|  |  |
| --- | --- |
| FIIW | logo FIIW.png |

Taak C++

Motherload

Laatste wijziging: 15 januari 20

Auteurs: Claessen Bram Kellens Maikel

Groep: EA-ICT1

Labobegeleider: Wouter Groeneveld

# Inleiding

In dit kort verslag vind u een beschrijving van de opbouw van ons spel aan de hand van een domain model.

# Algemeen model



Figuur de algemene flow van de werking van het spel

In Figuur 1 ziet u dat het programma start in main, hier wordt de engine gemaakt en geüpdatet. Hoe de engine werkt is buiten de scope van dit project. Verder wordt er in de main enkel de ‘start\_screen\_scene’ gemaakt.

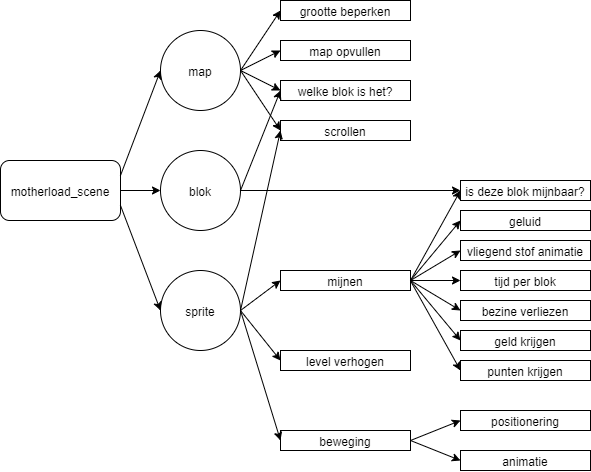
De start\_screen\_scene houdt op zich ook niet veel in. Hier wordt de timer, die later gebruikt zal worden, gestart.

Na de start\_screen\_scene, wordt er over gegaan naar de instruction\_scene. Deze laat enkele instructies en tips zien over hoe het spel best gespeeld wordt.

Na de start\_screen\_scene komt de motherload\_scene, en hier begint het echte werk. In de motherload\_scene wordt alles wat met de gameplay te maken heeft, uitgevoerd. Deze scene is ook te uitgebreid om in dit hoofdstuk te bespreken, een meer uitgebreide bespreking is te vinden in hoofdstuk .

Als laatste is er de game\_over\_scene, en zoals te zien in , staat tussen deze en de motherload\_scene een pijl in 2 richtingen. Dit is omdat er van beide scene’s naar de ander gegaan kan worden. In de game\_over\_scene worden de highscores bepaald en opgeteld, ook deze scene is ietwat uitgebreider en zal besproken worden in hoofdstuk .

# Motherload\_scene



Figuur blokschema motherload\_scene

Zoals te zien in , is de motherload\_scene op te delen in 3 stukken, met elks hun eigen verantwoordelijkheden. Deze worden hieronder meer in detail besproken.

## Blok

Er zijn verschillende mogelijke blokken in het spel, deze hebben allemaal hun eigen specificaties, zoals het uiterlijk, het verkregen geld bij het mijnen, de tijd die het kost om te mijnen,… Ook de lucht en de vrije plaats onder de grond wordt gezien als een blok.

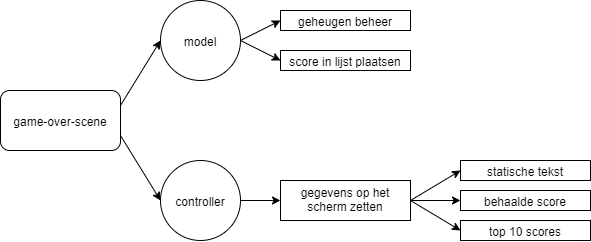
## Map

De map bestaat uit verschillende blokken, die er pseudo willekeurig in worden geplaatst aan de hand van de timer die gestart wordt in de start\_screen\_scene. Verder is de map beperkt in grootte, we hebben er voor gekozen om enkel in de y-richting te scrollen, en om de map in de x-richting te beperken op de randen van het scherm. Om de map op het scherm te laten zien wordt er een venster uit de globale map gekopieerd naar een kleinere map, die dan wordt afgebeeld op het scherm. Het is ook op ieder moment mogelijk om op te vragen welke blok op een bepaalde locatie in de map zit

## Sprite

De sprite heeft de meeste verantwoordelijkheden. Als eerste moet deze bij het verhogen van een level, een aantal parameters, zoals de grootte van de benzinetank, aanpassen. Verder moet de sprite, als deze moet bewegen, z’n coördinaten kunnen aanpassen, en van animatie veranderen. Ook is het belangrijk dat de sprite in het midden van een blok staat als deze die wil mijnen. Of dat de sprite valt als de blok onder de sprite lucht is. Het belangrijkste is natuurlijk het mijnen, heel het spel draait er rond. Het is belangrijk dat de sprite de juiste blok mijnt, en niet een blok meer naar links of naar rechts, dit hangt een beetje samen met de positionering. Een blok moet ook te mijnen zijn, bijvoorbeeld een steen of lucht kan niet gemijnd worden. Verder zijn er bij het mijnen heel veel parameters die moeten veranderen, het is bijvoorbeeld logisch dat diamanten mijnen, zilver mijnen of stilstaan een verschillend benzine verbruik heeft. Dus wordt de snelheid hiervan aangepast. Er zijn ook punten en geld te verdienen als een blok wordt gemijnd, maar opgelet, hiervoor dient de blok wel volledig gemijnd te worden, als er halfweg wordt gestopt, worden er geen geld of punten verdient, er gaat wel sneller benzine af. Verder wordt er bij het mijnen een beetje geanimeerd met wegvliegend stof.

# Game\_over\_scene



Figuur blokschema game\_over\_scene

Deze scene is eigenlijk vrij goed op te delen volgens het model-view-controller principe. De view is op een lager level geïmplementeerd en niet herkenbaar in onze code, de controller en het model is wel heel duidelijk herkenbaar.

## Model

In het model wordt het SRAM geheugen uitgelezen en wordt berekend of de behaalde score al dan niet in de top 10 moet komen, en als die er in moet komen, op welke plaats.

## Controller

In de controller worden de behaalde score, de top 10, en nog enkele statische tekst, op het scherm gezet.