Rainbow Racers - de Rainbow Road naar onze app

Leroy Speijker	Yarron Maas	Roelof Hoeksema	Chettawat Mekwan
s4553934	s4571665	s4341228	s4606078

17 juni 2016

Voorwoord

Tijdens onze cursus Research & Development hebben wij een app gemaakt. Onze app heet Rainbow Racers en dit verslag beschrijf het proces wat we hebben doorlopen om onze app te bouwen. Dit document is bedoeld om een redelijk beeld te geven van wat voor applicatie het is, en hoe we het als groepje gebouwd hebben.

Voor de opbouw van dit document, hebben wij gekozen om gewoon het voorbeeld op internet te volgen. Dit wil zeggen dat wij vier secties hebben, namelijk:

- Voorwoord
- Beschrijving
- Ontwerp
- Reflectie

Twee van deze punten worden nog verder uitgewerkt in verschillende subsecties.
Voor de beschrijving is dit:

- Inleiding
- Productverantwoording
- Specificaties

Voor ontwerp is dit:

- Globaal ontwerp
- Detailontwerp
- Ontwerpverantwoording

Beschrijving

Inleiding

Onze app is een race app, zoals jullie al konden verwachten met onze vrij onthullende naam "Rainbow Racers". We hebben deze app zo genoemd omdat racen en regenbogen een grote rol spelen in dit spelletje. Het idee is dat je zo ver mogelijk moet komen met je regenboog buggy, om zo veel mogelijk punten op te halen. Dit kan natuurlijk alleen maar moeilijker worden naar mate de snelheid steeds meer oploopt als je een grotere afstand afgelegd hebt. Er zijn ook nog obstakels die je moet ontwijken, en dit is zeker niet makkelijk. Het idee is heel simpel, het spel zelf leuk, maar ook verassend moeilijk.

Productverantwoording

Onze applicatie is een spelletje voor een redelijk brede gebruikersgroep. Je zou er jonge kinderen mee kunnen laten spelen, maar ook ouderen die eventjes een beetje vrije tijd hebben. Om deze redenen denken wij dat onze app een aardig geslaagd project is en dat het daarom ook zeker de moeite waard was om de app te bouwen. Het is een combinatie tussen een oude speelgoed racebaan, en de spellen van tegenwoordig waarin de high score eigenlijk alles is. Neem bijvoorbeeld Flappy Bird. Dit spelletje had een heel simpele opzet, met een daverend succes. Wij zien dat ons dat ook nog wel eens kan overkomen, omdat het best simpel op te pakken is en je daarna heel goed je best kan doen om een zo hoog mogelijke score neer te zetten.

Naar ons idee zijn wij op dit gebied de eerste die dat aspect gebruikt hebben om een 3D racespel te fabriceren. We hebben een tijdje rondgekeken maar niets was daadwerkelijk vergelijkbaar met ons spel. Af en toe kwamen we wel 2D racers tegen die iets vergelijkbaars deden maar het was dan net weer anders. Ze misten jammer genoeg allemaal iets, en waarom we dus besloten hebben ons idee realiteit te maken.

Natuurlijk zijn er wel apps die erop lijken, zoals Temple Run. Deze is echter toch wel anders dan ons spel. Daarin kan je alleen maar de hoek om, en springen of duiken. Je kan op de baan waar je loopt niet kiezen of je aan de linker kant van je weggetje, of aan de rechter kant van je weggetje wilt rennen. Dat kan bijvoorbeeld bij ons racespel wel. Daardoor vergt ons spel een grotere inzet, en natuurlijk een grotere precisie van onze gebruikers. Neem bijvoorbeeld onze schansjes, die zijn best lastig te raken. Daardoor denken we dat het uitdagend blijft, en toch wel verschilt van andere spellen in deze hoek.

Specificaties

Alhoewel Rainbow Racers een relatief simpele applicatie lijkt, zijn er redelijk zijn er hoge eisen gesteld om het spel goed bespeelbaar te laten zijn. Dit vloeide op een natuurlijke wijze voort uit ons idee om een spel te produceren dat simpel is, maar tegelijkertijd toch aantrekkelijk en verslavend om te spelen. Als men naar de eigenschappen van de applicatie kijkt, zijn er zowel functionele als niet-functionele eigenschappen te benoemen.

