|  |
| --- |
| Pixel Potato |
| Technische onderbouwing |
| Get Off Me |

|  |
| --- |
| Niek Oosterbaan, 0884141 & Swen Meeuwes, 0887127  8-10-2017 |

Inhoudsopgave

[Moeilijkheidsgraad - Game aspecten 2](#_Toc495255431)

[Medium - Online output (Google play service) 2](#_Toc495255432)

[Medium - Physics 2](#_Toc495255433)

[Medium - Collectables 3](#_Toc495255434)

[IVial.cs 3](#_Toc495255435)

[SpeedVial.cs 3](#_Toc495255436)

[Accept methode in concrete class 3](#_Toc495255437)

[Hard - Infiniteness 4](#_Toc495255438)

[Hard - Combo system 4](#_Toc495255439)

[Hardcore - Level Editor 5](#_Toc495255440)

[Hardcore - Gesture-based controls 6](#_Toc495255441)

[Gameplay entity abstraction 7](#_Toc495255442)

[Appendix A, Screenshots van editors 8](#_Toc495255443)

[Tutorial player 8](#_Toc495255444)

[Spawner editor 9](#_Toc495255445)

[Player editor 9](#_Toc495255446)

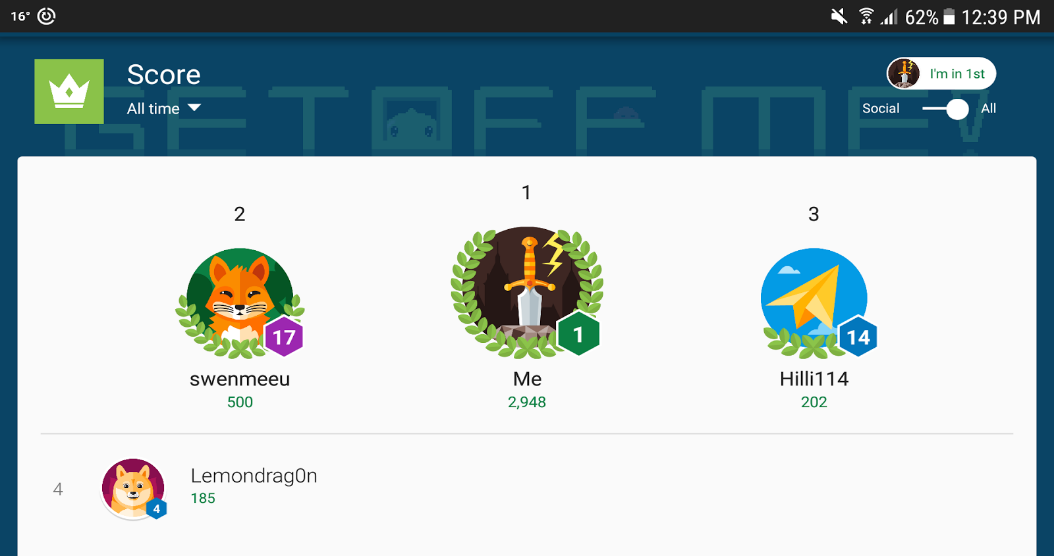
[Enemy editor 10](#_Toc495255447)

[Vial context 11](#_Toc495255448)

[Phase editor 11](#_Toc495255449)

# Moeilijkheidsgraad - Game aspecten

## Medium - Online output (Google play service)

Voor de leaderboards (en misschien wel toekomstige achievements) is er voor gekozen om gebruik te maken van ‘Google Play Services’. Dit platform biedt ons de mogelijkheid om leaderboard, achievements en events te beheren. Verder is de beheerder van het platform (Google) betrouwbaar.

De speler kan hier gebruik van maken zodra deze een account heeft bij Google Play Services. Als de speler deze al niet heeft natuurlijk! Dit is al gauw het geval omdat Google Play Services door meerdere games op de Play Store gebruikt wordt.

## Medium - Physics

In het begin was het physics system nog custom gemaakt, het was toen nog erg simplistisch. De movement van de slimes naar de player ging volgens de volgende formule; . Waarbij Pf = positie, Pi = initiële positie, v = velocity, t = time en a = acceleration. Echter toen er elementen zoals wrijving toegevoegd moesten worden is de keus gemaakt om over te stappen naar het physics systeem van Unity. Dit systeem verving toen het oude systeem en had als extra al een wrijving functionaliteit ingebouwd.

