Requirements and Analysis Document for Alien Run

Group 28

Atosa Daneshvar-Minabi, Alexander Gilabert, Ann Heijkenskjöld and Isak Wideskott

Objektorienterat Programmeringsprojekt TDA376
Chalmers tekniska högskola

October 24, 2022

Contents

1 Introduction

Platformer games have been of the most recognizable types of games for a long time, ever since some the most classic examples such as *Super Mario Bros* and *Sonic the Hedgehog* gained popularity. This project, which is a desktop application, is a side-scroller platformer game where an Alien runs in space while shooting and jumping over obstacles, inspired by the Swedish game *Geometry Dash*. The object of the game is to stay alive by avoiding the obstacles and collect as many gems as possible to gain the highest score a user can achieve. The purpose of the game is for the user to enjoy an easy to learn single-player game in their spare time for enjoyment purposes. Therefore it is made for non-experienced users as well as more experienced gamers. This is possible because the user can choose their preferred difficulty level when starting the game.

1.1 Definitions, acronyms, and abbreviations

- Desktop application - A software program that runs stand-alone on the desktop, instead of e.g. from a browser.
- Side-scroller - A side scrolling game, generally in 2D that scrolls from left to right or the other way round.
- Platformer - A game in which a player can run and jump on a platform.
- GUI - Graphical User Interface.
- UML-Diagram - Unified Modeling Language Diagram, a way to visualize a software system.
- MVC - Model View Controller, a software design pattern to structure a program.
- LibGDX - A Java development framework used to code games.
- Figma - A tool used to design GUIs.
- Sprite - An image that represents a game asset.

2 Requirements

All User Stories are ordered in estimated importance for the project. At the end of the project 24 out of 30 planned User Stories were implemented.

2.1 User Stories

1. ### Start Game
   Description: As a user, I want to have a functioning game that starts without any difficulties.
   Implemented: Yes.
   Acceptance criteria:
   - The player should be able to run a simple program on their machine with the framework LibGDX.
   - There should be a running application with the window pop-up.
   - The game shouldn't take more than 10 seconds to start.

2. ### Player Sprite
   Description: As a user, I want to be able to see and manipulate my player character.
   Implemented: Yes.
   Acceptance criteria:
   - The player should have the Alien character as a sprite that follows the players exact position.

3. ### Obstacles
   Description: As a user, I want obstacles I can maneuver in the game so that I feel challenged.
   Implemented: Yes.
   Acceptance criteria:
   - There should be at least one type of obstacle in the game.
   - The player should be able to see and jump over the obstacle.

4. ### Gems
   Description: As a competitive gamer, I want to be able to catch gems so that I can show off my skills in the game and have a goal other than not touching an obstacle.
   Implemented: Yes.
   Acceptance criteria:
   - There should be gems displayed on the screen that disappear when detected by the collision detector.
   - You should always be able to see the current points you have on the screen.
   - When you touch a gem you should gain points.
   - There should be a point algorithm that is determined by how far you have gotten in the game and the amount or types of gems you have collected.

5. ## Game Physics
Description: As a user, I want to have physics in the game so the movement feels natural.
Implemented: Yes.
Acceptance criteria:

- The game world should have a gravitational pull downwards all the time.
- The player should be able to jump in the game in a way that feels and looks somewhat natural as it does in the real world.

6. ## Collision Detection
Description: As a user I want to be able to notice when my player has collided with an object such as gems or an obstacle so I get a response from the game if I have gained a point or lost the game.
Implemented: Yes.
Acceptance criteria:

- The collision detector should know when a player has collided with an obstacle.
- The collision detector should know when a player has collided with a gem.

7. ## Simple Menu
Description: As an inexperienced gamer, I want a simple and intuitive menu when I launch the game so that I feel comfortable and guided.
Implemented: Yes.
Acceptance criteria:

- All buttons should have functionality on the page that leads to the expected results when clicking on them.

8. ## Score System
Description: As a competitive gamer I want a score system so that I know how well I've performed.
Implemented: Yes.
Acceptance criteria:

- The score should be present on the screen at all times during the gameplay so the user knows what score they have.
- The score should increase the further distance a player have travelled.
- The score should increase when a player collides, or rather catches, a gem.

9. ## Side-scroller
Description: As a casual gamer, I want a simple and consistent movement in one direction so the objective of the game is easy to understand from the start without having to read the instructions.

