Bilkent University

Department of Computer Engineering

Object Oriented Software Engineering Project

Space Out: Retro sci-fi platformer game

Analysis Report

Doruk Çakmakcı, Anıl Erken, Umut Berk Bilgiç, Arda Can Atasoy

Course Instructor: Bora Güngören

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1. Introduction

Space Out is retro style sci-fi platformer game that anybody can use to "space out" from their boring life. It is implemented using Java. The game contains three distinct worlds each equipped with distinct features such as varying gravity, different block textures, enemies, surfaces etc. Each world has two levels in which the character progresses through. Even though the game is not completely story-driven, there is a sense of story provided with continuity throughout level and world endings. The goal of each level is to reach the finish by running through the platforms, destroy enemies and getting powerups as you do so and jumping over obstacles and traps. The character will be controlled by the arrow keys.

The game contains several features such as a level maker, in-game achievements, local and cloud based global leaderboards and various character skins unlocked through the achievement system. It aims to deliver a user friendly gameplay experience.

This report will include analysis of the implementation of said features and objectives, explain gameplay elements and rules, functional and non-functional requirements and system models.

2. Current System

Since the current system does not exist as of now there is no content to include in this section.

3. Proposed System

The game is a sci-fi based, platformer which consists of three different worlds and two distinct levels in each. The player runs and jumps through the platforms and obstacles, kills the enemies by jumping on them in Earth, Moon and Mars, respectively and reaches to finish line. During this journey, player faces with different surfaces and block textures, different types of enemies and various gravities that make it harder to pass the platforms. Player has three lives to complete each level. Player can gain power-ups by beating the enemies. And based on pace of level completion, player can unlock in-game achievements and get a high rank in global leaderboard. As a result of these, different character skins will be unlocked. For more entertainment and creativity, player will be able to make his/her own level by dragging and dropping obstacles, platforms and enemies. So that player can design his/her own game and play however he/she wants.
Player can stop the game, return to main menu, exit game or continue playing. The game character will be controlled by keyboard.

3.1 Distinct Worlds

The story begins in Earth. By beating enemies and passing obstacles, the character first reaches to Moon. After some adventures in Moon, character reaches to Mars and challenges continue there. In all distinct worlds, player has to cope with different enemies, obstacles and varying gravity.

3.1.1 Earth

Earth is the start point of the game. There, player tries to beat wild animals, jump over obstacles and escape the traps that were designed by other people. If player is capable of completing two levels of Earth, then he/she is able to fly to the Moon. Here are the textures that form the Earth in the game:

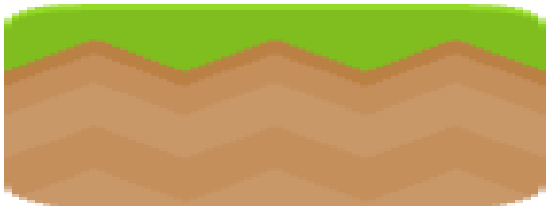


Image 1 – Earth Surface [1]

Image 2 – A trap block on Earth [2]

Image 3 – Enemy animal on Earth [3]

Image 4 – A trap on Earth [4]

3.1.2 Moon

In the Moon, main challenge is gravity. Since gravitational constant is only 1.6 there, controlling the character will be harder for the player. If two levels of Moon completed successfully, character will finally reach to Mars. Here are the textures that form the Moon in the game:

Image 5 – Moon surface [5]

Image 6 – An obstacle on Moon [6]

3.1.3 Mars

Mars is the last level of our game. If this level is also completed, user can deserve a position in leaderboard. Here, gravitational constant is 3.7. Compared to Moon, this gravity easier to handle and even can be an advantage since player can jump and fly over many things quickly. But there will be enemy aliens to fight with. Therefore player needs to beat them in order to complete the game. Here are the textures that form the Mars in the game:

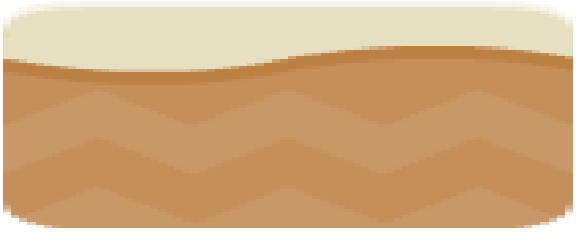


Image 7 – Mars surface [7]

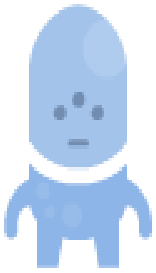


Image 8 – An enemy on Mars [8]

3.2 Leaderboard, Achievements and Unlocks

The player performance will be measured with the speed of completion of all levels. There will be a cloud based, global leaderboard in the game. Player can have a rank in the global leaderboard if he/she is successful enough, and can view best performances around the world. Also there will be a local leaderboard that player will be able to view his/her own best performances. Based on the rank on global leaderboard, player can unlock different character skins. The player on top of global leaderboard will unlock a special character. Another way to unlock character skins is reaching in-game achievements. These achievements and character skins that can be unlocked and used are listed below.

3.2.1 Achievements List

Although the full list of achievements is not yet decided on, we aim to produce a high number of achievements with varying difficulties and rewards, which will increase the game's replay-ability drastically. Here are some example achievements:

Survivor: Complete Earth without dying in one run.
Gotta Go Fast: Complete Earth without dying in one run under 5 minutes.
Intergalactic Conqueror: Complete all the story missions.
Defiler's End: Defeat the final boss on all levels on Mythic difficulty.
What Doesn't Kill Me: Die a total of 500 total deaths.
Not This Again: Fall into the trap of an enemy 100 times.
In a Galaxy Far Far Away: Run a total of 100 kilometers.

3.2.2 Character Skins List

We will draw or acquire these in a later date.

3.3 Power-ups

Player can earn power-ups and increase his/her performance by killing the enemies. The power-ups include extra live and extra speed.

3.4 Level Maker

Player will be able to play six levels in three different worlds. But also he/she can use his/her creativity and build his/her own level however he/she desires, and get exactly what he/she wants. To do this, player should click "Make Your Level" option in main menu and then rest is drag and drop by selecting platforms, enemies and obstacles.

3.5 Functional Requirements

Since Space Out is a Retro-style platform game, the player will only be able to move the main character by left and right arrow keys and the space keylocated on keyboard. Also, the arrow keys will give a constant acceleration instead of constant velocity to the main character. This aspect will help the finish times of different levels and players be divergent.

User will be able to change the character skins(for which, the player must have fulfilled the corresponding achievement for the character skin), increase/decrease volume level and change account settings from change setting option.

Space Out has 2 distinct leaderboard systems. One of which is global, cloud-based leaderboard which contains a list of all players that played a particular level and finished successfully. Players are listed according to their finish times. The other leaderboard system is local and lists different finish times of various trials for an episode.

Space Out offers different obstacles and power-ups during the game (i.e. spikes, traps, enemies as obstacles and double jump, speed etc. as power-ups).

Player will be able to look to achievements by pressing the achievements button located at the main menu The prizes for different achievements will not be displayed however, the player will be able to see if earning the corresponding achievement will unlock a character skin.

3.6 Non-Functional Requirements

3.6.1 Graphical Advantages

As it's known, bulk part of platformer games have animations and smoothness in the character and enemies motions. Thus, best animations have been selected for graphics since both of them provides a smooth gameplay to the user. Space Out is built on modernized pieces such as bricks, appearances of enemies and characters and background. And also animations and motions rendering is tried to work very smoothly in the gameplay of Space Out.

3.6.2 Game Performance

Space Out has been designed with respect to get high performance in order to work with each user's computer easily. To be able to achieve this, we try to use components which requires low system requirements. However, these component are also appropriate for modern and smooth gameplay.

3.6.3 Interface

Arrangement of the buttons is chosen with respect to ease of use and keeping the learning curve low therefore the arrow keys have been selected for the main controls. Also, we have made sure that new features of the game and labels which are changing such as scoreboard and life bars do not distract player's concentration but still provide accessible information with a quick glance. And we are sure that focusing on user-friendly interface will attract more users to Space Out.