## Medium - Collectables

Om de moeilijkheid van het spel op te krikken is er een systeem gekomen waarbij er collectables geactiveerd kunnen worden als risk/reward; Vials. Deze zijn geïmplementeerd als volgt: Er is een interface “IVial” die elke concrete class van een vial implementeerd. In deze interface staan meerdere “Apply()” functies die elk een parameter heeft van de verschillende soorten type objecten die een vial kan modificeren. Zo kan het object dat gemodificeerd moet worden aan de vial gegeven worden, en kan hij aangepast worden binnen de concrete class van een vial. Om dit te laten werken is er gebruik gemaakt van het “Visitor” design pattern. Elke concrete class die aangepast kan worden door een vial moest een functie “Accept” implementeren die de Apply functie aanroept. Hier geldt de ‘Apply’ methode als de ‘Visit’ call en de concrete vials als de visitors.

Dit ziet er als volgt uit in code:

### IVial.cs



### SpeedVial.cs



### Accept methode in concrete class



## Hard - Infiniteness

De insteek van het design was bepalend dat het een infinite spel zou worden. Het stress-effect wat men ervaart wanneer de speler groeit heeft vooral geholpen in dit besluit. Met een level systeem zou dit effect veel minder voorkomen.

De implementatie van een infinite level was goed te doen. Er is gekozen voor een “phase” systeem. Dit houdt in dat er per “phase” ingesteld kan worden wat voor soort vijanden/allies spawnen en hoe vaak. Wanneer de laatste “phase” bereikt is, blijft hij deze gebruiken om vijanden/ allies te spawnen. De gamedesigner heeft dan vanaf dit punt de keus om de spawn snelheid lineair of stijgend te maken. De grootste uitdaging hierin was voor de designer om een goede balans te vinden tussen moeilijkheidsgraad en plezier. Om deze game balans makkelijk te veranderen zijn ook hiervoor editors gekomen, zie “Hardcore - Level editor”.

## Hard - Combo system

Het combo systeem is uiteindelijk zo gemaakt dat spelers meer rewards krijgen voor het dichterbij laten komen van vijanden. Een risk/ reward factor dus. Dit is gedaan om de veteraan spelers toch een uitdaging te geven en het voor de casual spelers nog wel haalbaar te houden.

Het werkt als volgt: Er is een cirkel aangegeven voor de speler. Wanneer een vijand binnen de cirkel weg geswipet wordt, krijgt men +1 combo. Per extra combo krijgt men een variabele multiplier op je score die je krijgt per kill. Na X aantal combo’s gaat men de volgende “combo tier” in. Dit zorgt ervoor dat de cirkel waarin je de combo ontvangt, verkleind wordt.

## Hardcore - Level Editor

Bij het ontwikkelen van dit project hebben we verschillende tools aangeboden om het spel makkelijk te kunnen tweaken. Zo kon de game designer efficiënt te werken gaan!

Verder was dit niet alleen fijn voor de game designer maar ook voor de programmeurs, omdat dit de standaard was werd alles gelijk dynamisch opgezet.

Voorbeelden van de editors zijn:

* Tutorial sequence editor
  + Hierin kan de tutorial sequence aangepast worden door verschillende stappen te definiëren. De stappen kunnen later makkelijk toegevoegd, verplaatst, bewerkt en verwijderd worden.
* Spawner editor
  + Geeft toegang tot de spawn intervallen en snelheid van de spawns. Deze zijn afgezet tegenover de speeltijd en kunnen aangepast worden in een curve.
* Player editor
  + In deze editor kan de hoeveelheid levens en de shockwave ability aangepast worden.
* Enemy editor
  + Hierin kunnen vijand gerelateerde kenmerken aangepast worden. Voorbeelden hiervan zijn:
    - Gewicht
    - Levens
    - Snelheid
    - Random snelheid offset
    - Het aantal punten dat ze waar zijn
* Vial Context
  + Hierin staat wat iedere vial (difficulty modifier) doet en wat de unlock conditions zijn.
* Phase editor
  + Hierin kunnen de soort vijanden die spawnen bepaald worden. Er moet een tijd ingesteld worden vanaf hoeveel seconde de phase ingaat, en de “weights” van alle enemies. Dit werkt als volgt: een entity met weight 6 spawnt 1.5x zo vaak als een entity met weight 4. Dit “weight” systeem zorgt ervoor dat het makkelijker is om nieuwe entities toe te voegen en om de spawn rates in runtime aan te kunnen passen. Dit gebeurd bijvoorbeeld bij één van de vials, die ervoor zorgt dat er meer medics spawnen (+1 weight voor medics in elke phase).
  + Wanneer men een nieuwe entity toevoegt in deze interface, een prefab selecteert en weights invult, zal deze meteen meegenomen worden in de game.

