Requirements Analysis Document

Mattias Torstensson, Karl Wikström, Agnes Mårdh, Luka Mrkonjic

May 2017

Contents

Version: 3

Date: 27-05-2017
Revised by: Mattias Torstensson, Agnes Mårdh
This version overrides all previous versions.

1 Introduction

In this modern day and age games have inclined towards a more and more casual and wider audience than ever before. Nowadays games more than often serve as a tool for defeating boredom and wasting time while for example waiting for the bus or the lecture to start, which is Hook Hero's main goal. In simple terms the application's purpose is to provide quick entertainment by challenging the user to beat their previous record in the game, similar to an arcade game. The game is easy to learn but hard to master, which provides a great user experience for both experienced gamers as well as new ones. The target audience for the game are people who like challenging themselves in games via trial and error. Due to the easy controls of the game it's suitable to people of all ages. Since one "playthrough" of the game probably won't last longer than a couple of minutes due to the constant threat of a game over, the game can practically be picked up and played whenever the user feels like it.

1.1 General characteristics of the application

The application will be a offline singleplayer desktop game, that will be runnable on Mac/Windows/Linux operating systems.

The game will be a sidescrolling platformer of the endless runner variety. As such the game will feature a single player character that is always running forward relative to the world. The world will consist of endless procedurally generated obstacles that the player has to overcome by using the character's available movement abilities: jumping aswell as attaching its hook to an object in the world and swining across gaps with it. The player will accumulate score over time until they trigger the game's fail state either by crashing into something in the world or by falling off it. There is no win condition in the game, it will keep running until the fail state triggers.

1.2 Definitions, acronyms and abbreviations

- Player: The player playing the game, the player controls the Character.
- Character: The character that the player controls.
- World: The environment in which the character exists.
- Side scroller: A video game genre in which the gameplay action is viewed from a side-view camera angle.

- Hook: A tool a character uses to swing from the ceiling.
- In-game: When the player is actively playing the game, in other words while the game is unpaused and the player is not in a menu.

2 Requirements

2.1 User interface

The application will feature a graphical user interface which at the very least uses geometrical shapes of different colors to show the character and the world of the game. It should be possible to update the interface to show textures and/or animations.

2.2 Functional requirements

- In the main menuy the player will be able to:
    - Start the game
    - View previous highscores
        - Close the view of previous highscores
- In-game the player will be able to:
    - Jump
        - Fall
    - Use grappling hook
        - Detach grappling hook.
    - Trigger a game over
        - By colliding with an obstacle
        - By falling off the map
- In the game over screen the player will be able to:
    - View and save their score
    - Restart the game
    - Go to the main menu

Use cases sorted by priority, highest priority first:

- Jump
- Use grappling hook
- Trigger a game over

- View and save their score
- Restart the game
- Main Menu
- Go to the main menu

2.3 Non-functional requirements

2.3.1 Usability

The game is supposed to be easy to pick up and play for anyone and as such it should not require anything other than the short list of controls available in-game in order to be played.

2.3.2 Reliability

N/A

2.3.3 Performance

Input from the player should appear to be instantaneous at all times and the game should not suffer from any noticeable stuttering.

2.3.4 Supportability

The game should be runnable on Windows, Mac, and Linux operating systems.

The application should have automated tests for all parts that are viable for automated testing, any part of the application that doesn't have automated testing should be tested manually.

2.3.5 Implementation

To make the application possible to run on different operating systems it will use Java. The application will use the framework LibGDX for the graphical user interface.

