

CS 319 - Object-Oriented Software Engineering  
**Analysis Report**  
  
**Run, Dot Run!**  
Group 2-H  
Beyza Tuğçe BİLGİÇ

Gökalp KÖKSAL

Gökçe ÖZKAN

Emine Ayşe SUNAR

Table of Contents

[1. Introduction 1](#_Toc310454890)

[2. Overview 1](#_Toc310454891)4

[3. Requirement Specificaiton 6](#_Toc310454897)

[3.1. Functional Requirements 6](#_Toc310454898)

[3.2. Non-Functional Requirements 7](#_Toc310454905)

[4. System Model 7](#_Toc310454910)

[4.1. Use Case Model 12](#_Toc310454911)

[4.2. Dynamic Models 15](#_Toc310454917)

[4.3. Object and Class Model 16](#_Toc310454923)

[5. User Interface 17](#_Toc310454924)

[5.1. Navigational Path 17](#_Toc310454925)

[5.2. Screen Mock-ups 17](#_Toc310454926)

[6. References 38](#_Toc310454933)

1. **Introduction:**

In this project, we are going to develop a game called “Run Dot Run”. There will be a dot as the main character of the game and the goal of the dot is to reach the end of a sentence by passing letters in a limited time.

The game will include different levels, which are to be locked as the previous levels are completed successfully. In each level, the length of the sentence to pass will increase. In addition to this, each level will come with different kind of obstacles.

The game will be a desktop application. The user will control the game with keyboard. The game is for any user at any age.

1. **Overview:**

Run Dot Run is a game in which the user controls a dot and tries to pass over letters of a sentence and reach to the end of that sentence in a short duration. Only the dot will be user-controlled. During the game, user will move the dot by pressing spacebar, right and left keys on the keyboard. S/he will press left key to move the dot left; right key to move right and spacebar key to jump. As the user press spacebar and one of the direction keys at the same time, the dot will make a projectile motion, regarding which direction key(left or right) is pressed with spacebar. Otherwise, the dot will stay at its current position.

The user will be able to pause the game. Because in the game there is limited time to complete the level, it might be necessary for the user to pause the game to be able to continue playing after a while.

* 1. **Levels:**

The game will include 4 different levels. In each level, there will be longer sentences than those in the previous level. However, the proportion of time over the length of the sentence will decrease, which will make it harder to reach the end in the given time and the user will need to make less mistakes. It is relevant for all the levels that if the dot cannot pass the obstacles or the letters before the time is up, it will start from the beginning position and try to finish the game in the remaining time. As the level is completed successfully, the next level will be unlocked. User will be able to select any unlocked level to play in.

* + 1. **Level 1:**

In the 1st level, the letters of the sentences will have such a font that their upper surface will be flat. So, the dot will not fall from the letter unless it falls to the ground while it is jumping. In other words, the dot will not slip over the letters. Because this will be the easiest level, it will be like a warm up level. Therefore, the only difficulty will be to pass the letters and user will try to reach the end only without falling between the letters.

Rules for level 1:

* Do not drop the ball to the ground until the end of the sentence.
* Reach the end before time is up.
  + 1. **Level 2:**

In this level, the font of the letters will vary, so that the user will need to be careful in order not to make the dot slip over the letters having round edges. The sentence will be longer than in the 1st level.

Rules for level 2:

* All rules in the first level are relevant for this level
* Also be careful about the letters with round edges so as not to slip over them and fall
  + 1. **Level 3:**

In addition to the features of the 2nd level, different obstacles will come with 3rd level. The obstacles will be eraser and spike.

1. **Eraser Obstacle:**

During the game, there will be erasers coming from different directions. Some of them will be thrown from the right side, some of them will be falling from the sky and some of them will be moving upward from the space between the letters. So, all the erasers will have a direction that passes from the path of the dot. The user will try not to collide the dot with the erasers. Otherwise the dot will be erased and it will start from the beginning position. Same as if it falls to the ground, it will try to complete the level starting from the beginning but in the remaining time.

1. **Spike Ostacle:**

In addition to the eraser, there will be spikes on some letters in 3rd level. If the dot touches the spikes, it will be destroyed and go back to the beginning position and continue the level in the remaining time.