Laten we allereerst naar de functionele eigenschappen kijken. Uiteraard hebben we een logo gemaakt om het spel herkenbaar te maken. Dit is een vierkant bestaande uit twee letters R, de afkorting van Rainbow Racers, die gevuld zijn met een gekleurde sterrenhemel. Aan de linker onderzijde en de rechter bovenzijde zijn nog twee balken toegevoegd die tevens ook gevuld zijn met een gekleurde sterrenhemel. De balken zijn zowel toegevoegd om een abstracte weg af te bakenen, ter referentie naar ons racespel, als om het logo een bepaalde elegantie te geven, waardoor men eerder geneigd is om het spel op te starten.

Eenmaal op dit logo gedrukt, belandt men in de applicatie. Ons product bestaat uit twee verschillende schermen. Het eerste scherm dat wordt aangetroffen is het hoofdmenu. Op de achtergrond ziet men de ruimte, het oogt uitdagend, speels en mysterieus. In de rechter bovenhoek is een artwork toegevoegd met de naam van het spel. Aan de linkerzijde van het hoofdmenu treft men twee knoppen aan, "Play!" en "Instructions".

Als men op de "Instructions" knop drukt, verschijnen er instructies aan de rechterzijde van het scherm. Hierin wordt de intutieve touchbestuuring uitgelegd, de mogelijktijd om de camera te laten schakelen tussen third-person (standaard, birds view) en first-person (vanuit de auto). De instructies leggen ook het doel van het spel uit, namelijk blijf zo lang mogelijk in leven. Als instructie laatste een hint die erop wijst dat schansen het best in een recht pad benaderd kunnen worden om kans op overleven te vergroten..

Het drukken op de "Play!" knop zorgt ervoor dat het spel start. Het spel bestaat uit vier hoofdcomponenten: de muziek, een regenboogatmosfeer, een weg en een buggy. De muziek creert sfeer. De regenboogatmosfeer functioneert als achtergrond. De weg geeft een ondergrond om op te rijden en functioneert als steunpunt voor alle obstakels. De buggy is wat men bestuurt en een zo lang mogelijke afstand moet laten rijden om een highscore neer te zetten.

De muziek is geproduceerd door Kevin Macleod en zonder auteursrecht. In het menuscherm in het begin een rustig, mysterieus en melodisch deuntje maar na een tijdje wordt het erg opwekkend en avontuurlijk. In het spel zelf is er tevens ook een melodisch deuntje op de achtergrond. Deze klinkt wederom mysterieus, maar wekt bovenal spanning op. Het geluidsvolume is in te stellen met de mediavolumeregelaar op de telefoon.

De regenboogatmosfeer is een grote ronddraaiende bol met een vloeiende kleurengradint. Deze bol levert het grootste aandeel aan regenboogkleuren in het spel en bepaalt daarmee ook de dominante speelse en vrolijke sfeer.

De weg is een recht stuk dat zich oneindig lang uitbreid zolang men leeft in het spel. De weg is bedekt met een sterrenhemel. Hier is expliciet voor gekozen als metafoor voor de oneindigheid van het spel. Deze oneindige weg wordt aan weerszijden afgebakend door een cilindervormige stroom van lava die meegroeit met de weg. Op deze weg worden obstakels gegenereerd, schansen en muren. Hierbij is rekening gehouden met de afstand tussen combinaties van obstakels, zodanig dat het altijd mogelijk is om er voorbij te gaan en nog verder te kunnen racen (alhoewel het soms heel moeilijk kan zijn!).

De laatste functionele eigenschap is de buggy. Deze bestuurt men en tracht men zolang mogelijk te laten rijden. De buggy stuurt en beweegt simpel, maar speels. Met dank aan Unreal Engine 4 zijn de physics van de buggy zoals men kan verwachten. Sturen werkt soepel en verlaagt de snelheid van de buggy, als gevolg van de veranderingen van de krachten op de buggy. Bij het neerkomen na een schans, is er tijd nodig voordat er weer tractie onstaat tussen de banden en de weg, wat voor extra uitdaging zorgt. Tenslotte kan men ook nog wisselen naar een first-person view, dus je kijkt vanuit de auto. Deze view zorgt voor een nog intensere gameplay. Hierbij worden ook nog de snelheid en de versnelling getoond. Hierdoor wordt het spel wel flink moeilijker - muren zijn moeilijker te onderscheiden van elkaar, maar dit is extra uitdaging (en daarom is de standaard ook third person).