Implemented: Yes.
Acceptance criteria:

- The player should move to the right at a consistent speed.

10. **Gun**
Description: As a user I want the player to have a gun than can shoot obstacles so I have more than one way to avoid them.
Implemented: Yes.
Acceptance criteria:

- The player should have a gun that follows the player.
- When clicking the space bar the gun should fire a shot.
- The bullet and destroyable obstacle should disappear when the obstacle is hit with a bullet.

11. **Keyboard Input**
Description: As a casual gamer, I want the input keys to work as thought intuitively so that I can quickly jump in and play and learn as I go, without having to read the instructions.
Implemented: Yes.
Acceptance criteria:

- A player should be able to jump with the up arrow or the w key.
- A player should be able to get down quickly by pressing the down arrow or s key.
- A player should be able to shoot bullets from their gun by pressing the space key.

12. **Different Game Modes**
Description: As a user, I want an adjustable difficulty setting that's suited to my needs so that I can enjoy the game at the pace I want whether I think it's too easy or too hard.
Implemented: Yes.
Acceptance criteria:

- There should be an option of choosing an easy, intermediate and hard player that changes the difficulty of the game.
- The difficulty mode should be reflected in how the character jumps and runs.
- You should easily be able to distinguish which mode you're playing in based on the players sprite.

13. **Obstacle Variations**
Description: As a user I want different kinds of obstacles so that the game feels less monotonous and more exciting to play.

Implemented: Yes.
Acceptance criteria:

- There should be at least three different kinds of obstacles in the game.
- One type of obstacle should be destroyable by a gun bullet.

14. **Gem Variations**
Description: As a user I want different kinds of gems so that the game feels less monotonous and more exciting to play.
Implemented: Yes.
Acceptance criteria:

- There should be at least three different kinds of gems in the game.

15. **Moving Space Background**
Description: As someone who enjoys graphics in video games I want the world to have a more unique touch so that it feels well made and is more interesting to look at.
Implemented: Yes.
Acceptance criteria:

- There should be a moving space background that fits the theme of the game.
- The background should move faster if the player moves faster and vice versa.

16. **Player Animation**
Description: As a user I want my player to reflects the movements I make so the game feels more exciting to look at.
Implemented: Yes.
Acceptance criteria:

- The easy and default player should have running and jumping animations.
- The hard player should have the same animations as the other two but appear to be running faster.

17. **Pause Game**
Description: As a user I want to be able to pause the game so that I can take a break if needed to.
Implemented: No.
Acceptance criteria:

- There should appear a screen where you have the option to continue, exit or start a new game.
- All buttons on the pause screen should have functionality and work as expected.

18. ## Figma Design Screens
Description: As a user I want all the different game screens to look good which gives me the impression of a professionally made game.
Implemented: Yes.
Acceptance criteria:

- The instructions page should look like the Figma design.
- The menu page should look like the Figma design.
- The gameplay screen should look like the Figma design.

19. ## Game Over
Description: As a user I want to know when I've collided with an obstacle so that I know when I've lost the game.
Implemented: Yes.
Acceptance criteria:

- The collision detector should register a collision from the obstacle.
- The game should stop when the player has lost.

20. ## Graphical Variations of Players
Description: As a user I want the different difficulties or variations of players to be portrayed differently so I can easily distinguish between which mode I'm playing by quickly looking.
Implemented: Yes.
Acceptance criteria:

- The easy player should be displayed by an orange alien.
- The intermediate player should be displayed by a green alien.
- The hard player should be displayed by a purple alien.

21. ## Improved Gem, Obstacle and Gun Graphics
Description: As someone who enjoys graphics in video games I want the gems, obstacles and gun bullets to look like what they represent instead of just shapes so that the game feels more well-made and improves my user experience.
Implemented: Yes.
Acceptance criteria:

- The gems, obstacles and bullets should look like what they represent.
- Different variations of gems and obstacles should look different in comparison so that they're distinguishable.

22. ## Music
Description: As a user I want there to be music playing in the background so my experience is enhanced.
Implemented: Yes.

Acceptance criteria:

- There should be music playing in the background when the game starts.

23. **Smooth Performance**
Description: As a user, I want the game to be free of lag so that the performance of the game won't disturb my performance when I play.
Implemented: Yes.
Acceptance criteria:
    - There shouldn't be any noticeable lag when playing or starting the game at any point.
    - The game physics should be optimized and the textures should be reused and disposed of so they don't slow down the game.