Rules for level 3:

* All rules of level 2 are relevant for 3rd level
* Do not collide with the erasers
* Do not touch the spikes

* + 1. **Level 4:**

4th level is the last level of Run Dot Run game. In this level, as the dot touchs a letter, the letter will disappear in 1-2 seconds. If the dot does not jump to the next letter in this short time, it will fall and restart from the beginning to complete the path within the time left. Therefore the user should always be active and should move the dot without waiting. This will increase the possibility to make mistakes such as falling between the letters, colliding with erasers or touching spikes.

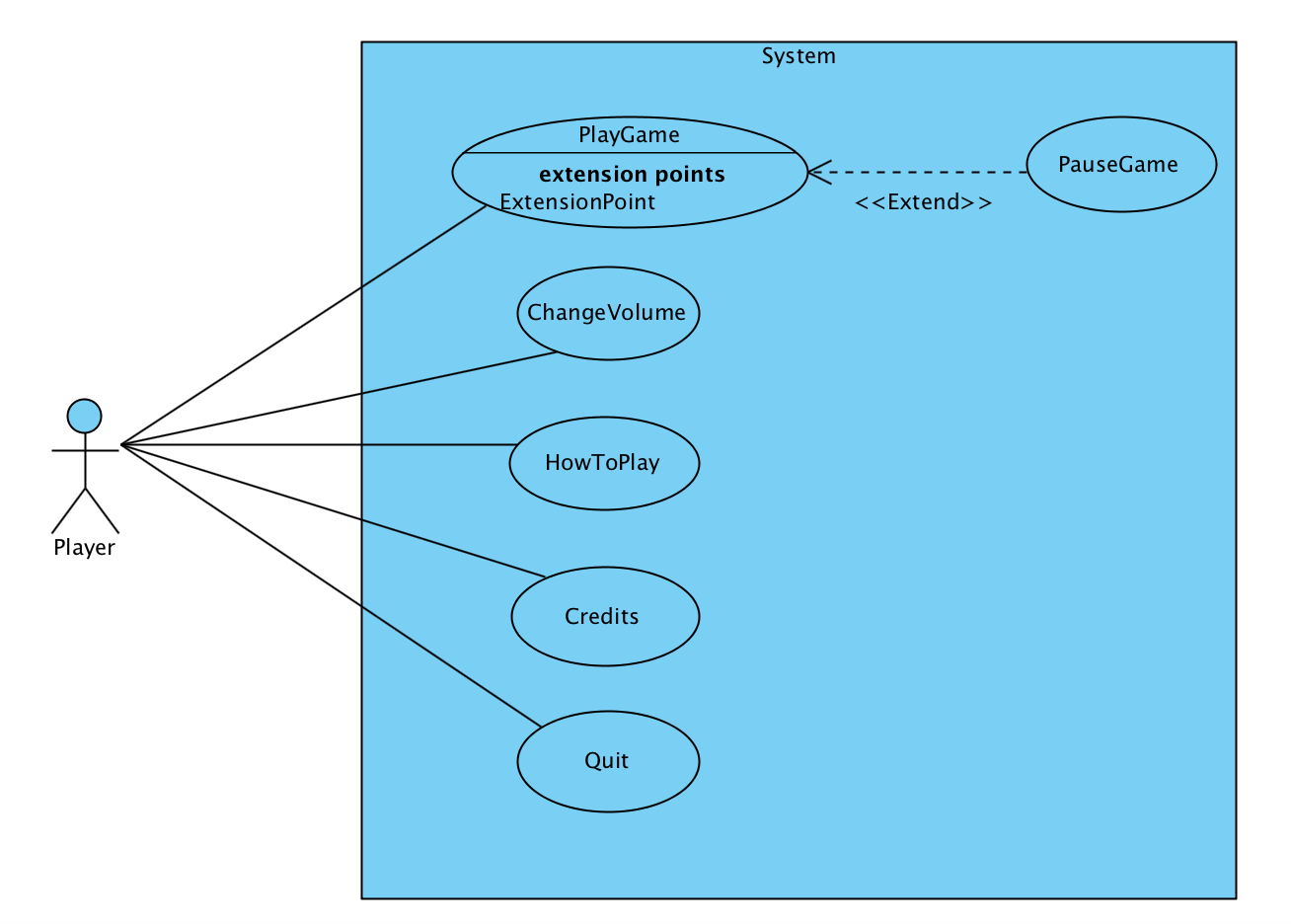
Rules for level 4:

* All the rules up to this level will be relevant for this level either
* Be careful about the time in which the letters disappear as they are touched, in order not to fall to the ground