**Screenshots van deze editors zijn terug te vinden in Appendix A.**

## https://lh4.googleusercontent.com/6yBprwQFbl7OQxGzaRlSGujcdJspSZayOONlOguE45sh5HXM-2lwOHHj0S-QzoXh_3S6CB62mvoZOq8RDCkZQiUXKI7ouw4NDF5CzG7xqFuXdGvAlT6cXrSawDM5ETTfhACzPNr9Hardcore - Gesture-based controls

Om de speler te belonen voor het halen van hoge combo’s is de ‘shockwave’ ability geïntroduceerd. Deze ability kan de speler activeren door middel van een ‘zoom out’ gesture.

Deze gestures (en mogelijk toekomstige gestures) worden afgehandeld in de ‘Input Manager’. Hierin wordt gekeken of de speler het scherm aanraakt met 2 vingers (of meer). Als dit het geval is dispatched de Input Manager een event die de het verschil tussen de lengte van de 1ste vector en 2de vector bevat.



# Gameplay entity abstraction

In onze game is het belangrijk dat we makkelijk nieuwe enemies en/ of allies kunnen toevoegen, bewerken en verwijderen. Hiervoor is een schaalbare en beheerbare architectuur nodig. Om dit te bereiken is er gebruik gemaakt van abstractie. Zo is de swipe functionaliteit geabstraheerd van de entiteit en heeft zo een eigen klasse gekregen. De ‘AbstractDraggable’ klasse bevat code om ‘iets’ te kunnen swipen of draggen. Echter heeft deze geen informatie over *wat* het gaat aansturen.

De abstractie lagen van de entiteiten zijn hieronder in het klasse diagram te vinden.

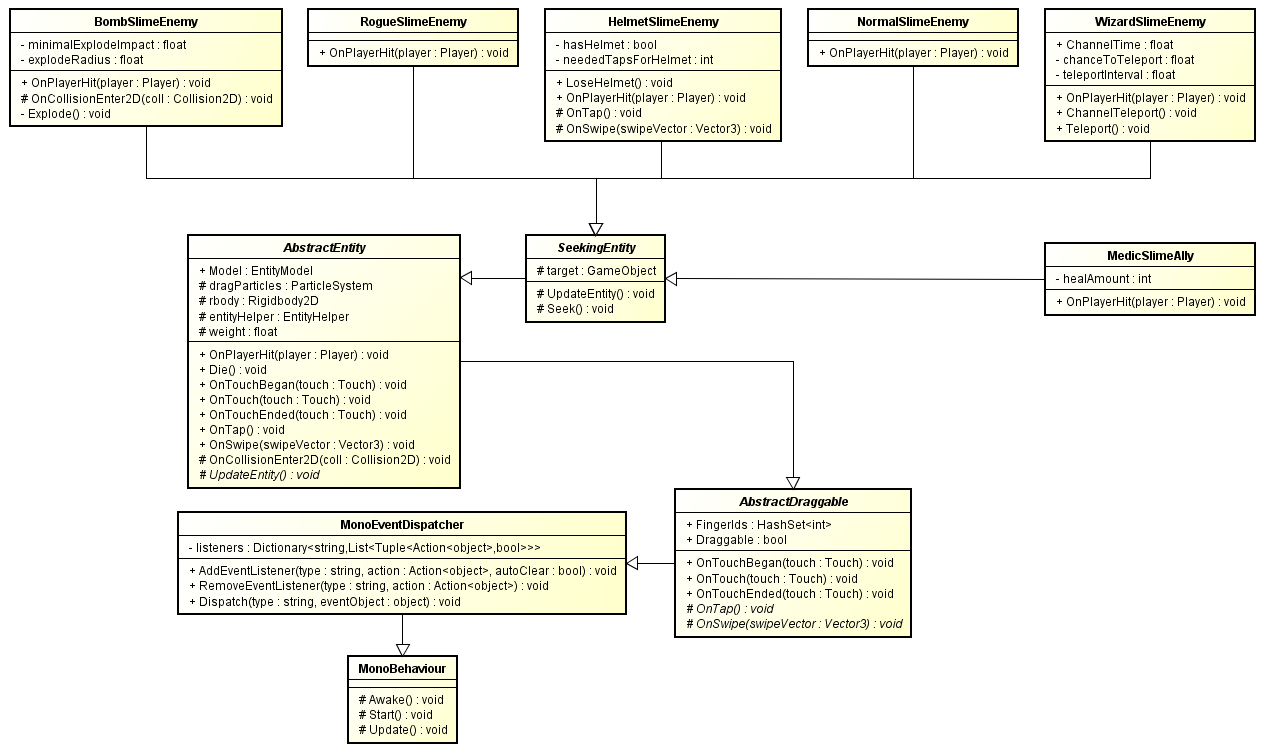
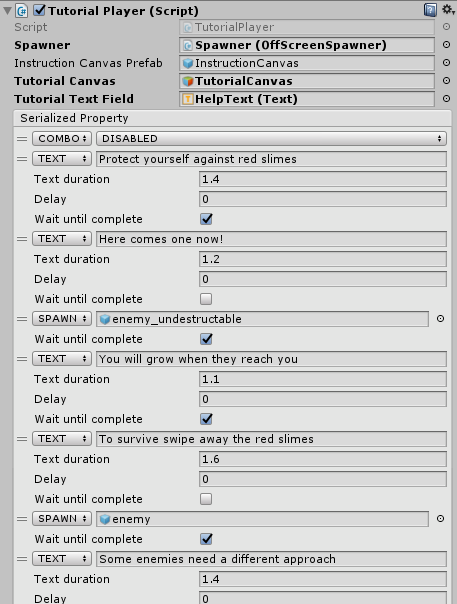


