UNIVERSITEIT GENT

# VERSLAG GSO

Moon Battles BattleBack 2 – Enhanced Lootbox Edition

Benjamin Fraeyman
GSO

# Contents

## Inleiding

Het spel die ik gemaakt heb is “Moon Battles BattleBack 2 – Enhanced Lootbox Edition” elke gelijkenis met star wars battlefront 2 is louter toevallig.

## Implementatie Wereld

Wordt gebruikt als achtergrond. Hiervoor wordt een WorldComponent gemaakt die de canvas bevat en een Texture. Ook wordt een een PositionComponent toegevoegd voor later de camera makkelijker te kunnen implementeren.

## Implementatie Speler

Bevat:

- PlayerComponent: om de speler te kunnen onderscheiden. Deze bevat de levenspunten, score en damagemultiplier.
- PositionComponent: positie van de speler.
- WeaponComponent: controle van huidige wapen met bijhorende wapensettings.
- TextureComponent: sprite van de speler.
- DifficultyComponent: Huidige moeilijkheidsgraad.
- HitBoxComponent: de hitbox van de speler.
- VelocityComponent: snelheid van de speler.
- ShieldComponent: schild actief of niet.

## Implementatie Vijand

Deze wordt gemaakt door de enemyfactory. Er werdt gekozen voor een factory om een makkelijk extra enemies toe te voegen.

Vijanden worden bijgehouden in het SpawnerSystem, hierdoor kan makkelijk gecontroleerd worden hoeveel enemies er zijn en hoeveel er moeten aangemaakt worden.

## Implementatie Wapen

Ik heb de keuze gemaakt om dit met een system te doen, dit leek mij het simpelste omdat je alle waardes op voorhand kan instellen en je niet in de code iedere keer die moet ingeven.

Naast het wapen heb ik kogels apart beschouwd, zo kan ik meerdere kogels maken en hoef ik geen rekening te houden met het aantal wapens.

## Doorlopen van het spel

1) InputSystem: controle van input.

   We overlopen alle ingeduwde knoppen en stellen ze een vector op die de richting van de speler aanduidt. (hierbij worden ook de knoppen gecontrolleerdt om een wapen te geven om het verbeteren makkelijker te maken)

   Hierna wordt de positie van de muis aangepast.

   Ten slotte wordt gekeken of de muis ingeduwd is of niet, om zo een kogel af te schieten of niet.

2) MovementSystem: verplaatsen van crosshair, player en enemies.

   Hier wordt rekening gehouden met de translatie van de camera.

3) CameraSystem: Correcte translatie van afstanden bijhouden.

   Hier wordt gecontroleerdt of we transleren of niet, of de speler beweegt of dat de achtergrond beweegt.

4) WeaponSystem: controle van welk type wapen de speler heeft.

   Om te weten of het wapen moet aangepast worden heb ik een ConditionalComponent, een verbetering is het systeem te registeren en unregisteren.

   Een wapen maken bestaat uit het aantal pellets, spread, enzovoort in te stellen.

   Wanneer het wapen ingesteld is verwijderen we de ConditionalComponent.

   *Edit* Dit heb ik na het maken van dit verslag aangepast zodat het systeem geregistered wordt wanneer de speler een nieuw wapen krijgt, hierdoor wordt de gameloop iets korter.

5) ReloadSystem: herladen van het wapen.

   Indien de juiste knop voor herladen is ingeduwd herladen we het wapen. Hierbij is er controle dat dit niet onmiddellijk gebeurt door een counter te maken die pass na bepaalde tijd de ammo weer volzet.

6) BulletSystem: aanmaken en verplaatsen van kogels.

   Enerzijds spawnen we een kogel wanneer er entity bestaat die een BulletComponent heeft en een TransformComponent. Anderzijds verplaatsen we een kogel wanneer die een BulletComponent heeft en een VelocityCompoenent.