1. **Functional Requirements**
   1. **Functional Requirements**

* **Moving the Dot:** The user will be able to move the dot using the keyboard. To move the dot right and left, the user will simply use the right and left keys and to jump, the user will use the spacebar. There will be a double jump feature to enable the dot to jump over larger obstacles. To double jump the user will click the spacebar twice.
* **Selecting Levels:** The user will be able to select a level among 4 levels with different difficulties. In each level a different obstacle will be introduced. The levels will be as the following:
  + - Level 1: There won’t be any obstacles, the user will simply jump between the letters without falling to the ground to win the game. The letters will have sharp edges to enable ease in the game.
    - Level 2: The letters will have round edges, thus the dot may slide from the edge of a letter.
    - Level 3: An eraser will be thrown in the horizontal axis, if the dot collides with the eraser, it will start from the beginning. Also, some spikes will be added on top of the letters. If the dot touches these spikes, it will start from the beginning.
    - Level 4: Some of the letters will disappear after a short amount of time when the dot touches them.
* **Moving Screen:** The screen of the game will move as the dot moves right and left.
* **Pause Game:** The user will be able to pause the game. The user may press resume and continue to the game or could press quit and return to the main menu of the game.
* **Time:** There will be a time limit in the game. The user must try to finish the game during the given time limit.
* **View Help:** The user will be able to get help by clicking the “?” button in the main menu.
* **Set Volume:** The user will be able to set the volume of the game.
* **View Credits:** The user will be able to view the credits. In the credits, the names of the developers will be listed and some of the citations used could be listed also.
  1. **Non-Functional Requirements**
* **Friendly user interface:** The user interface will be simple and easy to understand. All labels on the buttons will be straightforward, explaining the button’s functionality.
* **Extensibility:** The game will be designed in a way that it will accept updates without changing the existing design.
* **Smooth Graphics:** The game will be designed in a way to minimize lagging and thus creating a smoother gameplay.
* **Animations:** Some animations will be added to the game to make it fun to play.
  1. **Pseudo Requirements**
* The game will be implemented using java.
* Some pictures will be created using Photoshop CC 2015.

1. **System Model**
   1. **Use Case Model**

****

**Use Case Descriptions**

Use case #1

**Use case name:** PlayGame  
**Participating actors:** Player  
**Entry condition:** The user has opened the game and displayed the main menu.

**Main Flow of Events:**

1. Player starts the game.
2. The level screen is shown and the game begins with first level.
3. In case of failure, each level returns to beginning of the level until player completes the level.
4. Player finishes all the levels.
5. Main menu is displayed.

**Exit condition:**

* Player has completed all the levels successfully, or
* Player has paused the game and chosen to exit the game

**Alternative Flow of Event:**

* Player can pause the game and select to exit or resume game at any time.

Use case #2

**Use case name:** ChangeVolume  
**Participating actors:** Player  
**Entry condition:** The user has opened the game and displayed the main menu.

**Main Flow of Events:**

1. Player clicks on the volume button.
2. The volume bar appears next to the button.
3. Player will be able to adjust the volume by sliding the volume bar.

**Exit condition:**

* Player presses the resume button and continues to play the game, or
* Player presses the exit button and returns to the main menu.

Use case #3

**Use case name:** HowToPlay  
**Participating actors:** Player  
**Entry condition:** The user has opened the game and displayed the main menu.

**Main Flow of Events:**

1. Player clicks on the question mark on the main menu.
2. Player views hint about how to play the game.

**Exit condition:**

* Player presses the back button and returns to the main menu.

Use case #4

**Use case name:** Credits  
**Participating actors:** Player  
**Entry condition:** The user has opened the game and displayed the main menu.

**Main Flow of Events:**

1. Player presses the credits button.
2. Player accesses to information about the developers of the game.

**Exit condition:**

* Player presses the back button and returns to the main menu.

Use case #5

**Use case name:** Quit  
**Participating actors:** Player  
**Entry condition:** The user has opened the game and displayed the main menu.

**Main Flow of Events:**

1. Player presses the quit button.
2. The game screen will be closed.

**Exit condition:**

* Player presses the back button and returns to the main menu.

Use case #6

**Use case name:** PauseGame  
**Participating actors:** Player  
**Entry condition:** Player has already started to play the game.

