

14

**Technisch Verslag**

**- Shoot The Bully**

**Arnout Reitsma**

**Ilias Ameziane**

**Tim Hasselaar**

**Hylco Uding**

**Daniël van den Berg**

**Inhoudsopgave**

[Opdrachtomschrijving - 3 -](#_Toc403347169)

[Ons product - 3 -](#_Toc403347170)

[Inleiding - 4 -](#_Toc403347171)

[Ontwikkelomgeving - 5 -](#_Toc403347172)

[1. Visual Studio 2013 - 5 -](#_Toc403347173)

[2. SFML 2.1 - 5 -](#_Toc403347174)

[3. Github - 5 -](#_Toc403347175)

[4. Google Drive - 5 -](#_Toc403347176)

[5. Microsoft Office - 5 -](#_Toc403347177)

[6. Paint.net - 5 -](#_Toc403347178)

[7. Software Ideas Modeler - 5 -](#_Toc403347179)

[Shoot The Bully - 6 -](#_Toc403347180)

[1. Regels - 6 -](#_Toc403347181)

[2. Core-mechanics - 6 -](#_Toc403347182)

[3. Wapens - 7 -](#_Toc403347183)

[4. Design van de game - 7 -](#_Toc403347184)

[De map - 8 -](#_Toc403347185)

[Menus - 8 -](#_Toc403347186)

[Gamebesturing - 8 -](#_Toc403347187)

[Geluid - 9 -](#_Toc403347188)

[Code opbouw - 10 -](#_Toc403347189)

[Klassenverantwoordelijkheden - 10 -](#_Toc403347190)

[Gebruikersmodellen - 16 -](#_Toc403347191)

[1. Game sequence - 16 -](#_Toc403347192)

[2. Klassendiagram - 17 -](#_Toc403347193)

[3. Activity diagram - 18 -](#_Toc403347194)

[Testrapporten - 19 -](#_Toc403347195)

[Weapon Upgrades - 19 -](#_Toc403347196)

[8 Direction movement buttons and character speed - 22 -](#_Toc403347197)

[Bullets - 24 -](#_Toc403347198)

[Collision - 26 -](#_Toc403347199)

[Conclusie - 27 -](#_Toc403347200)

[Bronvermelding - 27 -](#_Toc403347201)

# Opdrachtomschrijving

In opdracht van de Hogeschool Utrecht is ons, Scrumbag Games, gevraagd om een 2D game te maken. Om deze applicatie te programmeren moeten wij gebruik maken van de programmeer taal C++, en de SFML 2.1 libraries. De game moet ongeveer 10 minuten speeltijd leveren, en voor het onderwerp van de game zijn we volledig vrij gelaten.

## Ons product

Wij hebben als team gekozen om de game 'Shoot The Bully' te maken. De speler bestuurd in deze game een character in een afgesloten rechthoekige ruimte. Het spel bestaat uit een aantal rondes van twee minuten. Gedurende deze twee minuten zullen er oneindig veel vijanden op de speler af komen. De speler moet met verscheidene wapens de vijanden van zich af zien te houden. Aan het eind van elke ronde van twee minuten komt er een baas van de ronde, welke de speler moet verslaan om verder te komen in het verhaal.

# Inleiding

In dit technische verslag worden alle aspecten van onze game gedocumenteerd. We zullen beginnen met welke programma's we hebben gebruikt voor het ontwikkelen van de game. Daar na gaan we alle eigenschappen van de game zelf uitlichten. Hier bij zullen we het bijvoorbeeld ook hebben over alle items uit de game, de opbouw van de map, en de besturing. We zullen ook nog de use-cases, het klassendiagram en het flow-diagram bespreken. Verder komen nog alle testrapporten en onze conclusie.

# Ontwikkelomgeving

Hier zullen we alle programma's uitlichten die we hebben gebruikt bij het ontwikkelen van onze game. Hier zal vooral worden besproken waar het programma vandaan komt, en waarvoor het primair is gebruikt.

## 1. Visual Studio 2013

Visual Studio is een programmeer omgeving voor C++. Via onze school hebben wij een gratis versie kunnen bemachtigen. In dit programma is alle code van onze game geschreven.

## 2. SFML 2.1

De SFML 2.1 libraries zijn open-source, en dus gratis, libraries. Deze libraries kunnen worden gecombineerd met C++ in Visual Studio om op een simpele en effectieve manier het grafische deel van de game te programmeren.

## 3. Github

Github is een service waarmee we online al onze code kunnen delen. Door middel van de online synchronisatie techniek van Github kunnen we makkelijk op meerdere computers aan dezelfde code werken. Deze manier is veel handiger dan bijvoorbeeld elke wijziging via de mail door te sturen naar elkaar.

## 4. Google Drive

Net als Github is Google Drive een manier om online bestanden te delen. We gebruiken Google Drive vooral om onze documentatie te delen. Google Drive is hier wat handiger voor omdat meerdere gebruikers op hetzelfde moment aan hetzelfde document kunnen werken.

## 5. Microsoft Office

Word en Excel zijn de twee voornaamste programma's die we gebruiken uit het microsoft office pakket. Word wordt bijvoorbeeld gebruikt om onze documenten als het Game Design te schrijven. Excel is bijvoorbeeld gebruikt om al onze items en attributen van deze items op een overzichtelijke manier op te slaan.

