|  |
| --- |
| TOelichting gyaraga\_  OOPD Beroepsproduct |

Timo Berendsen & Jasper Vastenholt

29 maart 2023

INHOUDSOPGAVE

[1 Gamelevel 4](#_Toc131001192)

[2 Timers 4](#_Toc131001193)

[3 ProjectileFactory 4](#_Toc131001194)

[4 Projectile 4](#_Toc131001195)

[5 Mob 5](#_Toc131001196)

[6 Enemy 5](#_Toc131001197)

INLEIDING

Dit document is een korte toelichting op de verschillende classes en methodes die geimplementeerd zijn in het spel Gyaraga. Dit spel is het resultaat van de eindopdracht van de cursus OOPD aan de HAN.

# Gamelevel

In het spel zitten meerdere levels, en om meerdere toe te kunnen voegen hebben we een class GameLevel aangemaakt die extended worden door GameLevel1 en GameLevel2 enzovoorts.

Binnen GameLevel bestaat er een public static instance variabele van één van deze levels, namelijk degene die gespeeld wordt. Deze variabele is ontzettend handig sinds je dan vanuit andere classes bijvoorbeeld addEntity() kan doen. Zonder deze variabele zouden we moeten werken met EntitySpawners die naar onze mening vrij lastig waren om te implementeren.

# Timers

Het spel maakt gebruik van timers om tijden bij te houden. Deze timers zitten in Yaeger gebouwd en waren naar onze mening best makkelijk te implementeren. De timers doen simpelweg iets om de zoveel milliseconden, deze worden berekend (zoals getRandomShootInterval() doet) of staan vast (zoals bij Explosion; na 350 milliseconden is de explosion sprite klaar, of zoals bij GameLevel; deze checkt elke 10 milliseconden of je gewonnen of verloren hebt).

# ProjectileFactory

Het spawnen van alle kogels was best een uitdaging. Om het werkend te krijgen hebben we een class genaamd ProjectilleFactory gemaakt. De ProjectileFactory krijgt als parameter mee welk wapen was afgevuurd en of het door een player of niet was. Binnen projectile zit een "getProjectile" die afhankelijk van de input de goede instantie maakt.  
Welk wapen is geschoten wordt via de ENUM ProjectileType gedaan. Dit is een klein lijstje met een aantal verschillende wapens. Dit wapen is te selecteren vanuit de game in de scene "WeaponSelect".

# Projectile

De projectile class is waar het hele spel om draait, zonder deze is namelijk er geen gameplay en kan je niet winnen en/of verliezen. Elke projectile heeft zijn eigen damage en speed en houd bij of het geschoten is door de speler of niet (firedByPlayer). firedByPlayer is nodig omdat anders de hit registratie mis gaat, anders zouden de mobs en de player namelijk zichzelf raken. Projectile heeft de methode hitMob(); deze regelt de hit registratie van een projectile op een mob. hitMob() wordt overrode door Rocket sinds deze een explosie creeert als het een mob raakt en de andere projectiles niet. Ook heeft projectile de methode shoot(); deze regelt de startpositie en richting van de projectile.

# Mob

Om alle enemies te spawnen hebben wij een class genaamd "Mob" deze is verantwoordelijk voor het inladen van de sprite van de enemies. Als er een mob moet worden gespawned geeft deze zijn sprite en locatie mee en wordt op het scherm geplaatst. Mob kan ook worden aangeroepen door de enemies om een kogel te spawnen. De shotgun werkt iets anders deze spawnt vijf kogel die allemaal naast mekaar geplaatst moeten worden.

# Enemy

De class "Enemy" extends mob deze geeft de functies van alle enemies door aan de class "Mob" sommige enemies hebben een speciale ability die ook in Enemy wordt aangeropen. Ook is deze class handig voor bijhouden of de enemy geraakt is.