24. **Instructions Screen**
Description: As an inexperienced gamer I want to have an option on the menu page of reading the games instructions so that I can understand how to play easily.
Implemented: Yes.
Acceptance criteria:
    - There should be an instructions page on the starting menu that takes the player to a screen which displays instructions on how to play the game.
    - The page should have a back to main menu button.
    - It should also have pictures of the players during the different game modes.

25. **Return from Mode Selection**
Description: As an indecisive user I want to be able to go back from the Game Mode page so that I can read the instructions or exit the game easily.
Implemented: Yes.
Acceptance criteria:
    - There should be a return button on the Game Mode page.

26. **Save Highscores**
Description: As a competitive gamer I want to be able to see my highest scores so I know how well I've performed.
Implemented: No.
Acceptance criteria:
    - There should be a button on the Menu that takes you to a highscore screen.
    - The screen should have a back to main menu button.
    - A player should be immediately notified when the game is over if they've beat their highscore.

27. **Options Screen**
Description: As a user I want there to be an options button so that I can customize my game experience.
Implemented: No.
Acceptance criteria:
    - There should be an options page on the starting menu that takes the player to a options screen.
    - The page should have a back to main menu button.
    - A user should be able to turn off the game music.
    - A user should be able to choose which keys to play with.

28. **Custom Player Skin**
Description: As an gamer I want to be able to customize my player sprite so that the game experience feels more personalized and fun.
Implemented: No.
Acceptance criteria:
    - On the options page a user should be able to customize their player appearance or unlock new player skins.

29. **Restart game**
Description: As an gamer I want to be able to restart a game at any point so that I can continue playing after losing a round.
Implemented: Yes.
Acceptance criteria:
    - The user should be able to be redirected to the start menu page where a new game can be set.
    - Pressing the key m will do this.

30. **Multiple Lives**
Description: As a user I want more than one life if I play on easy mode so that I can keep on playing at my pace if I hit an obstacle.
Implemented: No.
Acceptance criteria:
    - The orange player should have three lives.
    - The user should be able to see how many lives they have left in the game.

2.2 Definition of Done

All User Stories are completed when they fulfill the following criteria:

- All code related to the User Story's tasks compiles and runs.

- Any code related to the User Story with accessibility more than private is documented with JavaDoc.
- All code related to the User Story's tasks is reviewed and accepted by another group member.
- All code related to the User Story that's in the Model has and passes JUnit tests with 100% code coverage.
- The written code should follow the UML-diagram as planned.
- All branches related to the User Story are merged into the main branch.

2.3 User interface

The game is made to be user friendly for all audiences so the design is very simple and follows a common space theme. The game has a uniform design across all screens. It is made to fit all screens and all interactive buttons have pliancy. A first draft of the user interface was sketched on an iPad and looked like this.

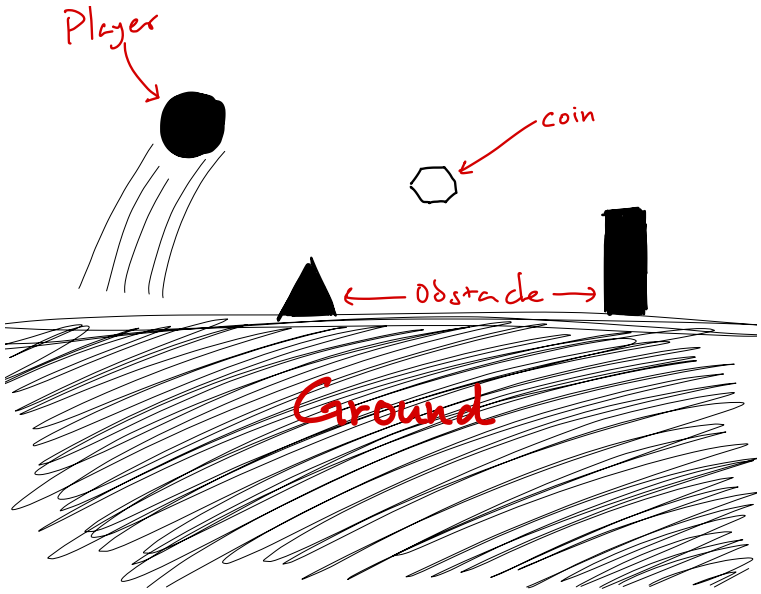

Figure 1: An original sketch of Alien Run.

