Laurens de Roy en Joshua van Synghel

Documentatie SNAKe

Inhoud

[1. Design Snake 1](#_Toc126410585)

[2. Verantwoordelijkheden & Afhankelijkheden 1](#_Toc126410586)

[2.1 Module Settings 1](#_Toc126410587)

[2.2 Module Element 2](#_Toc126410588)

[2.3 Module Score 2](#_Toc126410589)

[2.4 Module Winnaar 2](#_Toc126410590)

[2.5 Module Food 2](#_Toc126410591)

[2.6 Module Snake 2](#_Toc126410592)

[2.7 Module EntriesScoreboard 2](#_Toc126410593)

[2.8 Module SnakeGame 2](#_Toc126410594)

[2.9 Module Controller 2](#_Toc126410595)

[2.10 Module ControllerDeus 2](#_Toc126410596)

[2.11 Module LocalStorage 2](#_Toc126410597)

[3. Scorebord 2](#_Toc126410598)

[4. Local Storage 3](#_Toc126410599)

# Design Snake

Voor een overview van het design van Snake (onderdeel van mijn spelletjes website) zie document: Design.jpg

Het design van Snake heeft het Model View Controller designpatroon. Er zijn twee controllers die reageren op events in de view en die de view updaten. De controllers communiceren met de domeinlaag om kennis te hebben van de huidige toestand van het spel en voor het updaten van het spel. De domeinlaag heeft geen kennis van beide controllers.

# Verantwoordelijkheden & Afhankelijkheden

Dit hoofdstuk bespreek de modulen die onderdeel zijn van Snake, met een beschrijving van de verantwoordelijkheden en afhankelijkheden.

## Module Settings

Settings beheert een verzameling van constanten die door het spel gebruikt worden. Gekozen is om de constanten in een module op te nemen en niet om deze over alle modulen te verspreiden. Wij beschouwen dit namelijk als algemene settings van het spel. Een voordeel is dat deze algemene settings snel kunnen worden aangepast (bijv NUMFOODS). Nadelig is wellicht de hogere koppeling die ontstaat omdat meerdere modulen kennis moeten hebben van Settings.

## Module Element

Module Element is een object dat de bouwsteen vormt voor zowel de slang als het food. Element is niet afhanelijk van een andere modulen binnen de domeinlaag.

## Module Score

Module Score beheert de huidige score van het spel. De score kan alleen door andere modulen worden benaderd via get- en setmethoden. De module Score is niet afhanklijk andere modulen binnen de domeinlaag.

## Module Winnaar

Via de module Winnaar beheert de mogelijkheid om binnen het spel een bekend te maken. De winnaar bevat de opgegeven naam van de speler binnen de website. Deze variabele kan alleen worden benaderd via get- en setmethoden. Deze module is niet afhankelijk van andere modulen binnen de domeinlaag.

## Module Food

Food beheert de functionaliteit voor het maken van de voedsel elementen binnen het spel.

## Module Snake

## Module EntriesScoreboard

## Module SnakeGame

## Module Controller

## Module ControllerDeus

## Module LocalStorage

# Scorebord

In de specificatie wordt het voorstel gedaan om in de website functionaliteit op te nemen die het aantal keer bijhoudt dat de speler wint of verliest. De opdrachtnemers hebben gekozen om van dit voorstel af te wijken. In plaats van de voorgestelde functionaliteit, is ervoor gekozen om een scorebord in de website op te nemen die de top 3 ‘high scores’ bijhoudt die zijn behaald door verschillende spelers. Dit geeft een wat meer realistischere (en leukere) invulling van Snake.

De gekozen functionaliteit bezit alle uitdagingen, qua programmeerwerk, die samenhangen met de voorgestelde functionaliteit: spelers moeten bekend kunnen zijn binnen het spel, scores moeten kunnen worden onthouden en scores moeten worden bijgewerkt in verschillende velden. Er zou zelfs betoogd kunnen worden dat de gekozen functionaliteit complexer is omdat er meerdere spelers bekend kunnen zijn binnen het spel, behaalde scores met elkaar vergeleken moeten en er een aanvullende dynamiek is tussen het scorebord en het spel.

Het scorebord kent drie plaatsen die ingevuld kunnen worden met behaalde scores. Als het scorebord lege plaatsen kent, dan komt een behaalde score altijd op het scorebord. Indien het scoreboard 3 scores bevat dan wordt een behaalde score vergeleken met de scores om te bepalen of er sprake is van een high score (behaalde score is hoger of gelijk aan minstens één van de bestaande scores). Is dit het geval dan wordt de behaalde score op het scorebord geplaatst en verdwijnt een andere score. De bovenste entrie van het scorebord bevat de hoogste score en de onderste entrie de laagste. Als een behaalde score gelijk is aan een bestaande score op het scorebord, dan eindigt de behaalde score boven de al bestaande score op het scorebord. Hiermee belandt een speler na het behalen van de max score altijd op het scorebord en slaat het spel niet dood doordat er geen mutaties meer kunnen plaatsvinden in het scorebord als deze gevuld is met alleen maar maximum scores.

# Local Storage

Voor de het onthouden van spelgegevens in de browser is de WebStorage API omdat deze een vrij eenvoudige methode biedt voor het implementeren van de gevraagde functionaliteit. In de Local Storage van de webbrowser (in dit geval Firefox) worden gegevens onthouden als een (key, value) paar waarbij de key uniek is. Wordt er informatie toegevoegd aan de Local Storage met een key die al bekend is in de storage, dan wordt de value van deze key aangepast. Een van de uitdagingen is dat een Local Storage in principe ook door andere applicaties gebruikt kan worden, dus daar zal Snake rekening mee moeten houden.

Het spel maakt alleen gebruik van drie specifieke keys: placeOne, placeTwo, placeThree. Deze worden bewaard in een map structuur als entries van het scorebord. De value is het object EntryScore, die één score entrie weergeeft, met als eigenschappen een score en de naam van de winnaar. Deze map is bekend binnen de module Entries Scorebord en deze wordt ook verwerkt in de Local Storage. Hierbij wordt het object EntryScore wel omgezet naar een stringformat.

Als het spel Snake wordt opgestart dan wordt eerst de inhoud van de Local Storage ingelezen. Er vind daarna een filtering plaats om te zorgen dat de keys overblijven die in het spel worden gebruikt. Zijn de gebruikte keys aanwezig, dan wordt de value toegekend aan de entries van het scorebord, mits de value leeg is of het object EntryScore bezit. Naderhand wordt de map met de scorebord entries verder gevuld met lege object van EntryScore, zodat deze volledig is. Nadat een behaalde high score het scorebord wijzigt, wordt ook de Local Storage aangepast.