Project Gevorderd Programmeren
Space Invaders

Alle opgegeven voorwaarden in de opdracht zijn geïmplementeerd in het project.
De source code is opgedeeld in twee namespaces: Entites en MVC.
In Entities zijn alle spel-entiteiten geïmplementeerd. Alles wat in het spel een positie en een oppervlakte heeft wordt gezien als spel-entiteit.
In MVC is het MVC-pattern samen met het Observer-pattern geïmplementeerd.

Namespace Entities

Alle spel-entiteiten wodern afgeleid van de Entity base class. Deze bevat int members “x” en “y” die zijn positie in een 2D veld voorstellen en “width” en “height” die zijn oppervlakte voorstellen. Afgeleide klassen hiervan bevatten kleine toevoegingen.
Zo heeft onder andere de klasse Wall een int member hp. Hiermee kan je een schild vormen voor de speler. Andere klassen in de namespace Entities breiden gelijkaardig Entity uit om alle spel-entiteiten te kunnen voorstellen.

Daarnaast bevat Entities nog een Factory class die volgens het Abstract Factory Design werkt. Hiermee kunnen op een simpele wijze objecten gemaakt worden van Entity-afgeleide klassen die geinitialiseerd zijn op bepaalde waarden.

Namespace MVC

MVC staat hier voor Model View Controller, het design dat in combinatie met het Observer pattern gebruikt wordt voor het project.

De klasse Observer is een interface voor het Observer pattern dat gebruikt wordt. De klasse GameView leidt hier van af. GameView is een observer van GameModel.
GameModel zal dan op zijn beurt updates sturen naar GameView zodat GameView zijn members kan updaten. Zo wordt het observer pattern in werking gesteld.
GameView heeft niet enkel één functie notify() die alle members zal nakijken en updaten, maar een aantal verschillende notify functies. Zo kunnen er veel nutteloze updates worden vermeden.

Bijvoorbeeld: De invaders bewegen enkel tussen bepaalde tijdsintervallen (neem bv.1 seconde). De speler beweegt 60 keer per seconde naar links of rechts. Met één grote notify functie zouden de invaders 60 keer per seconde geupdated worden, wat nutteloos is. Door notify op te splitsen kunnen we de invaders enkel op de juiste momenten updaten.

De klasse GameModel is het model in het MVC pattern en bevat al de spel logica en spel-entiteiten.
GameView is de View van het MVC pattern en bevat enkel members uit de SFML library. Deze zijn nodig voor de grafische output van het spel. Een kleine toevoeging aan de library van mezelf is de MySprite klasse. Deze bevat een integer waarmee we een klein animatie effect kunnen geven aan de invaders.
Dan komen we terecht bij GameController, deze heeft GameModel en GameView als members.
GameController zorgt voor de interactie tussen de speler en GameModel en dus ook GameView.
De interactie tussen speler en spel is mogelijk gemaakt door de SFML library.

Exception Handling

Exceptions komen in het programma voor wanneer er een fout optreed tussen het programma en de buitenwereld. In dit geval worden er enkel exceptions gethrowed wanneer er een image of text font ontbreekt of fout wordt ingeladen. Er wordt dan een MyException gethrowed, deze afgeleide klasse werd aangemakt om simpelweg snel te kunnen throwen met een korte omschrijving van de fout. Bijvoorbeeld: throw MyException("Could not load spriteSheet");