**Main Flow of Events:**

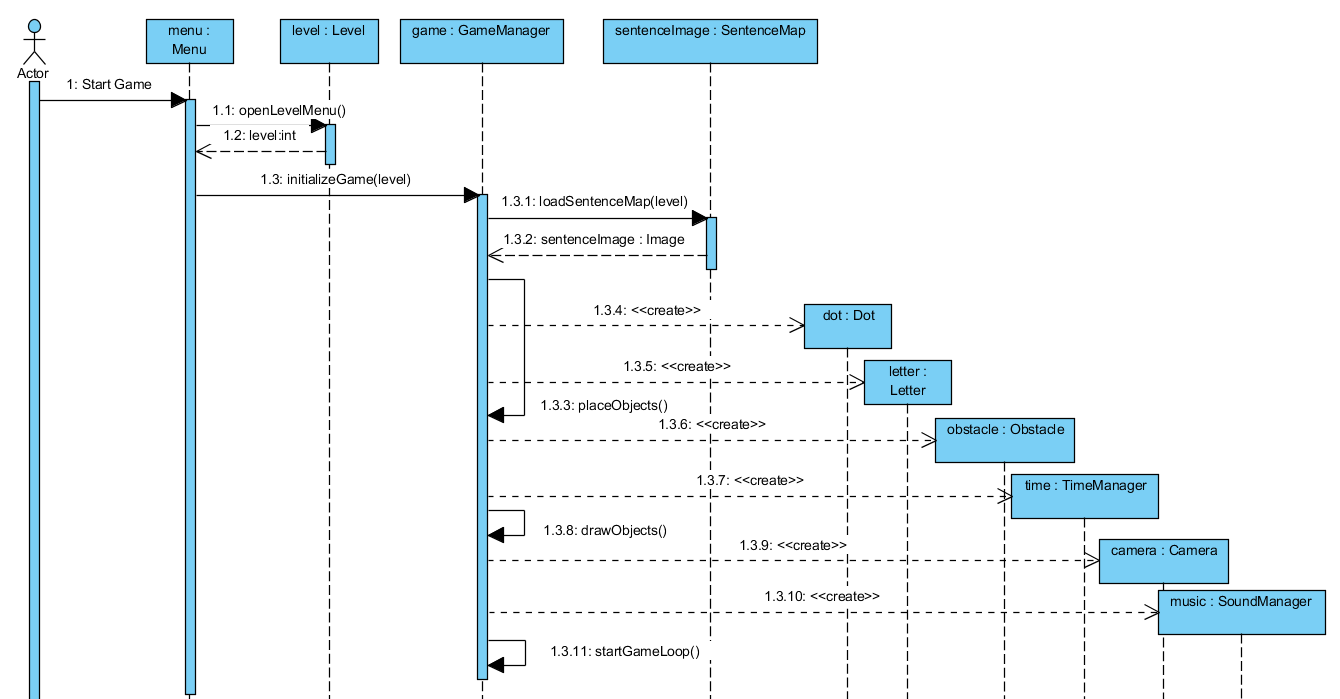
1. Player decides to pause the game and presses the pause button during the game.
2. The exit and resume options are displayed.
3. Player presses one of the buttons and continues or exits the game.

**Exit condition:**

* Player presses the resume button and continues to play the game, or
* Player presses the exit button and returns to the main menu.
  1. **Dynamic Models**
     1. **Sequence Diagrams**