## 6. Paint.net

Paint.net is in principe een uitgebreidere versie van het alom bekende paint. Het voegt verschillende functies toe die handig zijn bij het bewerken van afbeeldingen. Het is dan wel niet net zo uitgebreid als bijvoorbeeld photoshop, maar het is ideaal voor onze doeleinden en bovendien een gratis applicatie

## 7. Software Ideas Modeler

Dit programma is gebruikt om bijvoorbeeld onze use-cases en klassendiagrammen te maken. Het is een gratis programma gespecialiseerd in deze taken.

# Shoot The Bully

In dit hoofdstuk worden alle eigenschappen van de game belicht. We zullen beginnen met de verhaallijn, en de core-mechanics en regels van de game. We zullen ook alle enemies, bullies en powers uit de game beschrijven, samen met alle bijbehorende attributen.

## 1. Regels

De regels van het spel houden in wat de speler wel en/of niet mag doen. De regels die wij voor onze game hebben vastgesteld zijn:

* De speler mag maar 3 wapens bij zich hebben, een (ge-upgrade) mes en 2 wapens die de speler in de shop heeft gekocht.
* Als de speler dood gaat moet hij de laatste wave over doen.
* De speler kan niet meer geld uitgeven dan hij heeft.
* De speler kan niet het spel starten zonder dat hij de tutorial heeft gezien.
* De speler kan niet specifieke levels opnieuw doen, alleen weer bij level 1 beginnen.
* De speler kan alleen schieten met wapens als hij daar genoeg ammunitie voor heeft.
* De speler kan niet door muren heen lopen.
* De speler kan alleen vanaf een bank op een tafel klimmen.

## 2. Core-mechanics

Het spel moet zich ook aan een aantal regels houden. Deze regels noemen we de core-mechanics. De core-mechanics van onze game zijn:

* Het spel is een top down shooter. De speler bevind zich in een afgesloten rechthoekige ruimte (een kantine van een middelbare school),
* De speler kan met de muis zijn wapen mikken en met de WASD, of pijltjestoetsen, rondlopen met zijn character.
* De map heeft meerder aparte objecten die vernietigd kunnen worden (denk aan tafels die door midden breken of borden die kapot vallen in scherven).
* Bullies spawnen net buiten het scherm.
* Bullies lopen altijd recht op je af. Als er zich een obstakel bevind tussen de bully en de speler, dan loopt de bully een aantal pixels opzij. Net zo lang tot dat er zich geen obstakel meer tussen hem en de speler zit.
* Elke ronde duurt 2 minuten. In deze twee minuten spawnen oneindig veel bullies, met een maximum aantal tegelijk op het scherm.
* Het maximum aantal bullies tegelijk op het scherm word bepaald door de ronde waar in de speler zich bevind, en de moeilijkheid waar op de speler het spel speelt.
* Elke ronde komen er sterkere bullies (dit houdt in: meer levenspunten en een hogere loopsnelheid).
* Sommige bullies laten een random gekozen power vallen als ze dood gaan.
* Als een bully wordt geraakt door een kogel neemt zijn hp af.
* Een wapen mag alleen worden herladen als het magazijn van het wapen niet meer vol zit.
* Aan het einde van een wave kan je in de shop nieuwe wapens kopen of bestaande wapens upgraden.
* Als je een bully neer schiet (of slaat) krijg je jouw gestolen lunchgeld terug ( willekeurig tussen de 50 en 100 euro).
* De speler kan niet door obstakels heen lopen.
* De speler kan op lage obstakels (zitbanken) klimmen.
* De speler kan via een bankje op een tafel klimmen.

## 3. Wapens

Voor de wapens hebben wij gekozen uit een selectie melee wapens en geweren. We hebben ervoor gekozen 6 verschillende wapens in de game te stoppen. Elk wapen heeft zijn eigen voor- en nadelen.

Voor de melee wapens geldt bijvoorbeeld dat ze oneindige ammunitie hebben en veel damage doen, maar ze kunnen alleen gebruikt worden op een hele korte afstand. Tussen de twee melee wapens zit ook nog het verschil dat de knife de kortste afstand heeft, maar sneller attackt. Het zwaard heeft een iets langere range maar slaat langzamer.

Voor de geweren is er wat meer diversiteit. Je hebt bijvoorbeeld het pistool, welke weinig damage doet maar oneindige ammunitie heeft. Als andere wapen in dit slot kun je kiezen voor de assault rifle, welke een all round wapen is. Hij heeft een hoge fire rate, redelijke damage en veel kogels. Als de speler comfortabel is met het spelen met een wapen zonder oneindig ammunitie is de assault rifle een heel sterk wapen.

Voor het derde wapen slot kan de speler kiezen uit twee wapens met hoge damage. Deze zijn de shotgun en de sniper. De sniper zoomt het beeld van de speler uit en heeft de hoogste damage, en de shotgun heeft meer kogels per magazijn. Dus ook al zijn de wapens beide geschikt om veel damage te doen in weinig schoten, toch is er een redelijke afweging tussen de twee.

## 4. Design van de game

Wij hebben gekozen om de game weer te geven op een scherm van 640 bij 480 pixels. De standaard grootte van game objecten zijn 32 bij 32 pixels. Deze grootte geld bijvoorbeeld voor het character van de speler en vijandelijke bullies.