2.3.6 Legal

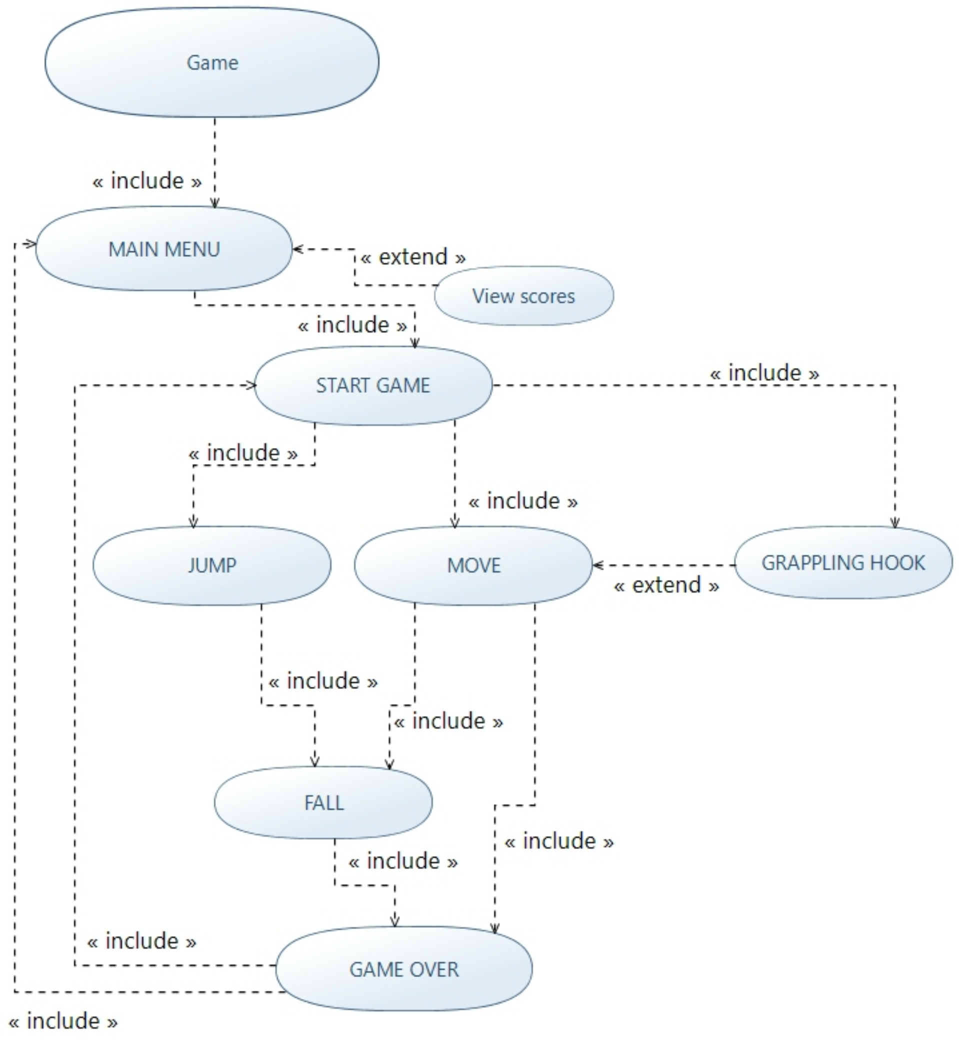
The LibGDX framework is distributed with the Apache 2.0 license[1].

2.3.7 Packaging and installation

The application will be delivered as a runnable .jar file with all the required assets inside of it. Because of this users will be required to have the Java Runtime Environment installed.

3 Use cases

Use case overview:



3.1 Use case listing

MAIN MENU
Summary: This is how the user navigates through the main menu.
Priority: Medium
Extends:
Includes: START GAME
Participators: Player

Normal flow of events:
Player presses the “Start Game” button.

	Actor	System
1		shows main menu
2	Clicks ”Start” button	
3		Starts the game

Alternative flow:
Player presses the “High Scores” button.

	Actor	System
1		shows main menu
2	Clicks ”High Scores” button	
3		Shows the top 10 high scores
4	Clicks ”close” button	
5		Closes the high score list

START GAME
Summary: This is what happens immediately after the player starts the game via the main menu.
Priority: High
Extends:
Includes: JUMP, MOVE, GAME OVER, GRAPPLING HOOK,
Participators: Player

Normal flow of events:

	Actor	System
1		Starts a new instance of the game.
2		Runs the game.

JUMP
Summary: The player presses the jump button.
Priority: High
Extends:
Includes: FALL
Participators: Player, Character

Normal flow of events:
Player character jumps without anything special happening.

	Actor	System
1	Player presses space	
2		Moves character upwards
3	Character reaches peak of jump	
4		See FALL

Alternative flow:
Player character jumps and collides with an object above it.

	Actor	System
1	Player presses space	
2		Moves character upwards
3	Character collides with ceiling	
4		See GAME OVER

MOVE
Summary: The character moves to the right at all times while in-game. The camera follows the player so it gives the impression that the world moves towards the player.
Priority: High
Extends:
Includes: FALL, GAME OVER
Participators: Character

Normal flow of events:
Player character moves without anything special happening.

	Actor	System
1		Moves character to the right. Increases score.

Alternative flow:
Player character moves off of the object it is currently standing on.

	Actor	System
1		Moves character to the right. Increases score.
2	Falls off ground	
3		See FALL

Alternative flow:
Player character runs into an obstacle.

	Actor	System
1		Moves character to the right. Increases score.
2	Collides with obstacle	
3		See GAME OVER

FALL
Summary: This happens when there isn't any ground below the player and they are not currently jumping or swinging with the grappling hook.
Priority: High
Extends:
Includes: GAME OVER
Participators: Character

Normal flow of events:
Character falls and lands on ground.