3.6.4 Extendibility, Maintainability, Compatibility

These three concepts should be well-designed in today's computer programs. Java is used in the implementation of Space Out, this makes the game compatible with a very wide range of computer systems. And also, objects in the game have been designed to be able to have less common properties and features, it will be advantageous for the maintainability of Space Out. When one part does not work properly, its effect of the whole game will be less and it will be easier to handle it. In terms of extendibility, Space Out has three different worlds and each one can be extended in many ways and users' suggestions will be taken into consideration during decision-making.

3.6.5 Legal

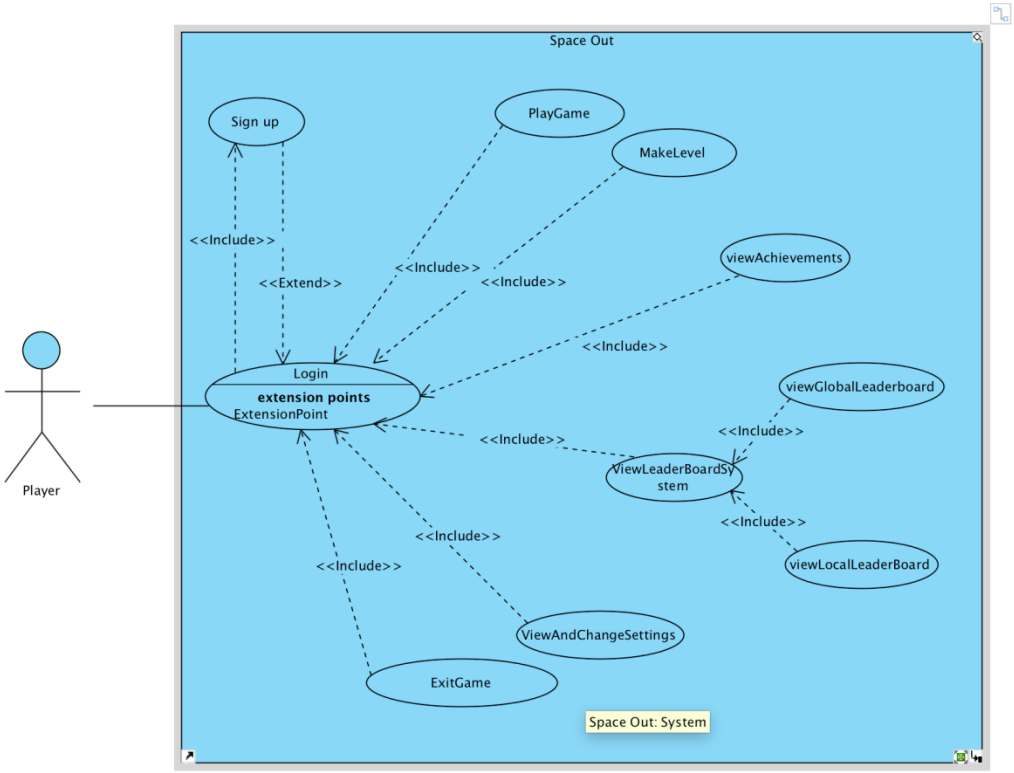
Copyright and fair use is a big contemporary debate. We intent to use copyright-free or properly licensed material, and include the authors of any tools, materials or producing bodies in our credits and license files.

3.7 System Models

Will be added in a later date.

3.7.1 Use case model

This section provides the main, general use case model of Space Out. Use case scenarios are included below.



Use Cases will be numbered as #1 ,#2, etc. The Figure above will be divided into different use cases.

Use Case #1

Use Case Name: Sign Up
Participating Actors: Player
Entry Condition: Player has opened the game.
Main Flow of Events:

1. Player starts the game.
2. The system displays login page.
3. Player presses the sign up button.
4. Player fills the accountName and password fields present in sign up page.
5. Player clicks the sign up button.
6. System creates the account and displays a success message.