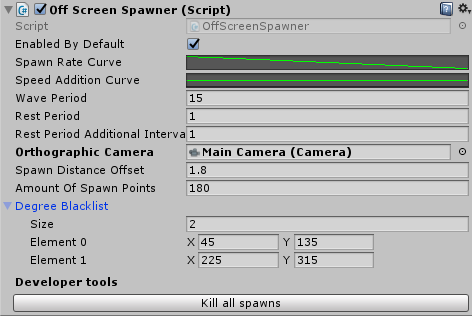
Figure , Klasse diagram 'entiteiten'

# Appendix A, Screenshots van editors

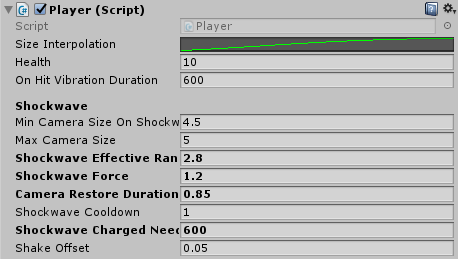
## Tutorial player



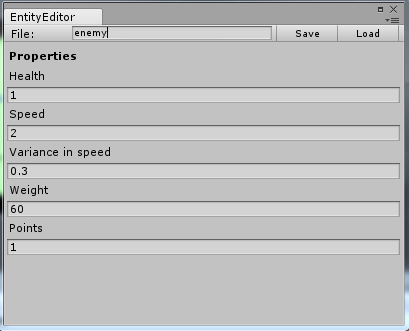
## Spawner editor



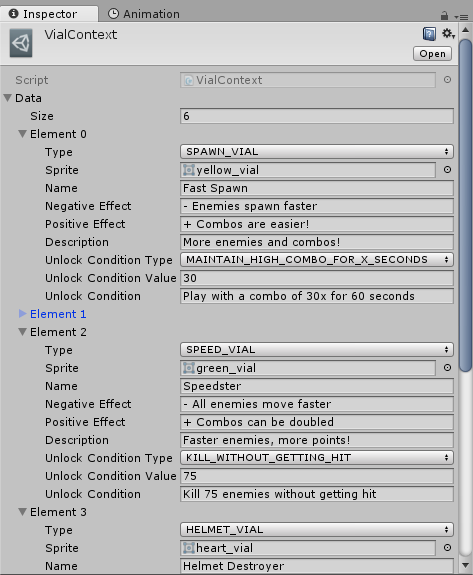
## Player editor



## Enemy editor



## Vial context



## Phase editor

