Laurens de Roy en Joshua van Synghel

Documentatie SNAKe

Inhoud

[1. Design Snake 1](#_Toc126501427)

[2. Verantwoordelijkheden & Afhankelijkheden 1](#_Toc126501428)

[2.1 Module Settings 1](#_Toc126501429)

[2.2 Module Element 2](#_Toc126501430)

[2.3 Module Score 2](#_Toc126501431)

[2.4 Module Winnaar 2](#_Toc126501432)

[2.5 Module Food 2](#_Toc126501433)

[2.6 Module Snake 2](#_Toc126501434)

[2.7 Module EntriesScoreboard 2](#_Toc126501435)

[2.8 Module SnakeGame 2](#_Toc126501436)

[2.9 Module CanvasController 3](#_Toc126501437)

[2.10 Module ScoreboardController 3](#_Toc126501438)

[2.11 Module LocalStorage 3](#_Toc126501439)

[3. Scorebord 3](#_Toc126501440)

[4. Local Storage 4](#_Toc126501441)

# 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 (en aangepaste) toestand van het spel. De domeinlaag heeft geen kennis van beide controllers. Een belangrijke keuze wat betreft het design van de webapplicatie, is dat indien de slang buiten het canvas loopt deze aan de andere kant te voorschijn komt.

# Verantwoordelijkheden & Afhankelijkheden

Dit hoofdstuk bespreekt de modulen die onderdeel zijn van Snake, met een beschrijving van de verantwoordelijkheden en afhankelijkheden. Als er gesproken wordt van een koppeling tussen modulen, dan wordt hiermee bedoeld dat een module (eenzijdig) kennis nodig heeft van de andere module, waarmee dus een afhankelijkheid wordt aangeduid.

## Module Settings

Settings beheert een verzameling van constanten die door het spel gebruikt worden. Gekozen is om de constanten niet te verspreiden over verschillende modulen, maar om deze op te nemen in één module. 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 koppelingsgraad die ontstaat omdat meerdere modulen kennis moeten hebben van Settings.

## Module Element

Module Element beheert de bouwsteen, namelijk het object Element, voor zowel de slang als het voedsel. Element is niet afhankelijk van andere modulen.

## Module Score

Module Score beheert de huidige score van het spel. De score kan alleen door andere modulen worden benaderd via een get- en setmethode. De module Score is niet afhankelijk van andere modulen.

## Module Winnaar

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

## Module Food

Food beheert de functionaliteit voor het maken van de voedselelementen binnen het spel. Hiervoor is een koppeling nodig naar Element en Settings, omdat een voedselelement uiteindelijk een Element is met een standaard kleur en specifieke coördinaten.

## Module Snake

Snake beheert de functionaliteit voor het creëren van de slang binnen het spel. Hiervoor is een koppeling nodig naar Element en Settings, omdat de slang een reeks van Element objecten is met per segment een standaard kleur en specifieke coördinaten.

## Module EntriesScoreboard

EntriesScoreboard beheert de scores op het scorebord. Een score (entry) bestaat uit de naam van een winnaar, de behaalde score en de plek op het scorebord (plaats 1, 2, 3). Deze module bevat onder meer ook functies voor het aanpassen van de scores op het scorebord en het controleren of een behaalde scores een ‘high score’ is. Hiervoor is een koppeling nodig naar de module Score.

## Module SnakeGame

SnakeGame beheert de elementen (slang en voedsel) die daadwerkelijk op het Canvas zullen worden getekend. Na iedere beweging van de slang moet worden bepaald op basis van een aantal factoren (nieuwe richting, botsing met andere elementen op het canvas etc.) wat de nieuwe coördinaten zijn van de slang. SnakeGame bevat onder andere functionaliteiten voor het creëren en updaten van elementen. Hiervoor zijn er koppelingen nodig naar de module Snake en Food.

Daarnaast speelt SnakeGame ook een rol in het bepalen en vastleggen van het spelresultaat. Na iedere beweging van de slang wordt beoordeeld of het spel ten einde is gekomen naar aanleiding van bijvoorbeeld een botsing en aansluitend wordt bepaald wat het eindresultaat is. In deze module zijn daarom de variabelen aanwezig die de status van het spel en de plek van de behaalde score op het scorebord beheren. Om te bepalen wat de plek van de behaalde score is, bevat SnakeGame een koppeling naar module EntriesScoreboard. Als laatste is er ook een verbinding naar Settings.

## Module CanvasController

CanvasController zorgt ervoor dat alle elementen (slang en voedsel) op het Canvas worden getekend. Hiervoor is een koppeling nodig naar SnakeGame (en Settings) omdat de (nieuwe) coördinaten nodig zijn voor het tekenen van de elementen op het Canvas. Deze module bevat ook de functionaliteit voor het starten en stoppen van Snake. Hiervoor is een koppeling nodig naar scoreboardController omdat bij het starten en stoppen van het spel de inputvelden voor de naam van de winnaar moeten worden verwijderen.

## Module ScoreboardController

ScoreboardController is verantwoordelijk voor het scoreboard, het scoreveld en het inputveld voor de naam van de winnaar, binnen de applicatie. Hiervoor zijn koppelingen nodig met de modulen EntriesScoreboard, Score en Winnaar. Ook heeft deze controller de verantwoordelijkheid voor het halen en muteren van de gegevens die in Local Storage aanwezig zijn. Vandaar dat er ook een koppeling is met module LocalStorage.

## Module LocalStorage

Module LocalStorage is verantwoordelijk voor het inlezen en muteren van gegevens in de Local Storage van de webbrowser (in dit geval Firefox). Deze module heeft geen koppelingen met andere modules.

# Scorebord

In de specificatie wordt het voorstel gedaan om in de website functionaliteit op te nemen welke 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 bestaande scores om te bepalen of er sprake is van een high score (dwz 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 plaats van het scorebord bevat de hoogste score en de onderste 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. Hiermee belandt een speler na het behalen van de maximum 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 gebruikt, omdat deze een vrij eenvoudige methode biedt voor het implementeren van de gevraagde functionaliteit. Andere opties zijn verder niet overwogen. 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, een object met als eigenschappen een score en de naam van de winnaar. Deze map is bekend binnen de module EntriesScoreboard 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 worden eerst de keys van de Local Storage ingelezen. Er vindt daarna een filtering plaats om te zorgen dat alleen de keys overblijven die in het spel worden gebruikt. Vervolgens worden met de overgebleven keys, de values opgehaald. Deze inhoud wordt opgeslagen in een map. Hierna worden de (key,value) paren toegekend aan de map entriesScoreboard, mits de value leeg is of het object EntryScore bezit. Naderhand worden eventuele keys zonder value in de map entriesScoreboard gevuld met lege objecten van EntryScore, zodat de map volledig is. Nadat een behaalde ‘high score’ het scorebord wijzigt, wordt ook de Local Storage aangepast.