Uiteraard zijn er ook niet-functionele eigenschappen. Het spel is redelijk veeleisend om een soepele gameplay te kunnen garanderen. Allereerst is er een goede hoeveelheid ruimte nodig in het geheugen om het spel te installeren want de APK van het spel is maarliefst 250MB groot. Daarnaast moet de telefoon een redelijk recente versie van Android hebben. Als dit is gelukt, zal het spel pas goed tot zijn recht komen als het mobiele toestel enigszins krachtig is, zodat zowel de gameplay als animaties soepel zijn.

Hieruit kunnen we ook nog een eigenschap afleiden - we willen minimaal 30 frames per second kunnen renderen, zodat het spel altijd soepel loopt. Voor de gameplay is uiteraard ook een touchscreen vereist. Tenslotte is er een (media)volumeregelaar nodig om het volume aan te passen van de muziek. Deze zit op alle moderne Android telefoons, dus daarom hebben we geen in-game volume regelaar geimplementeerd.

Ontwerp

Globaal Ontwerp

Wij hebben voor onze app Unreal Engine 4 gebruikt. Hierdoor is het vrij lastig om alle modules van onze app uit te leggen, want hiervoor is eigenlijk kennis van Unreal Engine 4 nodig. Om deze reden zal ik niet (diep) op de technische aspecten van onze componenten, maar ze generaal uitleggen door te zeggen uit welke componenten ons spel bestaat en wat deze doen. Ook zijn er veel kleine dingetjes waardoor het spel draait, ik leg hier alleen een aantal hoofdcomponenten uit. Ons spel heeft een simpele basisfunctionaliteit: er moet een raceauto zijn, er moet een weg zijn, er moeten obstakels zijn, er moeten regenboogelementen in zitten, er moet score bijgehouden worden en er moeten duidelijke menus zijn.

De rol van de auto is ons spel is simpel: de auto moet uit zichzelf gas geven, dus het is alleen mogelijk naar links of naar rechts te sturen. Er zijn ook geen remmen.

De rol van de weg is ook simpel: deze moet solide ondergrond bieden voor de auto om op te rijden, want als je nergens grip op hebt, kom je niet ver vooruit met draaiende wielen (want de physics zijn realistisch).

De obstakels zijn er om het spel ook echt een spel te maken, want zomaar in een rechte lijn vooruit rijden is natuurlijk niet erg leuk. Omdat het spel ook iedere keer weer een beetje anders moet voelen zijn deze obstakels willekeurig geplaatst uit een set voorgedefinierde obstakels (er kan wel enige variatie zitten in de obstakels zelf).

Het regenboogelement van het spel moet ook aanwezig zijn, want de app heet nou eenmaal Rainbow Racers. Hiervoor hebben we gezorgd door de achtergrond alle kleuren van de regenboog te geven, die snel en soepel afwisselt.

De score bijhouden is ook belangrijk, want anders weet je niet hoe goed je het gedaan hebt en is het niet een echt spel.

Ten slotte hebben we nog de menus, die er zijn zodat je niet direct het spel ingegooid wordt als je de app opstart. Ook kan dit menu instructies geven, en een Game Over scherm bieden waar de score te zien is.

Detailontwerp

De auto is gebouwd in blueprints. Deze was al beschikbaar in UE4, maar we hebben de auto voor ons doeleinde wat moeten aanpassen. Deze aanpassingen zorgen ervoor dat de auto niet kan remmen, constant gas geeft en dat de touch besturing op het scherm van een telefoon werkt.