Alternative Flow of Events: Steps are same until step #6. This flow happens when there is already an account with the same name that was entered by the Player. In this case system fails to create an account and displays a failure message.
Exit Condition: Player returns to the login page.

Use Case #2

Use Case Name: Login
Participating Actors: Player
Entry Condition: Player has opened the game.
Main Flow of Events:

1. Player starts the game.
2. The system displays login page.
3. Player fills the account name and password fields present in login page.
4. Player presses the login button.
5. System checks for the correctness of the information entered. If the information is correct, System grants access to the account

Alternative Flow of Events: Steps are same until step #4. This flow happens when the information entered to account name and password does not match. In this case system prompts the user to check the correctness of the information entered.
Exit Condition #1: Player is directed to main menu page (this condition is satisfied when the event flows as stated as main flow of events).
Exit Condition #2: Player stays in log in page (this condition is satisfied when the event flows as stated as alternative flow of events).

Use Case #3

Use Case Name: PlayGame
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the play game button
4. Player selects a world.
5. Player selects an episode from the selected world.
6. Player finishes the episode with life remaining greater than 0. Player's enemy body count and timing is displayed and saved to global and local leaderboards.

7. Achievement fulfillment is checked to see if the player satisfied all of the conditions for an achievement.
8. Player is rewarded with any achievement that was completed.
9. Any reward from the achievements are claimed by the user.

Alternative Flow of Events: Steps are same until step #5 and after step #7 (inclusive). There are different alternative flows to this is use case and each of them leads to incomplete level. This flow happens when the character is killed (life remaining = 0) by any object (enemies, traps, spikes) or action (fall to void).
Exit Condition : Player is directed to main menu page.

Use Case #4

Use Case Name: MakeLevel
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the make level button
4. Player selects a world.
5. Player fills the empty canvas with different blocks, enemies, traps or spikes stated at overview section of this report.
6. Player is required to put start and finish blocks to specify boundaries of the level.
7. Player presses the finish level button.
8. Player names the level he/she just created.

Exit Condition: Player is directed to main menu page.

Use Case #5

Use Case Name: View Achievements
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the View Achievements button.
4. Player presses the back button to return main menu.

Exit Condition: Player is directed to main menu page.

Use Case #6

Use Case Name: ViewLeaderBoardSystem
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the LeaderBoard button
4. Player selects a either global or local leaderboard to view.
5. Player presses the back button to return main menu.

Exit Condition : Player is directed to main menu page.

Use Case #7

Use Case Name: ViewOrChangeSettings
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the settings button.
4. Player can select to change character skins, change volume level, change account settings.
5. Player presses the back button to return to main menu

Exit Condition : Player is directed to main menu page.

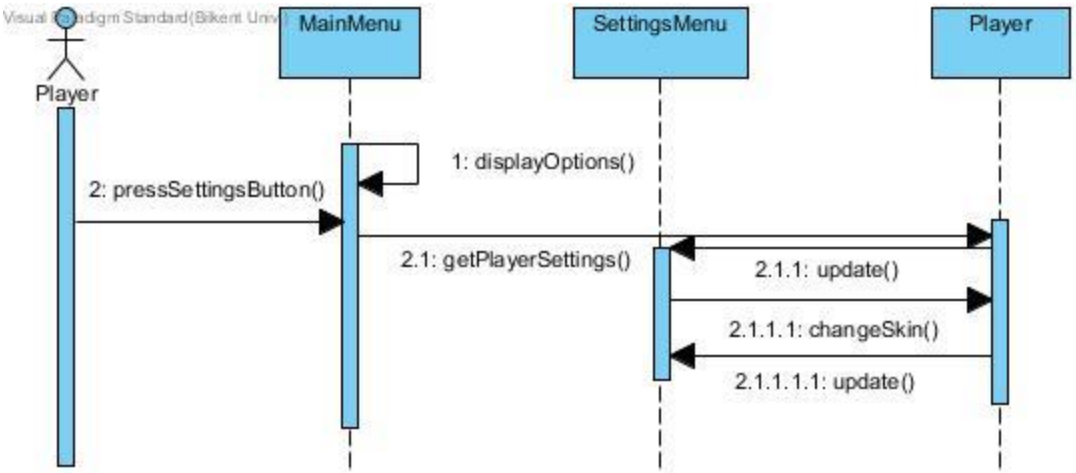
Use Case #8

Use Case Name: ExitGame
Participating Actors: Player
Entry Condition: Player has opened the game and logged in to any account successfully.
Main Flow of Events:

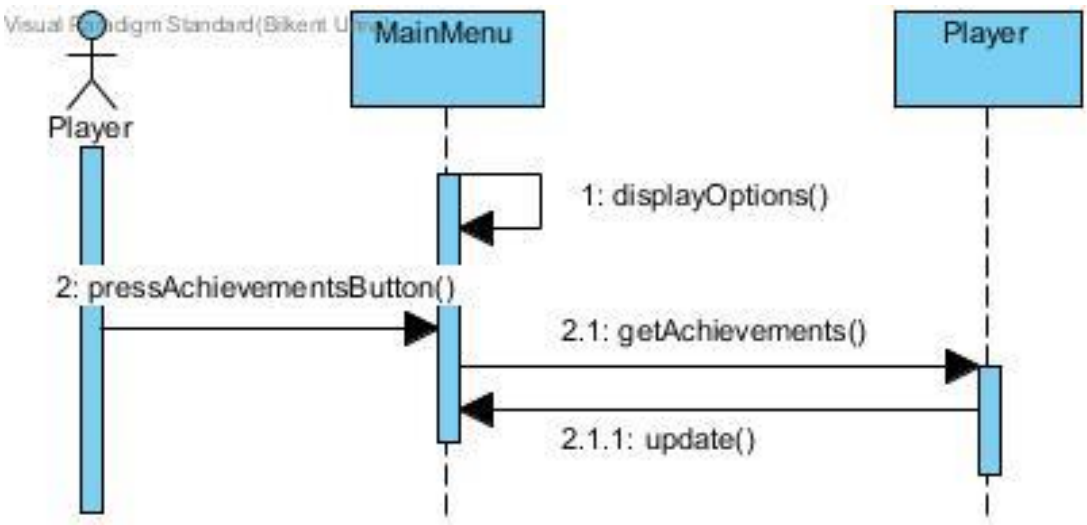
1. Player starts the game and logs in successfully.
2. The system displays main menu.
3. Player presses the exit button