   Bij het spawnen stellen we de huidige positie in en wordt een spread ingebracht en wordt een geluid afgespeeld (verbetering is dat geluid maar 1maal zou afgespeeld worden). Hierna wordt de TransformComponent verwijdert.

   Bij het verplaatsen wordt gecontroleert of de maximale range overschreden werdt of niet (indien deze kleiner of gelijk is aan 0). Wanneer de kogel van een launcher is dan ontploft deze ook.

7) HitRegSystem: Controle of enemy geraakt is of niet.

Om de code wat in te korten gebruiken we hier een optional player, die wordt dan automatisch aangemaakt. Hierdoor hoeven we niet te controleren of deze al bestaat.

We overlopen alle enemies in het spel en controleren voor iedere kogel of deze intersect met de enemie. Indien dit zo is dienen we schade toe aan de enemy, indien het een launcherbullet is dan ontploft deze ook.

Hierna wordt gecontroleerdt of de enemy nog genoeg leven heeft om verder te gaan.

Wanneer een enemy de speler aanraakt dan wordt er schade toegekend aan de speler (tenzij er een shield actief is). Wanneer de speler niet meer genoeg leven heeft sluit het spel af (verbetering is om alle systems uit te schakelen en enkel het renderen te behouden met een game over logo).

8) SpawnerSystem: spawnen van enemies en bonussen

   Indien het aantal enemies niet het maximum is dan maken we nieuwe enemies aan, dit maakt het spel iets moeilijker daar er een constante flow van enemies is. In plaats van deze enemies random op het veld te plaatsen, heb ik deze op 4 vaste posities rond de speler geplaatst. Hierdoor stijgt de moeilijkheid van het spel.

   Ook hebben we een mogelijkheid van een bonus die kan spawnen. Dit gebeurt wanneer een random getal een waarde heeft tussen 0 en 1000 ( 1 kans op 1000). Dit type crate is een weaponcrate, het is een gegenereerde crate die een random wapen bevat. De positie is vooraf ingesteld boven de speler. Dit is niet random omdat dit spel zich focust op lootcrates, een actueel onderdeel van bijna alle spellen.

9) XPSystem: bijhouden van score en moeilijker maken van het spel.

   Hier heb ik een extra functie gemaakt “isBetween”. Deze functie controleert of een waarde tussen 2 andere waardes ligt.

   Indien de score van de speler groter of gelijk is aan de benodigde score dan maken we het spel moeilijker en wordt de score gereset en wordt de benodigde score hoger geplaatst. Om de speler te helpen wordt er wel een lootcrate van het type 2 gegeven. Dit type lootcrate is de levelcrate. In deze levelcrate zitten de 4 bonussen die normaal konden gekozen worden maar door de druk van de investeerders nu achter een random systeem zitten (had ik gevraagd op het eerste overlegmoment en kreeg ik toestemming voor om te doen).

   Daarna wordt de moeilijkheid effectief aangepast. In plaats van deze volledig generiek te maken heb ik er voor gekozen om dit incrementeel in de code te plaatsen. Zo heb ik effectieve stages in het spel en omdat het wapen random toegekend wordt is de kans dat je zonder extra crates te kopen het spel verslaat zeer klein (op vraag van investors zo gemaakt). Indien je toch aan het maximale level raakt sluit het spel zich af waardoor je toch alle progressie kwijt bent en je moet herbeginnen.

10) BonusSystem: staat in voor het verwerken van bonussen.

    Om de code wat in te korten gebruiken we hier een optional player, die wordt dan automatisch aangemaakt. Hierdoor hoeven we niet te controleren of deze al bestaat.

    Eerst controleren we of er bonussen zijn met een ConditionalComponent, deze zijn crates die en wapen bevatten en zorgen ervoor dat we de het WeaponSystem moeten doorlopen.

Daarna controleren we de rest van de bonussen.

Indien de speler over de hitbox loopt en deze nog niet geopend is dan wordt de geopend en kan de speler zien welke upgrade hij kreeg.