### De map

De map van het spel zal de kantine van een middelbare school zijn. Het is een rechthoekige kamer die 4 maal zo groot is als het scherm waar op de speler het spel speelt, dus twee maal de hoogte en twee maal de breedte.

Elke ronde word op dezelfde map gespeeld. Toch wilde we dat er enige variatie zat in het spel met betrekking tot de map. Daarom hebben we besloten elke ronde de map kleine beetjes te veranderen.

De eerste ronde zal er een rode gloed over de map hangen die de woede van Marcel weergeeft. Alle objecten zoals bijvoorbeeld de tafels zullen in het eerste level ook heel erg scheef staan. Hoe verder de speler komt met de rondes, hoe rustiger de map word. Dit houdt in dat de rode gloed weg trekt en dat alle objecten netter worden neergezet. In de laatste ronde zullen dus alle tafels en bankjes volledig recht staan.

De map word omringd aan alle vier de kanten door muren, en er zal in elke hoek een pilaar staan. De zaal zal gevuld worden met tafels en bankjes en dergelijke objecten die je in een kantine kunt vinden. Het midden van de map zal vrij worden gehouden, zodat de speler daar elke ronde kan beginnen.

### Menus

Als de speler zich in een menu van de game bevind zal de achtergrond bestaan uit de map, waar de camera willekeurig over heen beweegt. Ook zullen er een aantal knoppen worden weer gegeven afhankelijk van in welk menu de speler zich bevind.

De menu's zullen bestaan uit:

* Het main menu; waar de speler naar de opties kan, en het spel kan starten..
* Het opties menu; waar de speler bijvoorbeeld het volume kan veranderen, of het bloed aan/uit kan zetten.
* Het in-game menu; in het in-game menu word het spel gepauzeerd, en kan de speler het spel opnieuw starten, of terug naar het main menu.

### Gamebesturing

Voor de besturing van onze game hebben we gekozen voor een vrij standaard schema voor top-down shooters. De speler kan zich in het spel rond bewegen met beide de W, A, S en D knoppen, of gebruik maken van de pijltjes toetsen.

De speler kan zijn wapen mikken met de muis, en het wapen gebruiken door met de muis te klikken. Ook kan de speler zich door de menu's navigeren met behulp van de muis.

Ook kan de speler tussen zijn gekozen wapens kiezen met de knoppen 1, 2 en 3. Voor het pauzeren van het spel kan de speler de escape knop, of de letter 'P' gebruiken.

### Geluid

Voor onze game gebruiken we achtergrond muziek, welke vanaf het opstarten vanaf de game zal draaien. Ook zullen we schietgeluiden gebruiken voor alle geweren, en melee wapens. Ook word er een geluidje geimplementeerd voor het moment dat de speler gewond raakt.

# Code opbouw

We hebben geprobeerd onze code zo overzichtelijk mogelijk te houden, en zullen daarnaast ook nog in dit hoofdstuk een aantal keuzes toelichten. Ook zullen we documenteren wat de verantwoordelijkheden van bepaalde klassen zijn.

## Klassenverantwoordelijkheden

Voor het opstarten van de game zijn vooral de volgende klassen belangrijk:

* Main
  + In de Main worden de instances van de LevelController, SoundController en Hudcontroller aangemaakt. Ook wordt de instance van de GameController aangemaakt, en wordt door het aanroepen van een functie hierin de game gestart. Op deze manier is het gehele spel in één keer geladen.
* LevelController
  + In de LevelController worden alle levels geladen en worden alle GameObjecten bijgehouden. Dezen worden vanaf hier dus ook geupdated, gemoved en gedrawt. Door dit bij elkaar te houden kunnen er gemakkelijk objecten en features worden toegevoegd.
* Logo
  + Logo wordt gebruikt om een SplashScreen te laden en deze voor 5 seconde op het scherm te laten zien. Als deze 5 seconde voorbij zijn wordt het hoofd menu gestart.
* GameController
  + In de GameController wordt het spel bestuurd. Dit betekend dus dat hierin het gameWindow en de FPS bijgehouden en bestuurd. Er is gekozen om deze controller aan te maken omdat deze een laag vormt tussen het spel (de gameObjecten) en de desktop.

Daarna, in het main menu, komen de volgende klassen aan bod:

* Checkbox
  + ToggleGore
  + ToggleMusic
  + ToggleSound
  + De Checkbox klasse is een GameObject voor op de HUD waarop geklikt kan worden om een setting aan of uit te zetten. Deze Checkbox laad zijn waarde uit en slaat zijn waarde op via de SettingsController.
* Clickable
  + De clickable klasse is de superklasse voor alle GameObjecten waarop geklikt kan worden. Door een virtual function click() te overriden kan een child acties uitvoeren als er op de clickable geklikt wordt. De clickable handelt het berekenen van de positie van de muis en de input van de muisknoppen af.
* MainMenu
  + MainMenu is een button waarop geklikt kan worden om naar het Main Menu level te gaan.
* Options
  + Options is een button waarop geklikt kan worden om naar het options level te gaan.
* Tutorial
  + Tutorial is een button waarop geklikt kan worden om naar het tutorial level te gaan.
* TutorialScreen
  + In het tutorial screen word door middel van tekst het verhaal van onze game weergegeven. Ook word de besturen van de game aan de speler uitgelegd.
* TextureManager
  + De textureManager is de klasse die er voor zorgt dat alle textures ingeladen worden, en dat ze ook maar één keer ingeladen moeten worden. Hij slaat alle textures op in een map om deze later, indien nodig, aan gameObjecten door te geven.
* SoundSlider
  + SoundSlider wordt gebruikt om het niveau van de in-game geluiden aan te passen.
* MusicSlider
  + MusicSlider wordt gebruikt om het niveau van het achtergrond geluid aan te passen.
* SettingsController
  + De settingsController is de controller die er voor zorgt dat alle settings ingeladen en opgeslagen kunnen worden. Deze controller zit dus tussen de gameObjecten en de savefile. Er is voor gekozen om de settings op te slaan in een plaatje, omdat dit, in tegenoverstelling tot een tekstfile, geen zoekfuncties vereist. In plaats daarvan wordt een bepaalde setting met een bepaalde pixel geassociëerd.
* Play
  + Play is een button waarop geklikt kan worden om het spel te beginnen.