Exit Condition : Player leaves the game

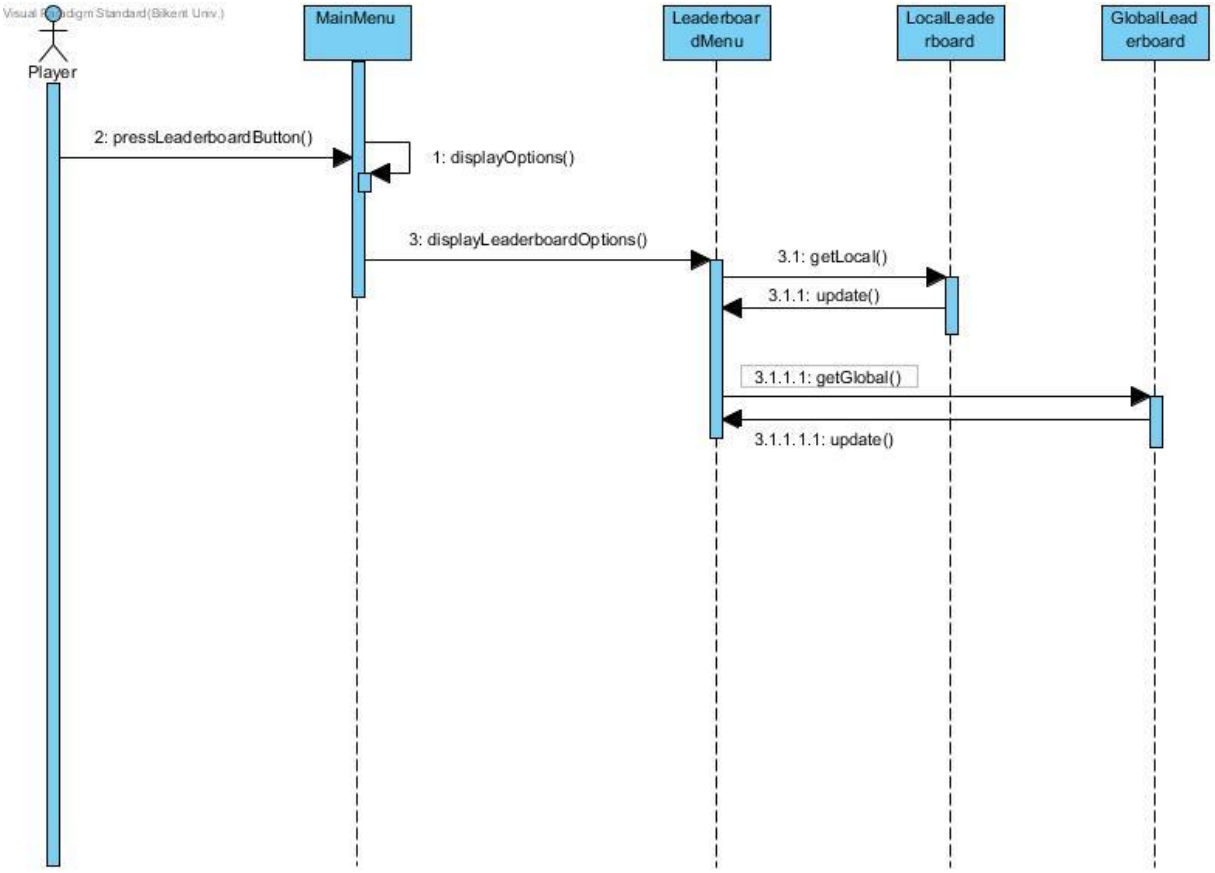
3.7.2 Dynamic model



Model for the viewAndChangeSettings use case



Model for viewAchievements use case.