	Actor	System
1		Moves character downwards.
2		Character lands on a platform and stops falling.

Alternative flow:
Character falls and misses the ground.

	Actor	System
1		Moves character downwards.
2	Character falls offscreen.	
3		See GAME OVER

GAME OVER
Summary: The player triggers a game over via FALL or MOVE.
Priority: High
Extends:
Includes: START GAME, MAIN MENU
Participators: Player, Character

Normal flow of events:
Player triggers the game over screen and wants to try again.

	Actor	System
1		Character stops running.
2		Display 'game over' dialog and saves score
3	Clicks on "Play Again"	
4		Removes the game over dialog and restarts the game. (See START GAME)

Alternative flow:
Player triggers the game over screen and wants to go back to the main menu.

	Actor	System
1		Character stops running.
2		Display 'game over' dialog and saves score
3	Clicks on "Main Menu"	
4		Returns the player to the main menu screen (See MAIN MENU)

GRAPPLING HOOK
Summary: The player uses the grappling hook.
Priority: High
Extends: MOVE
Includes:
Participators: Player, Character

Normal flow of events:
Player isn't currently using the grappling hook and decides to use it.

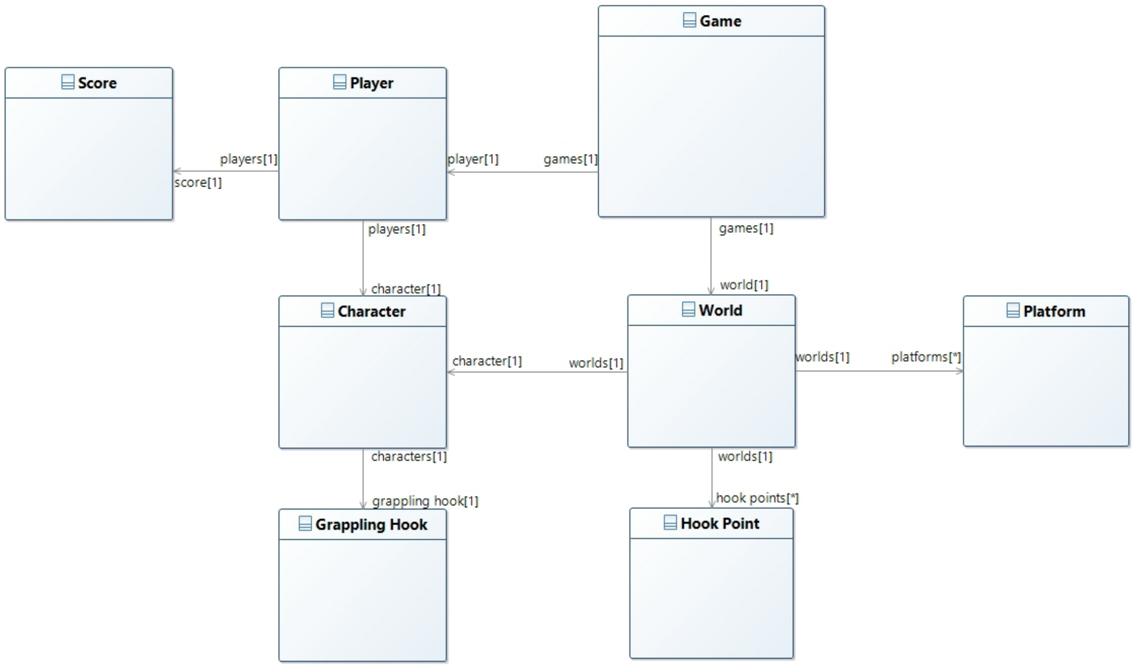
	Actor	System
1	Player presses the grappling hook button.	
2		Attaches hook to hook point, if available

Alternative flow:
Player is already using grappling hook and doesn't release the button while the character is moving.

	Actor	System
1		Character swings with rope
2	Player releases the grappling hook button.	
3		Grappling hook detaches and character stops swinging.

Alternative flow:
Player is using the grappling hook and releases the button.

4 Domain Model



4.1 Class responsiblities

- Game, the overall representation of the game.
- Player, the person playing the game.
- Score, the score of the player has in the current session of the game.
- Character, the character the player is playing as.
- Grappling Hook, the grappling hook the character can use to get around.
- World, the world the character is in. Contains procedurally generated Platforms and Hook Points.

- Hook Point, a part of the world that the character can attach its hook to.
- Platform, a platform in the world that the character can stand on top of, or collide with the side of.

References

[1] *LICENSE*. From 27 April 2017. URL: `https://github.com/libgdx/libgdx/blob/master/LICENSE`.