De weg en de obstakels worden gegenereerd door een stuk C++ code wat alle random generation afhandelt, de zogenaamde RoadGenerator. Deze geef je als parameters de beschikbare objecten om obstakels mee te bouwen en de weg & randen. Uit deze bouwblokjes kan de RoadGenerator een uitdagende weg bouwen die toch altijd mogelijk is. De RoadGenerator zelf handelt niet het GameOver event af - dit doen de individuele bouwblokjes. Deze hebben allemaal OnCollision events, waardoor ze bij een botsing een bepaalde functie uitvoeren (bij ons dus het GameOver scherm).

De regenboogachtergrond is gebouwd in blueprints. Er was een soort van 'lucht' beschikbaar in UE4, maar deze voldeed niet aan onze doelen, dus we moesten deze aanpassen. Om ons gewenste regenboog element te krijgen hebben we eerst een kleine blueprint gemaakt die met een slim trucje een Material maakt wat eruitziet als regenboog. Deze Material hebben we vervolgens op de lucht van UE4 toegepast, maar dit was natuurlijk statisch, dus je zag alleen maar een deel van de regenboog. Om ervoor te zorgen dat je heel de regenboog ziet, hebben we bij de lucht het 'Tick' event aangezet. Deze zorgt ervoor dat iedere keer dat het spel update, de lucht iets uitvoert. Om de lucht te laten draaien hebben we aan dit 'Tick' event een module toegevoegd om de lucht iedere keer met 1 graad te draaien rond de Z-as. Vervolgens hebben we de lucht ingesteld op 'Dynamisch' in plaats van 'Statisch'. Toen dit allemaal gebeurd was hadden we dus een mooie regenboog als achtergrond die constant van kleur wisselde. Het gewenste effect was dus bereikt. We hebben ook nog zelf een flinke tijd lucht proberen te bouwen, maar dit bleek zeer moeilijk - toen realiseerden we ons dat het misschien makkelijker was om de voorgebouwde lucht aan te passen, en dit lukte na wat sleutelen.

De score is uiteindelijk afgehandeld in de zogenaamde "Level Blueprint". Dit is de code die runt als een level geladen wordt en als een level in gebruik is. Hiervoor hebben we eenzelfde 'Tick' event aan het level toegevoegd dat telkens 1 bij de score doet. De score was dus geregeld, maar we liepen vervolgens tegen een probleem op - hoe lieten we dit aan de gebruiker zien tijdens en na het spel? Dit hebben we uiteindelijk opgelost door deze Score een publieke variabele te maken en een paar trucjes uit te halen om het Level object in de "HUD"(wat er op het scherm te zien is) blueprint te krijgen en vervolgens de Score eruit te halen, en toen hetzelfde trucje voor het menu gedaan. Toen dit klaar was hadden we dus ons scoresysteem en konden we er iets mee!

De menus hadden we ook nog enige problemen mee. Deze zijn ook ontworpen in blueprints, maar we wilden ook muziek tijdens onze menus en levels. Om een menu te gebruiken zonder dat het spel op de achtergrond liep, moesten we het spel op pauze zetten. Dit pauzeerde helaas ook de muziek! Na lang zoeken hebben we hier ook een trucje op gevonden - we konden de "Time Scale"(hoe snel de tijd voorbij gaat) van het spel op 0 zetten, waardoor audio gewoon af kon spelen, maar het spel wel effectief op pauze stond. Hierdoor liepen we weer tegen een ander probleem op - de score liep op in het menu, want de time scale op 0 zorgt er niet voor dat de 'Tick' events niet meer gebeuren. We moesten dus ook weer even sleutelen aan het scoresysteem zodat deze ook rekening hield met onze vorm van "pauze". Toen dit geregeld was hebben we de menus ontworpen en uitgetest, en gelukkig was, na al dat sleutelen, ons menu af. We hadden een mooi menu met goede graphics, duidelijk instructies, duidelijke knoppen en muziek.

Ontwerpverantwoording

Een van de beslissingen die we moesten maken was over de besturing van de auto. Wij hebben ervoor gekozen om alleen maar de mogelijkheid te geven tot links en rechts sturen, en dus niet remmen/gas geven. De auto geeft zelf gas en remt nooit. Dit wilde we doen om ons spel moeilijker, maar ook uniek te maken. We hebben zelf ook alle mogelijkheden getest, en dit bleek voor ons bij verreweg de leukste optie van spelen. Ook werkt dit goed uit voor de usability: de besturing is hierdoor heel simpel, tik links op het scherm om links te sturen en rechts op het scherm om rechts te sturen! Dit lost ook direct het probleem op van kleine knopjes die je moeilijk kunt aantikken aangezien je het hele scherm kan gebruiken.