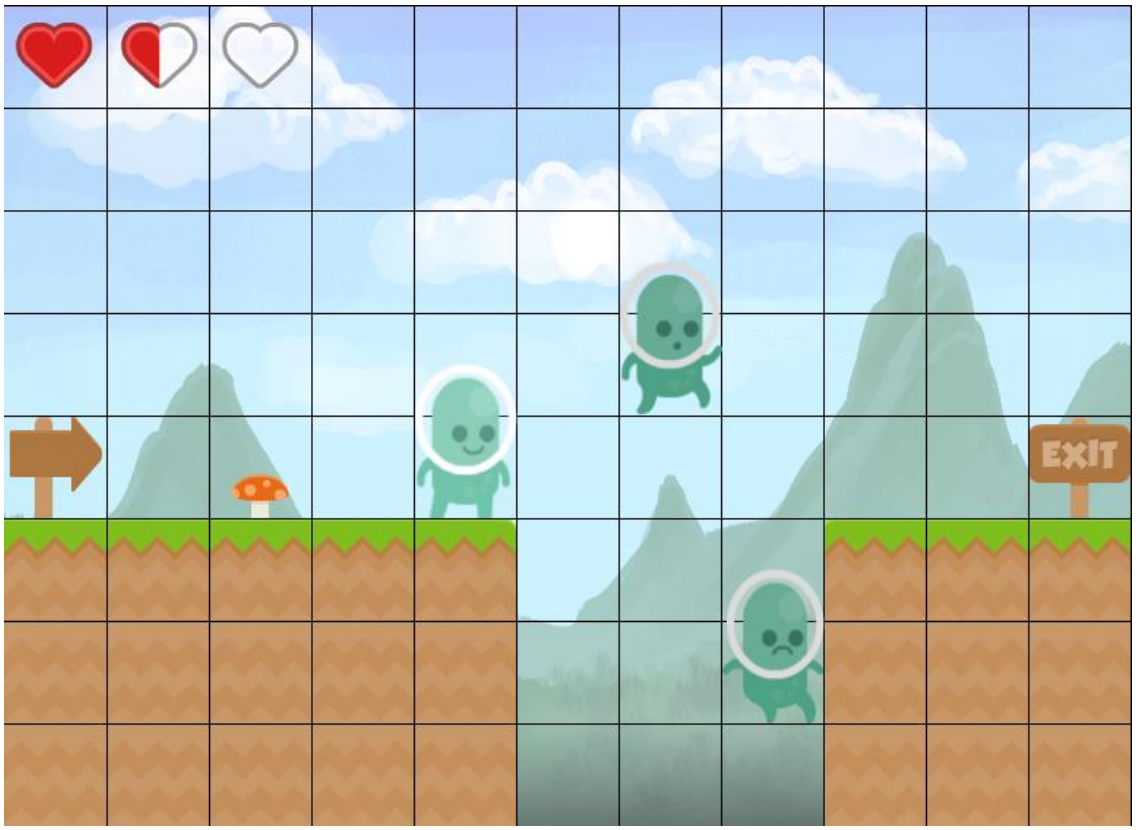


Model for viewLeaderboardSystem use case.