It was later on decided upon that the game environment would be in space and the player sprite should be an Alien, hence the name Alien Run. Later on some designs were made in Figma and the GUI was implemented with that design. When the user starts the game they're welcomed to the game menu which looks like this.

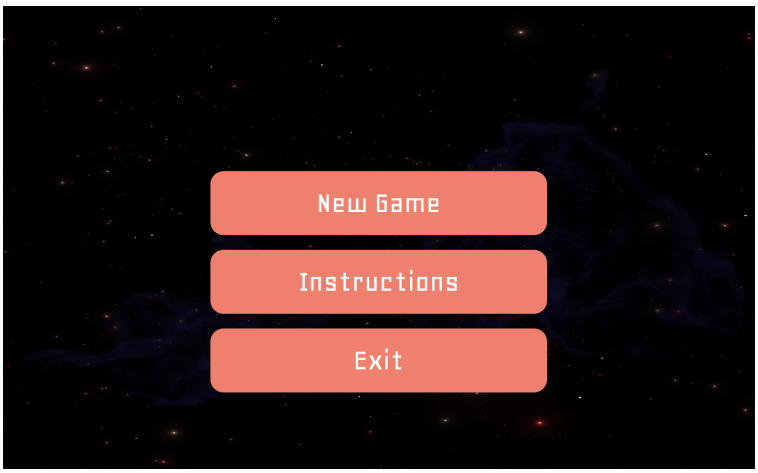

Figure 2: Starting screen of Alien Run, the game menu.

From there on they can choose to click the buttons New Game, Instructions and Exit. Exit will quit the application. If the player hovers over any of the buttons they will appear to be highlighted for pliancy. This communicates to the user that they are interactive buttons. If the user clicks on Instructions they will be taken to this screen.

Figure 3: The instructions screen.

If they click on the back button they will be taken to the menu again. If they click Start Game they will be taken to this screen where they will have to choose their player and the difficulty level of the game.

Figure 4: The player will have to choose a game mode on this screen.

After they've chosen their player the game will start, which looks something like this. The obstacles and gems are randomly generated so it won't look exactly like this every time.

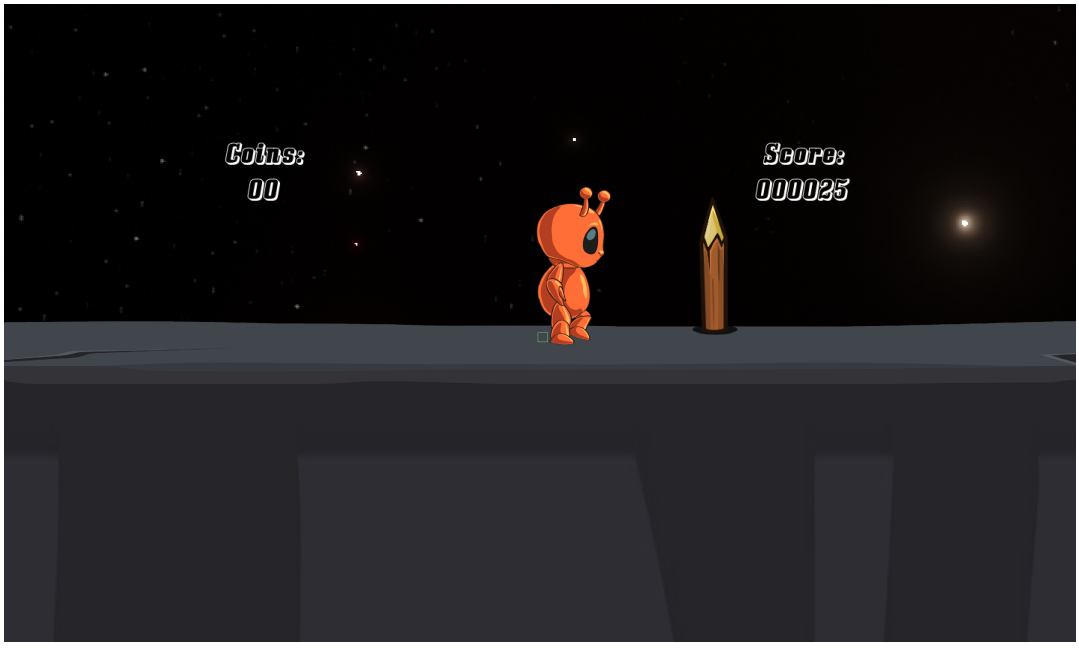

Figure 5: The main game view.