Een andere beslissing die we moeste maken was welke obstakels we in ons spel gingen doen. Er waren veel ideeen, maar uiteindelijk hebben we ervoor gekozen om het simpel te houden, om de simpliciteit van ons idee te behouden. Dit resulteerde in 2 verschillende hoofdobstakels die zelf nog heel verschillend kunnen zijn: een muur en een schans. De muur kan varieren doordat er of n grote muur is of een aantal smalle muren met een gat ertussen. De schans kan varieren door de plek waarop de schans over het muurtje springt, deze afwijking wordt willekeurig bepaald door de RoadGenerator. Door de keuze van simpele obstakels is het spel toch nog vrij moeilijk, maar wel goed te leren en behoudt het zijn simpliciteit. Ook kun je met simpele obstakels garanderen dat het ALTIJD mogelijk is om te blijven leven als je goed genoeg bent, zodat de speler zich niet bedrogen voelt als hij afgaat. Nu kan de speler van zijn fouten leren zodat hij volgende keer een nog hogere score kan halen!

Reflectie

Over het algemeen zijn we zeer tevreden over onze applicatie. We waren erg blij dat we de physics, zoals zwaartekracht, niet hoefden te programmeren, en mede daardoor konden we het ons toch wel veroorloven om er een 3D spel van te maken. Het ontwerpen van de graphics viel eerst wel wat tegen, omdat het dus 3D is maar toen we dat eenmaal een beetje onder de knie hadden ging dat verder prima. Uiteindelijk zijn we ook zeer tevreden over het uiterlijk van het spel. De simpliciteit geeft een gevoel van overzicht. Hierdoor wordt men niet te veel afgeleid door de graphics en kan er gefocust worden op het besturen van de buggy. Tegelijkertijd wordt er door de regenboogkleuren en lava een prettige maar toch uitdagende sfeer gecreerd.

Daarnaast zijn we zeer tevreden over de besturing van het spel. Het is intutief door het feit dat er aan de linker- en rechterzijde van het scherm gedrukt moet worden om de buggy te sturen. Door de uitschakeling van de registratie van meerdere aanrakingen, hebben we ook een uitdaging in de besturing toe kunnen voegen. We waren eerst bang dat onze auto te snel zou accelereren, of dat de besturing te responsief was maar dat was allemaal wel goed gelukt, door middel van het aanpassen van de specificaties in de blueprint van de buggy.

Een grote uitdaging was het feit dat een groot deel van het spel 'randomly generated' zou moeten worden, namelijk zowel de weg als de obstakels. Het is helaas niet gelukt om de weg willekeurig te genereren. Daarover zijn we wel enigszins ontevreden. Wel is dit gelukt met de obstakels. Hier zijn we erg trots op, aangezien het erg goed is gelukt om de 'randomness' van obstakels te implementeren in het spel met een stuk C++ code. Hierbij is ook meteen rekening gehouden met de haalbaarheid. Naar onze mening heeft dit de grootste bijdrage geleverd aan de verslavingsfactor van het spel.

Ook over het ontbreken van powerups, coins en online multiplayer zijn we teleurgesteld. Wellicht waren we te ambitieus. Alhoewel we ook hebben ingezien dat het pittig zou zijn om dit te verwezenlijken. Al met al vinden we het zonde dat dat ons niet gelukt is.

Uiteindelijk hebben we allemaal erg goed leren werken met Unreal Engine 4, en hebben we onze kennis over C++ weer even wat opgefrist. Ook de samenwerking is prima verlopen. Hoewel het veel tijd heeft gekost om de applicatie in elkaar te zetten en er veel frustraties zijn opgewekt, hebben we allemaal, stuk voor stuk, genoten van het proces. We kijken daarom ook allemaal uit naar een volgende projectmatige cursus.