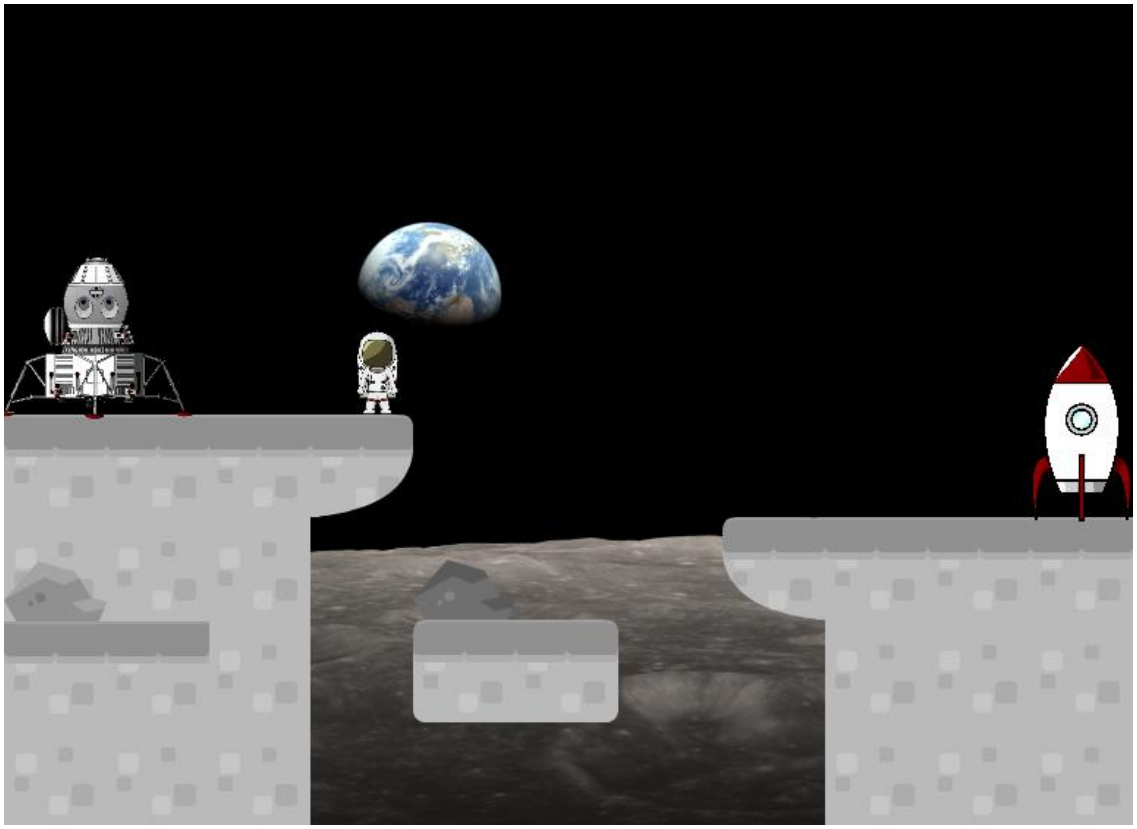
3.7.3 Object model

3.7.4 User interface and screen mock-ups

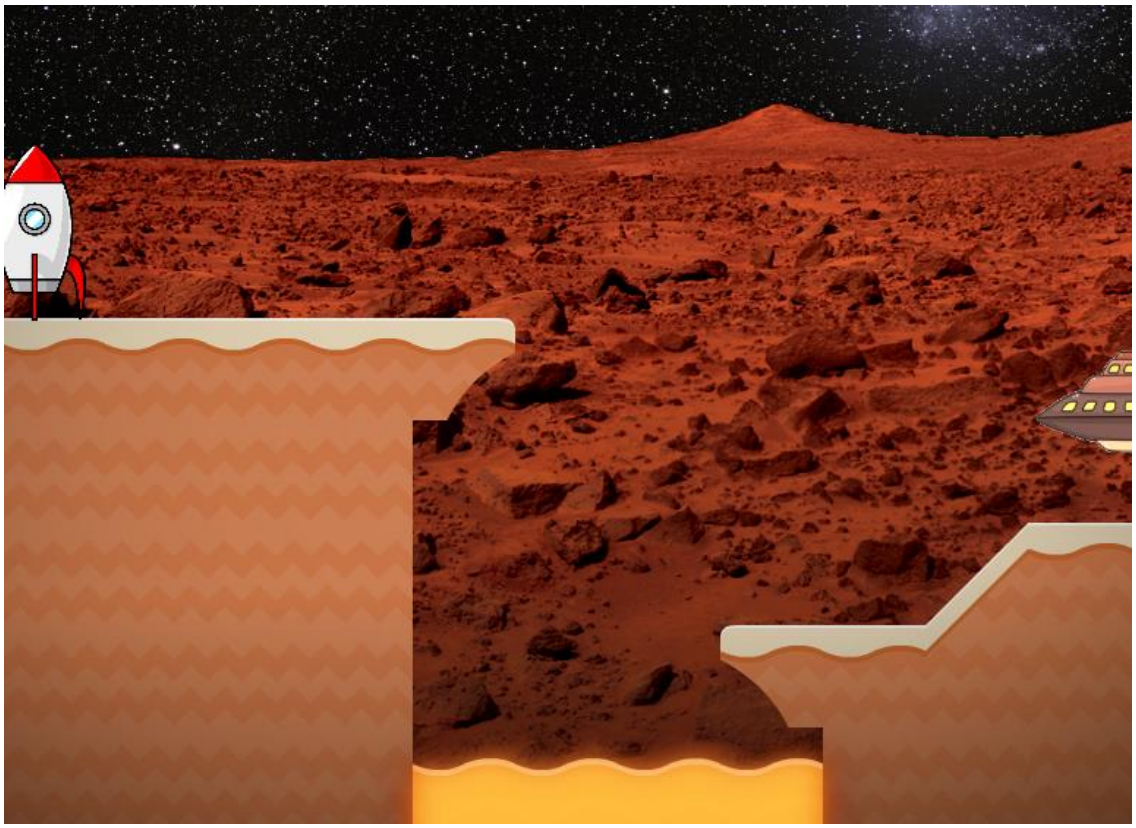
The start screen menu. Vapor-wave and sci-fi themed fonts and color choices. Simple and through menus.



Level mockup for Earth, tile grid is visible.



Level mockup for Moon, tile grid is invisible.



Level mockup for Mars, tile grid invisible.

Pause screen mockup. The level is blurred in the background for a sense of layering.

Leaderboards and settings mockup. UI elements and overlays use blur to create a sense of depth and hierarchy.