Wanneer de game op start beginnen de volgende klassen te werken:

* Animation
  + De animation klasse zorgt ervoor dat animaties in onze game mogelijk worden. Hij laad alle benodigde plaatjes voor een bepaalde animatie af, en laat deze op volgorde afspelen als het hem word gevraagd.
* Bench
  + De bench klasse is voor alle banken in ons spel. Ze doen eigenlijk niet heel veel bijzonders naast het staan in de map. De speler kan de banken gebruiken om op tafels te klimmen.
* Collision
  + De collision klasse kan worden gebruikt om te kijken of bepaalde game objecten elkaar raken. Hij kan tussen bijvoorbeeld twee objecten, of een object en een lijn uitrekenen wat de twee dichtst bij elkaar liggende punten zijn, en of deze objecten elkaar raken.
  + Dit word bijvoorbeel gebruikt om te kijken of een speler beide een bankje en een tafel raakt als hij een tafel op wil klimmen.
* Exception
  + De exception klasse word gebruikt om onze eigen exceptions te kunnen gooien in onze console. Dit werd vooral gebruikt bij het playtesten om te kijken waar mogelijk fouten in onze game zaten.
* Factory
  + De factory klasse zorgt ervoor dat alle objecten uit de .level file in de game geladen worden. Dit is een standaard en bekend systeem.
* GameObject
  + De gameObject klasse word eigenlijk gebruikt voor elk object in de game. Elk object, van button tot bully in onze game is een game object. Op deze manier kunnen we alles bijvoorbeeld in dezelfde vector stoppen om makkelijk onze hele game te kunnen updaten en te drawen.
* GameObjectManager
  + De GameObjectManager zorgt ervoor dat er een gameObject opgevraagd kan worden aan de hand van een string. Hiervoor is een aparte klasse aangemaakt om een te lange switch in de Factory te voorkomen.
* HudController
  + De HudController zorgt ervoor dat alle GameObjecten die op de hud getekend moeten worden geupdated, gemoved en gedrawt worden. Dit wordt in een andere klasse dan de LevelController gedaan om de gameObjecten goed gescheiden te kunnen houden. Een alternatief zou twee aparte lijsten in de LevelController zijn geweest, maar dit zou zorgen voor een onoverzichtelijke LevelController.
* NextRound
  + NextRound een Button waarop gelikt kan worden om de volgende ronde te laden.
* Player
  + De player klasse is de codering van de character die de speler bestuurd in de game. Deze klasse draait bijvoorbeeld de character mee met de muis van de speler en laat hem ook naar de besturing van de speler lopen. Ook houd deze klasse bij welke wapens de speler momenteel allemaal gebruikt en zorgt hij ervoor dat het huidig geselecteerde wapen afgevuurd word als de speler klikt.
* Powerup
  + De powerup klasse is verantwoordelijk voor alle power ups in de game, en voor het geld. Op het moment dat een bully dood gaat is er een kans dat er een willekeurige power up spawnt. Wat deze power up doet, en wat er gebeurt als de speler hem op pakt word allemaal geregeld in deze klasse.
* Resume
  + Resume is een Button waarop gelikt kan worden om het spel te vervolgen.
* SoundController
  + De SoundController zorgt ervoor dat alle geluiden afgespeeld kunnen worden, en maar één keer ingeladen worden. Ook zorgt deze controller voor het afspelen van de background muziek.
* Table
  + De tafel is een object dat een groot deel van de game vult. Naast in de game wereld staan doet de tafel niet veel. De speler kan niet bovenop een tafel klimmen, tenzij hij dit via een bankje probeert.
* Trashcan
  + De trashcans zijn pure obstakels in de game. De speler kan niet door ze heen lopen, en kan ook niet via andere obstakels boven op ze klimmen.