Nu dat de crate geopend is kan de speler over de upgrade lopen om zo de verbetering te krijgen. Er wordt telkens in console uitgeprint wat de upgrade is zodat deze kan gecontrolleerd worden nadien.

Iedere upgrade doet iets anders, de meeste zitten direct ingewerkt in het BonusSystem, maar andere maken gebruik van een ander system die nu pas in het systeem wordt geplaatst.

Nadat alle bewerkingen op de bonus zijn toegepast wordt deze uit het spel gehaald.

11) SoundSystem: geluiden afspelen.

    Dit plaatste ik als laatste in mijn spel, het is niet de beste implementatie omdat ik niet alle types geluiden overloop, sommige laat ik direct in de code afspelen.

12) RenderSystem: alles naar het scherm tekenen gebeurdt hier.

    Dit systeem wordt als laatste doorlopen en plaatst alle textures op het scherm.

    Hier is het nut van de camera het meeste zichtbaar, sommige textures moeten getransleerd worden en andere niet.

    Hier heb ik redelijk wat tijd ingestoken om de afbeeldingen te roteren met de setRotate functie.

    Ook heb ik besloten om de ui hier te coderen en uit te printen daar het gewoon wat waardes overlopen is en een paar balken te tekenen.

    Om het verbeteren makkelijker te maken kan je door bepaalde toetsen in te drukken jezelf maximaal leven geven of een ander wapen geven.

## Gebruikte patterns / functies

- Factory: Aanmaken van bonussen, enemies en bullets
- Singleton: alle textures worden maar 1 keer ingeladen
- Iterator: Bij de spriteloader heb ik een iterator gebruikt om over de images te lopen. Ik weet dat dit niet de beste methode is maar ik vind dat het een speciale toepassing is van iterators, ik heb deze daarom niet vervangen met de methode die ik gebruik om mijn explosies te tekenen.

## Geteste patterns

- Flyweight: om telkens maar 1 enemy of bullet te maken heb ik geprobeerd deze met een flyweight te doen. Het probleem dat ik hierbij had is dat eens een enemy gedood was, alle enemies dood waren. Dit kwam doordat alle enemies dezelfde component gebruikten en dus tegelijk dood gingen.
- Strategy: Ik heb geprobeerd om mijn launchercode en normale wapen code te splitsen (ook bonuscode, maar ik heb dit in verschillende systems gestoken). Ik heb dit uiteindelijk moeten

opgeven omdat ik er te veel tijd aan het insteken was, achteraf gezien had dit een makkelijke fix moeten zijn maar door tijdsgebrek heb ik die niet extra meer geimplementeerd.

## Besluiten

- In dit labo heb ik leren omgaan met deels gemaakte code.
- Ik heb mijn spel vergeleken door op steam het spel Crimsonland eens te spelen en heb daardoor bepaalde beslissingen genomen die ik anders niet ging gedaan hebben.
- Ik vind het jammer dat ik niet extra veel patterns heb kunnen gebruiken, achteraf gezien vind ik nu veel dingen die ik misschien beter anders had gedaan maar door tijdsgebrek heb ik besloten de werking voorop te plaatsen.
- !Alles is gecodeerd op een 144Hz scherm, indien dit getest wordt op een 60Hz scherm loopt alles veel trager en is het spel makkelijker!
- Mijn gameloop tijd is maximaal 148ms in het begin omdat hij alles inlaad, dit duurt 1-2 loops, daarna is mijn gameloop ongeveer 5ms met maximaal ongeveer 20ms.
- Volgend semester heb ik wat minder vakken en ik hoop een 2$^{de}$ implementatie van het spel te maken die veel efficienter is en meer kan.
- Bij het maken van het spel waren factories en singleton mijn belangrijkste patterns.
- De tijd die ik besteed heb aan het labo ligt niet laag, ik heb een werkend spel in 4 weken ontwikkeld.