade
    * Officiële website: http://clockworkchilli.com/
    * Ondersteuning voor HTML5
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Zest3D
    * Officiële website: http://zest3d.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Voldoet niet aan zeven *must have* eisen
  - BuildBox 2.2.0
    * Officiële website: https://www.buildbox.com/
    * Support door ontwikkelaar
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten
  - Löve 0.10.1

    * Officiële website: https://love2d.org/
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- Ren'Py 6.99.10
    * Officiële website: https://www.renpy.org/
    * Maximale kostprijs bedraagt €500 per jaar of éénmalig €1200
    * Heeft gedetailleerde documentatie zoals voorbeeldprojecten en voorbeeldtesten

- SunBurn 2.1.11.F
    * Officiële website: https://www.synapsegaming.com/
    * Ondersteuning voor ontwikkeling in 3D
    * Support door ontwikkelaar

Bijlage B

Unity

B.1 BasicFPScript.cs

```
using UnityEngine;
using System.Collections;

public class BasicFPScript : MonoBehaviour {
        private const float SPEED = 10.0F;
        // Use this for initialization
        void Start () {
                Cursor.lockState = CursorLockMode.Locked;
        }
        // Update is called once per frame
        void Update () {
                float x = Input.GetAxis("Horizontal") * SPEED;
                float y = Input.GetAxis("Vertical") * SPEED;
                x *= Time.deltaTime;
                y *= Time.deltaTime;
                transform.Translate(x,0,y);
                                if(Input.GetKeyDown("escape"))
                        Cursor.lockState = CursorLockMode.None;
        }
}
```

Codefragment B.1: BasicFPScript.cs

B.2 MouseLookScript.cs

```
using UnityEngine;
using System.Collections;

public class MouseLookScript : MonoBehaviour {

        private Vector2 mouseLook;
        private Vector2 smoothV;
        private const float sensitivity = 5.0f;
        private const float smoothing = 2.0f;
        private GameObject character;

        void Start () {
                character = this.transform.parent.gameObject;
        }

        void Update () {
                var md = new Vector2(Input.GetAxisRaw("Mouse X"),
                        Input.GetAxisRaw("Mouse Y"));
                md = Vector2.Scale(md,
                        new Vector2(sensitivity * smoothing,
                        sensitivity * smoothing));
                smoothV.x = Mathf.Lerp(smoothV.x, md.x, 1f
                        / smoothing);
                smoothV.y = Mathf.Lerp(smoothV.y, md.y, 1f
                        / smoothing);
                mouseLook += smoothV;
                transform.localRotation =
                        Quaternion.AngleAxis(-mouseLook.y,
                        Vector3.right);
                character.transform.localRotation =
                        Quaternion.AngleAxis(mouseLook.x,
                        character.transform.up);
        }
}
```

Codefragment B.2: MouseLookScript.cs

B.3 AIScript.cs

```
using UnityEngine;
using System.Collections;

public class MoveTo : MonoBehaviour
{
    private const float ATTACKSPEED = 5.0F;
    private const float SPEED = 10.0F;
    private const float ATTACKRANGE = 10.0F;
    private const float MEELEERANGE = 2.0F;
    private const float STOPRANGE = 1.135F;
    public Transform target;
    private NavMeshAgent agent;

    void Start()
    {
        agent = GetComponent<NavMeshAgent>();
    }

    void Update()
    {
        agent.SetDestination(target.position);
        if (Vector3.Distance(transform.position,
                target.position) > ATTACKRANGE)
        {
            agent.speed = SPEED;
            GetComponent<Renderer>().material.color
                        = new Color(0, 0, 255);
        }
        else if (Vector3.Distance(transform.position,
                target.position) < ATTACKRANGE
                && Vector3.Distance(transform.position,
                target.position) > MEELEERANGE)
        {
            agent.speed = ATTACKSPEED;
            GetComponent<Renderer>().material.color
                        = new Color(255, 0, 0);
        }
        else if (Vector3.Distance(transform.position,
```

```
                target.position) < MEELEERANGE
                && Vector3.Distance(transform.position,
                target.position) > STOPRANGE)
        {
            agent.speed = ATTACKSPEED;
            GetComponent<Renderer>().material.color
                        = new Color(0, 255, 0);
        }
        else
        {
            agent.speed = SPEED;
            GetComponent<Renderer>().material.color
                        = new Color(0, 255, 0);
        }
    }
}
```