Voor de shop, samen met de wapens, worden de volgende klassen gebruikt:

* Upgrade
  + De upgrade klasse representeert de upgrade button in de shop. Elke button word toegewezen aan een bepaald wapen, en een bepaald attribuut van dit wapen. Als de speler op deze knop klikt zal dat bepaalde attribuut van het wapen worden geupgrade, als de speler daar genoeg geld voor heeft.
* switchWeapon
  + De switchweapon is een button die er voor zorgt dat de speler in de shop kan wisselen tussen de beschikbare wapens op een bepaald slot in zijn inventory
* Weapon
  + De weapon klasse is de parent klasse van alle wapens. Dit stelt ons in staat om alle verschillende wapens in één lijst te plaatsen, en ook makkelijk te wisselen tussen wapens in de inventory van de speler.
* WeaponCard
  + De weapon card is wat je ziet in de shop. Het lijstje met het plaatje van het wapen, de upgrade buttons en de statistieken van een wapen. Deze klasse zorgt er voor dat al deze elementen goed worden weer gegeven.
* WeaponManager
  + De weapon manager zorgt voor de wapens die speler bij zich heeft en dat alle upgrades behouden blijven bij het wisselen van een level. Ook kan de weapon manager alle upgrades resetten bij het restarten van de game.
* Shop
  + Shop is een Button waarmee naar het level Shop wordt gegaan.
* ShopCard
  + ShopCard is een overkoepelende klasse voor de shop. De ShopCard maakt de drie verschillende kaarten aan en laat deze updaten en tekenen.
* Knife
  + Knife zorgt ervoor dat je verschillende handwapens makkelijk gemaakt kunnen worden en deze kan laten zien op het scherm. Ook zorgt deze klasse ervoor dat er wordt gekeken of er een vijand door het handwapen wordt geraakt.
* Gun
  + Gun zorgt er voor dat er verschillende vuurwapens gemaakt kunnen worden. Gun zorgt voor het aantal kogels dat je in je wapen hebt zitten en ook hoe snel je kan schieten. Als er op de linker muis knop wordt gedrukt wordt er een nieuwe Bullet aan gemaakt die in de richting van waar het wapen naar toe staat heen wordt geschoten.
* Bullet
  + Bullet zorgt er voor dat er dat er verschillende kogels door het beeld kunnen vliegen. Als een kogel een vijand raakt wordt er aan de hand van welk wapen de kogel is geschoten, leven van de vijand afgetrokken. Bullet houdt ook in de gaten dat als de kogel buiten het beeld wordt, de kogel wordt verwijderd.

Met betrekken tot de bullies hebben we de volgende klassen:

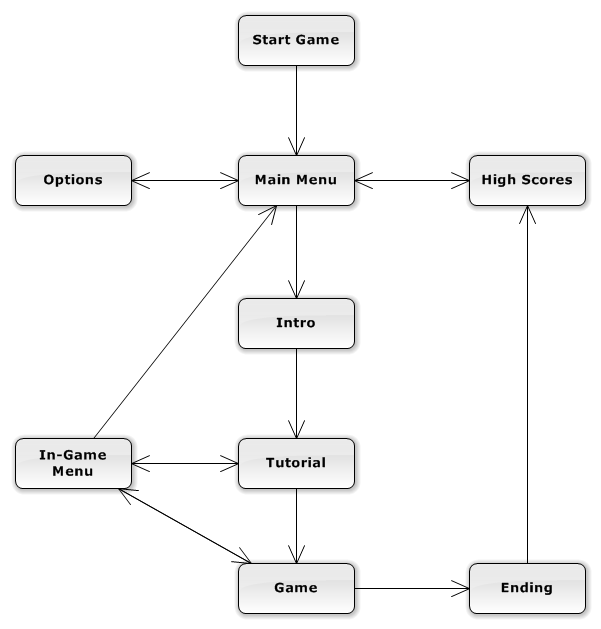
* bobbyBoss
  + Baas van de 5de Ronde
* dunkenBoss
  + Baas van de 1ste Ronde
* Enemy
  + Enemy is een basis klasse voor de normale bullies, maar een parent klasse voor de verschillende bazen. Enemy zorgt ervoor dat de bullies naar de speler toe lopen. Als de bullies dicht bij genoeg zijn zullen ze de speler slaan.
* ethanBoss
  + Baas van de 2de Ronde
* parkerBoss
  + Baas van de 4de Ronde
* zoeyBoss
  + Baas van de 3de Ronde

Voor de particles word gebruikt gemaakt van de volgende klassen:

* ParticleManager
  + De particleManager is de klasse die alle lijsten met particles bij houdt. De level controller heeft een pointer naar dit object, welke hij door kan geven aan elk particleEmitter object. De particleManager zorgt ook voor het updaten, en tekenen van elke particle in de game. We laten dit door één object doen zodat we konden kiezen op welke laag van de game de particles worden getekend. We creëren de particle manager na alle objecten in de game, maar vóór de player en de enemies, zodat de particles bovenop alle objecten worden getekend, maar de player en de enemies er nog steeds overheen lopen.
* Particle
  + De particle is in principe een hele simpele class. Op het moment dat hij word gecreëerd krijgt hij een positie, een richting en een snelheid mee. Elke update zal hij zijn snelheid verminderen tot hij stil komt te liggen, op welk moment hij zichzelf weer zal verwijderen. Het klinkt niet als iets heel spectaculairs, maar op het moment dat je er genoeg tegelijk maakt in een game, ziet het er toch een stuk beter uit.
* ParticleEmitter
  + Alle objecten die particles moeten kunnen spawnen extenden de klasse particleEmitter. In deze klasse kunnen alle eigenschappen van de particles worden geset. Op het moment dat een game object dan particles wil spawnen hoeft hij maar één simpele methode van de emitter aan te roepen, en dan zoekt de particleEmitter contact met de particleManager en zal hij een aantal particles toevoegen aan de game.