When a player touches an obstacle they have lost and Game Over is written on the screen.

Figure 6: The player has lost the game.

3 Domain model

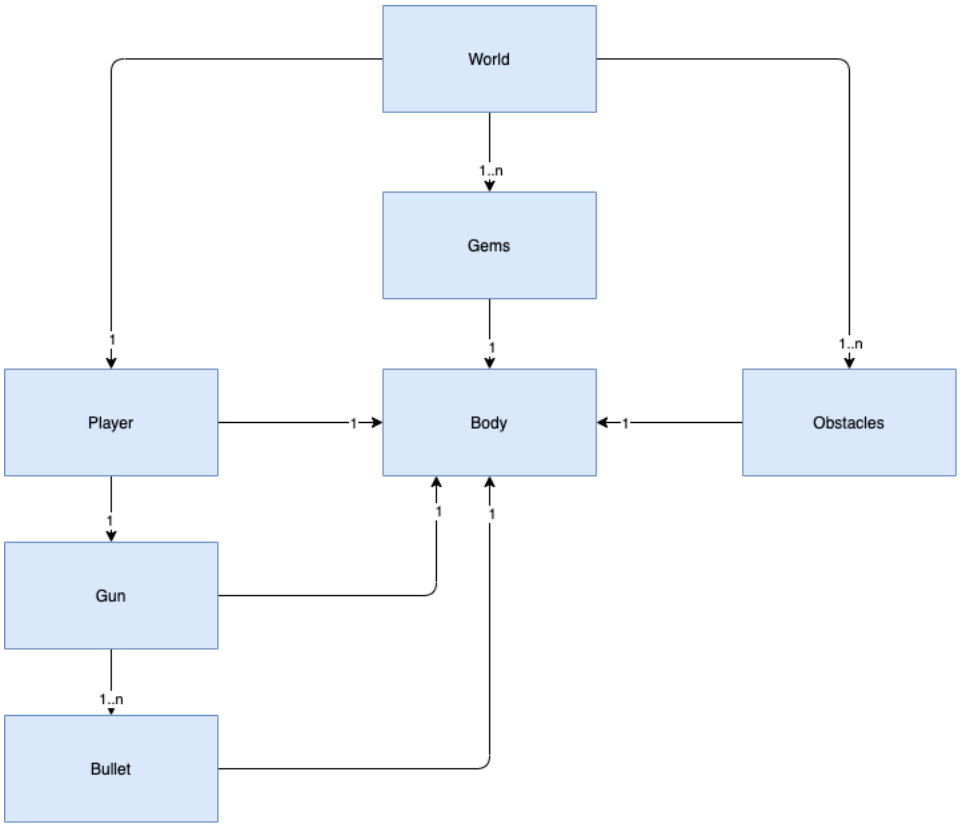

Figure 7: A UML of the domain model.

3.1 Class responsibilities

World

The World class is responsible for the logic of the whole game, it contains the player and the handlers that create the gems and the obstacles.

Player

The player classes are responsible for all the logic and creation of the three different kinds of players and their states in the game. They are also responsible for how the physics of the game feels for the user during the different game modes.

Gems

The gems classes consist of the three different kinds of gems one can collect in the game. These classes hold all the logic for the gems. E.g. they should disappear when they make contact with the player and not appear on top of obstacles.

Obstacles

The obstacle classes are a group of three different kinds of obstacles a user can maneuver in the game. They all hold the logic of the obstacles, and one obstacle class also holds the responsibility of making an obstacle disappear when it has made contact with a bullet from the gun.

Gun

The gun class is responsible for the logic and creation of the gun. This gun belongs to the player and follows its movement throughout the game.

Bullet

The bullet class shoots off from the position of the gun and disappears if there is contact made by a destroyable obstacle.

Body

All objects, that is, player, obstacles, gems, gun and bullet consist partly by a body by composition. A singular object has a singular body. The body class is responsible for the position and all the movement and forces that act upon these objects.

4 References

The following tools and libraries were used in the making of this project:

Tools

- IntelliJ
- Git and Github
- Trello
- Figma
- Draw.io
- Google Drive
- Discord

Libraries

- LibGDX
- JUnit