Codefragment B.3: AIScript.cs

Bijlage C

Unreal Engine

C.1 FPPlayer.cpp

```
#include "UnrealProject.h"
#include "FPPlayer.h"

AFPPlayer::AFPPlayer()
{
        PrimaryActorTick.bCanEverTick = true;
}

void AFPPlayer::BeginPlay()
{
        Super::BeginPlay();
}

void AFPPlayer::Tick( float DeltaTime )
{
        Super::Tick( DeltaTime );
}

void AFPPlayer::SetupPlayerInputComponent(class UInputComponent*
        InputComponent)
{
        InputComponent->BindAxis("MoveForward", this,
                &AFPPlayer::MoveForward);
        InputComponent->BindAxis("MoveRight", this,
                &AFPPlayer::MoveRight);
```

```
        InputComponent->BindAxis("Turn", this,
                &AFPPlayer::AddControllerYawInput);
        InputComponent->BindAxis("LookUp", this,
                &AFPPlayer::AddControllerPitchInput);
}
void AFPPlayer::MoveForward(float Value)
{
        if ((Controller != NULL) && (Value != 0.0f))
        {
                FRotator Rotation =
                        Controller->GetControlRotation();
                if (GetCharacterMovement()->IsMovingOnGround()
                        || GetCharacterMovement()->IsFalling())
                {
                        Rotation.Pitch = 0.0f;
                }
                const FVector Direction =
                        FRotationMatrix(Rotation).GetScaledAxis(
                                EAxis::X);
                AddMovementInput(Direction, Value);
        }
}

void AFPPlayer::MoveRight(float Value)
{
        if ((Controller != NULL) && (Value != 0.0f))
        {
                const FRotator Rotation =
                        Controller->GetControlRotation();
                const FVector Direction =
                        FRotationMatrix(Rotation).GetScaledAxis(
                                EAxis::Y);
                AddMovementInput(Direction, Value);
        }
}
```

Codefragment C.4: FPPlayer.cpp

C.2 FPPlayer.h

```
#pragma once

#include "GameFramework/Character.h"
#include "FPPlayer.generated.h"

UCLASS()
class UNREALPROJECT_API AFPPlayer : public ACharacter
{
        GENERATED_BODY()

public:
        AFPPlayer();
        virtual void BeginPlay() override;
        virtual void Tick( float DeltaSeconds ) override;
        virtual void SetupPlayerInputComponent(class
                UInputComponent* InputComponent) override;
        UFUNCTION()
                void MoveForward(float Val);
        UFUNCTION()
                void MoveRight(float Val);
};
```

Codefragment C.5: FPPlayer.h

C.3 AIGameMode.cpp

```
#include "UnrealProject.h"
#include "AIGameMode.h"
#include "Engine.h"
#include "FPPlayer.h"

AAIGameMode::AAIGameMode(const FObjectInitializer&
        ObjectInitializer)
    : Super(ObjectInitializer)
{
        DefaultPawnClass = AFPPlayer::StaticClass();
}
```

```
void AAIGameMode::StartPlay()
{
	Super::StartPlay();
	StartMatch();
}
```

Codefragment C.6: AIGameMode.cpp

C.4 AIGameMode.h

```
#pragma once

#include "GameFramework/GameMode.h"
#include "AIGameMode.generated.h"

/**
 *
 */
UCLASS()
class UNREALPROJECT_API AAIGameMode : public AGameMode
{
	GENERATED_BODY()
	virtual void StartPlay() override;
	AAIGameMode(const FObjectInitializer& ObjectInitializer);
};
```

Codefragment C.7: AIGameMode.h