# Gebruikersmodellen

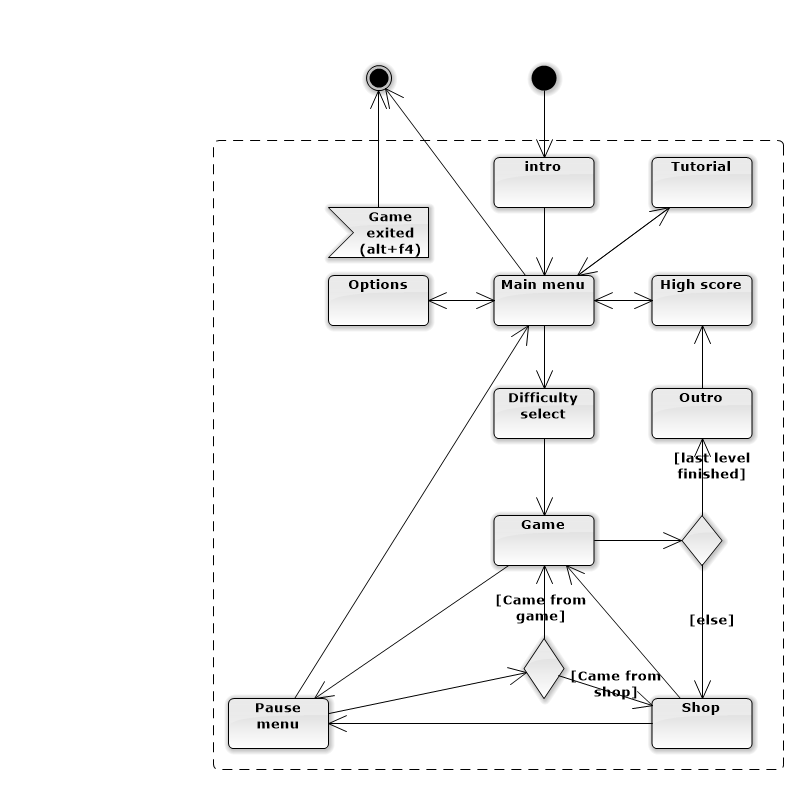
## 1. Game sequence



## 2. Klassendiagram



## 3. Activity diagram



# Testrapporten

## Weapon Upgrades

Ilias Ameziane

9-11-2014

Inleiding

Shoot The Bullet is een top down shooter, Het spel bevat wapens die ge-upgrade kunnen worden. De wapens moeten immers sterker kunnen worden naarmate je veder in het spel komt. In dit test rapport worden de upgrades dan ook uitvoerig getest.

Doelstelling

Als de speler bij het shop scherm op de linker muis drukt, bij de gekozen wapen upgrade. Dan moet het level van de upgrade met 1 geel balkje verhoogt worden(dit geldt alleen als de speler genoeg geld heeft voor de upgrade en als de wapen nog niet volledig ge-upgrade is). De wapen moet hierna ook daadwerkelijk sterker zijn geworden. De onderzoeksvraag luidt: “Wat gebeurt er met de wapens als je diverse upgrades koopt?”.

Verwacht wordt dat als de speler een upgrade komt, dat een geel balkje bij de gekozen upgrade wordt toegevoegd. Daarnaast moet de upgrade de wapen daadwerkelijk verbeteren.

Methode

Als testmethode wordt een demo versie van de game “Shoot The Bully” gebruikt. De tester gaat de game spelen en de upgrades en de effecten daarvan op de wapens uitvoerig testen.

De randvoorwaarden zijn:

-de speler moet upgrades kunnen kopen(de aantal gele balken laat zien welke level de upgrade is, geen betekent 0)

-de upgrade verbetert daadwerkelijk het wapen

De methode om te bekijken of de speler een upgrade kan kopen, bestaat uit een gametester en de game zelf. De gametester bekijkt met behulp van zijn ogen of er een geel balkje verschijnt als hij/zij een upgrade koopt in de shop. Omdat het balkje van zwart naar geel zou moeten veranderen indien de upgrade gekocht wordt, zijn de ogen een betrouwbare middel om te controleren of dat klopt(uiteraard voldoet de gametester aan de voorwaarde dat hij/zij niet kleuren blind is).

De methode om te bekijken of de gekochte upgrade het wapen ook echt beinvloed bestaat uit een gametester en het game zelf. Het spel bevat 3 verschillend upgrades: fire rate, damage, reload speed. Bij al deze upgrades kunnen de ogen zien of er een vershil is na de upgrade of niet, immers met meer damage gaan de vijanden sneller dood, met meer fire rate schiet je meer kogels per seconden, en als laatste met een snellere reload speed herlaad de wapen sneller. Dus de gametester gaat met behulp van zijn/haar ogen controleren of de gekochte wapen upgrade ook daadwerkelijk een effect op het wapen heeft.

Resultaten

Het kopen van de upgrades in het shop scherm, deed het prima. In de afbeeldingen hieronder kunt u het voor en na situatie zien.

afb 1: het voor situatie van het shop scherm(nog geen upgrades gekocht).

afb 2: na situatie, hier zijn enkele upgrades gekocht.

Zoals u kunt zien in de afbeeldingen als de speler een upgrade koopt dan komt bij de desbetreffende upgrade een gele balkje te staan in plaats van een zwart balkje.

Bij de wapens hadden alle upgrades hun effect.

-fire rate gaf meer snelheid in het afvuren van kogels, of te wel meer kogels per seconden. Op level 5 schoot het pistol 3 kogels per seconden meer dan zonder upgrades.

-damage gaf meer schade, de pistol kon een vijand op level 5 in 4 schoten neerhalen en zonder upgrades kostte dat 11 kogels.

-reload speed verkorte de wachttijd van het herladen, op level 5 herlaadde de pistol in 1 sec in plaats van 2 sec zonder upgrades.

Conclusie

De upgrades in het shop scherm kunnen gekocht worden, en het gele balkje laat zien dat de upgrade daadwerkelijk hebt gekocht en welke level de upgrade is.

De upgrades hebben ook daadwerkelijk hun effect op het wapen, de wapen word dus sterker naarmate je hem meer upgrade.

## 8 Direction movement buttons and character speed

Ilias en Arnout

28-09-2014

Inleiding

In Shoot The Bully gebruikt de speler movement buttons om zijn/haar character te besturen. In dit test rapport wordt getest of de movement buttons wel correct functioneren en of de beweeg snelheid goed is. Aangezien dit een belangrijk onderdeel is van de game, want zonder de buttons kun je niet spelen.

Doelstelling

Het doel is om te testen of de movement buttons wel correct functioneren, dit betekent dat er getest wordt of de character wel de juiste richting op gaat of dat de character überhaupt wel beweegt of niet, klopt de snelheid van de character?. Ook kijk ik wat de beste besturing is en hoe we de benen mee kunnen laten bewegen met de richting.

De onderzoeksvraag is: “functioneren de movement buttons en snelheid wel volgens de plan?”

De verwachting is dat de movement buttons werkenn volgens plan, of te wel:

- Dat de character beweegt.

- Dat de character de juiste richting op gaat.

- Dat de snelheid van de character klopt in alle 8 richtingen(links, rechts, voren, achteren en de 4 diagonale richting.

Methode

Dit wordt getest door de game “Shoot The Bully” op te starten en vervolgens de buttons 1 voor 1 uit te testen. De voorwaarden zijn als volgt: De snelheid klopt met de waarde dat is geprogrammeerd, de character beweegt als je een button indrukt en dat de character beweegt in de juiste richting.

- De snelheid wordt bekeken met de ogen van de tester en wiskundige formules die gebruik maken van tijd en afstand.

- Of de character beweegt of niet en daarbij ook nog of de character beweegt in de juiste richting of niet wordt getest met behulp van de ogen van de tester. De tester kan zien of de character beweegt en of de character in de juiste richting beweegt.

Voor de besturing ga ik beide opties proberen: de pijltjes toetsen en WASD en zorg ik dat de player daarbij mee beweegt. Ik kijk of de snelheid beweeg snelheid altijd het zelfde is in elke richting door alle mogelijkheden proberen.

Resultaten

Snelheid:

De snelheid klopte in de richtingen: links, rechts, voren en naar achteren, maar verassend genoeg klopte deze niet in de 4 diagonale richting. We hadden ondekt dat de beweging van het player object ~30% sneller was als de character een diagonale richting op liep.

Beweging/ correcte richting:

De character kon volgens plan bewegen in alle 8 richtingen en de daarbij behorende buttons

Conclusie

De character movement buttons functioneren goed en daarmee klopt een core mechanic van de game. De snelheid klopte niet bij de diagonale richting vanwege een klein programmeer foutje. Dit hadden we opgelost door onze berekening achter de beweging aan te passen.

De speler kan zijn character naar wens bewegen met de beschikbare buttons met een constante snelheid in alle 8 richtingen.

Door dit testverslag zijn we meer te weten gekomen over de beweging van het player object en hebben we het kunnen aanpassen.

## Bullets

Ilias Ameziane

9-10-2014

Inleiding

Shoot The Bullet is een top down shooter, om te kunnen shieten heb je naast wapens ook kogels nodig. Kogels zijn daarom een belangrijke core mechanic van de game en moeten daardoor uitvoerig getest worden.

Doelstelling

Als de speler op de linker muisknop drukt, moet er een kogel worden afgeschoten. Naast dat de kogel als sprite op het beeld moet verschijnen, moet de kogel ok het juiste richting op vliegen.

De onderzoeksvraag luidt: “Wat gebeurt er met de kogel als de speler op de linker muisknop drukt?”.

Verwacht wordt dat als de speler op de linker muisknop drukt, de kogel van de wapen wordt afschoten en verschijnt op het beeldscherm. Daarnaast vliegt de kogel ook in de juiste richting.

Methode

Als testmethode wordt een demo versie van de game “Shoot The Bully” gebruikt. De tester gaat de game spelen en de kogels uitvoerig testen.

De randvoorwaarden zijn:

-de kogel verschijnt vanuit het pistool zodra er op de linker muisknop wordt gedrukt

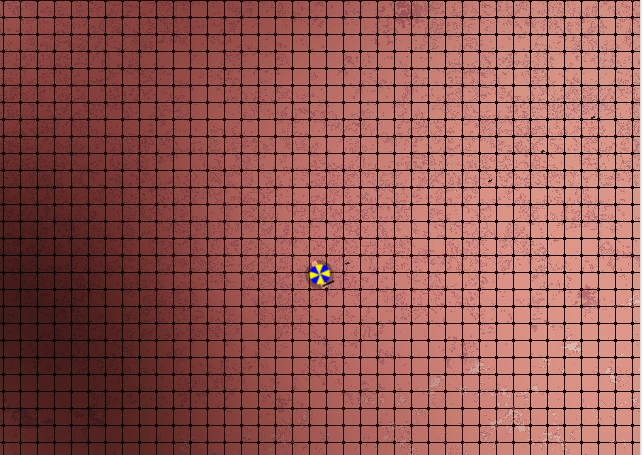
-de kogel vliegt in de juiste richting

De methode om te bekijken of de kogel verschijnt nadat er op de linker muisknop is gedrukt, bestaat uit een gametester en de game zelf. De gametester bekijkt met behulp van zijn ogen of de kogel verschijnt vanuit het pistool zodra de linker muisknop wordt ingedrukt. Omdat de sprite van de kogel vrij zichtbaar is en langzaam beweegt zijn de ogen een goede manier om betrouwbare resultaten te verkrijgen .

De methode om te bekijken of de kogel de juiste richting op vliegt indien hij verschijnt, bestaat evenals uit een gametester en de game zelf. Ook hierbij gebruiken we de ogen van de gametester, alleen kijken we nu of de kogel de punt van de muis passeert op de plaats waar de linker muisknop is ingedrukt. De ogen zijn hier uitstekend voor omdat de muis voor deze test op het zelfde plaats blijft en er dus alleen naar gekeken moet worden of de kogel de muis passeert.

Resultaten

De kogel sprite verscheen op de scherm zoals u op de figuur hieronder kan zien.



Helaas verscheen de kogel niet vanuit de pistool, maar vanuit het midden van de character.

De kogels vlogen de correcte richting op, alleen de sprite veranderde niet van richting. Volgens de sprite gaat de kogel rechtdoor terwijl hij naar rechtboven vliegt. Ook klopte de richting niet meer als de game resolutie wordt aangespast(beeldscherm groter/kleiner maken)

Conclusie

De kogels verschijnt op scherm alleen niet vanuit de gewenste plaats(de pistool in plaats van de speler). De kogel volgt de richting indien de scherm in zijn default resolutie blijft, en anders is er een afwijking. De kogel sprite past zich niet aan de gekozen richting.

De basis werkt volgens plan, de kogel verschijnt en gaat de juiste richting op. Alleen de bovenstaande minor problemen moeten nog worden verholpen.

## Collision

Daniël van den Berg

5-11-2014

Inleiding

Shoot The Bullet is een top down shooter, waarbij de speler door een level, met daarin meerdere willekeurige obstakels, kan lopen. De speler mag echter niet door deze obstakels heen kunnen lopen, maar tegen de obstakels tot stilstand komen.

Doelstelling

Als de speler tegen een obstakel loopt moet hij tot stilstand komen. Ook moet hij weer van dit obstakel weg kunnen lopen zonder er in vast te komen zitten.

De onderzoeksvraag luidt: “Werkt de collision volgens verwachting?”.

Verwacht wordt dat als de speler tegen een tafel aan loopt, hij tot stilstand komt en hierna weer weg kan lopen van de tafel.

Methode

Als testmethode wordt een demo versie van de game “Shoot The Bully” gebruikt. De tester gaat in de game op meerdere manieren tegen de tafel aan lopen.

De randvoorwaarden zijn:

-De speler komt tot stilstand tegen de tafel.

-De speler kan zonder problemen weer van de tafel weglopen.

-De speler kan via een bank, of andere tafel, op een tafel klimmen.

De methode om de eerste randvoorwaarde te testen is als volgt:  
1) De speler loopt in de richting van een tafel.  
2) Op het moment dat speler tegen de tafel aanloopt neemt hij waar of hij ook daadwerkelijk gestopt word.

De methode om de tweede randvoorwaarde te testen is als volgt:  
1) De speler loopt tegen een tafel aan en komt tot stilstand.  
2) De speler probeert de tegenovergestelde kant op te lopen en neemt weer of dit zonder problemen gebeurt.

De methode om de derde randvoorwaarde te testen is als volgt:  
1) De speler loopt naar een plek waar een bank tegen een tafel aan staat.  
2) De speler loopt op de bank.  
3) De speler loopt nu tegen de tafel aan.  
4) De speler loopt heen en weer over de tafel.  
5) De speler loopt nu, van de eerste tafel, over op de volgende tafel.  
6) De speler loopt van de tafel af, en valideert opnieuw de eerste randvoorwaarde.

Resultaten

De speler komt correct tot stilstand tegen een tafel. Ook kan de speler correct via een bank op de tafels klimmen. Echter komt de speler regelmatig vast te zitten in tafels. Dit gebeurt vooral als de speler zichzelf omdraait terwijl hij tegen de tafel aan staat.

Conclusie

De speler kan helaas niet zonder problemen van de tafel weglopen. Aan de andere randvoorwaarden is echter wel voldaan. Dit betekent dus dat er nog wel aan het collision systeem gewerkt moet worden, tot deze bug verholpen is.

# Conclusie

We hebben een game gemaakt die aan alle eisen voldoet, maar we hadden wel meer eisen aan ons eigen spel. Die we uiteindelijk door te weinig tijd niet hebben kunnen realiseren. Alsnog kunnen we met ze allen de conclusie trekken dat we een goeie en leuke game hebben gemaakt. Die zeker 10 minuten te spelen is en veel verschillende en goed uitgewerkte features heeft.

# Bronvermelding