**Sequence Name:** Start Game

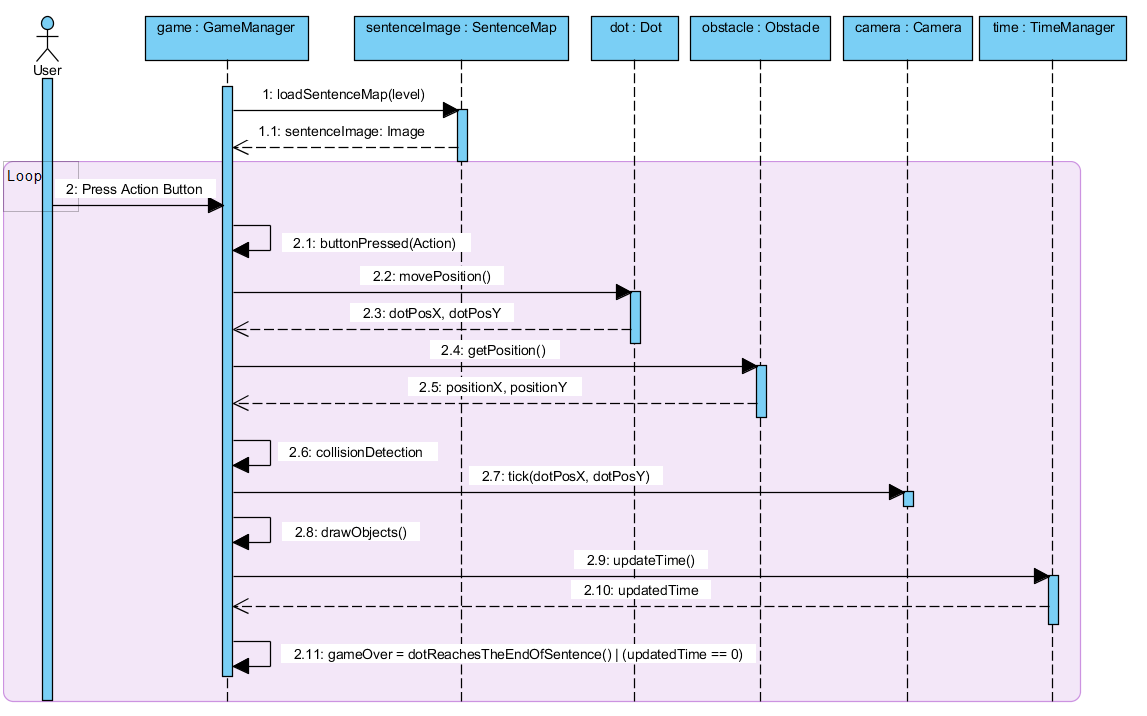
**Scenario:** Player Deniz clicks on the game icon and the main menu of the game opens. Deniz clicks on the “Start Game” button to start playing the game. Before entering the game screen, the level menu screen opens and Deniz selects an unlocked level to play. After selecting the level, the game window initializes. First, the selected level is sent to the SentenceMap class to get a random image of a sentence that belongs to the selected level. This image will be used to specify the background objects. According to the map, the place of the Dot and the letters will be decided and these objects will be initialized accordingly. Later, the obstacle and time object will also be initialized. Since all objects are placed, they are then drawn on the background. Finally the camera object will be initialized to enable the moving of the screen as the Dot moves. After all objects are initialized and are drawn, the game starts by entering the game loop.



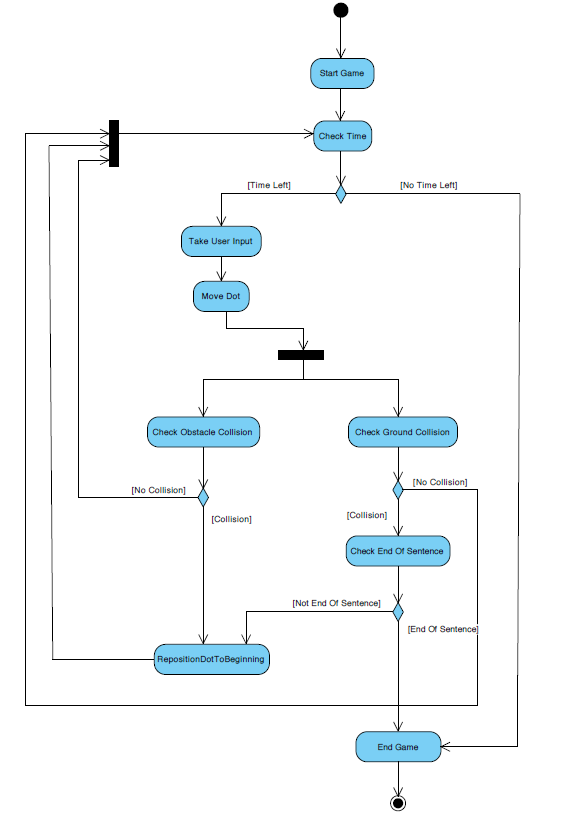
**Figure: Shows the sequence diagram for the Start Game scenario**

**Scenario Name:** Play Game

**Scenario:** The objects were placed to the background according to the image taken from the SentenceMap class. Later the game loop starts and player Deniz gives inputs to the GameManager class. According to the input taken from Deniz, the action that is going to be made will be specified in the GameManager. The Dot is moved according to the action decided and the updated position of the Dot is given to the GameManager. The position of the obstacles are also taken. Later the collision of the Dot with the ground and the obstacles are checked. The updated positions are of the object are drawn. The camera is moved according to the position of the dot. If the Dot reaches the end of the sentence or if the time reaches 0, the game will be finished.

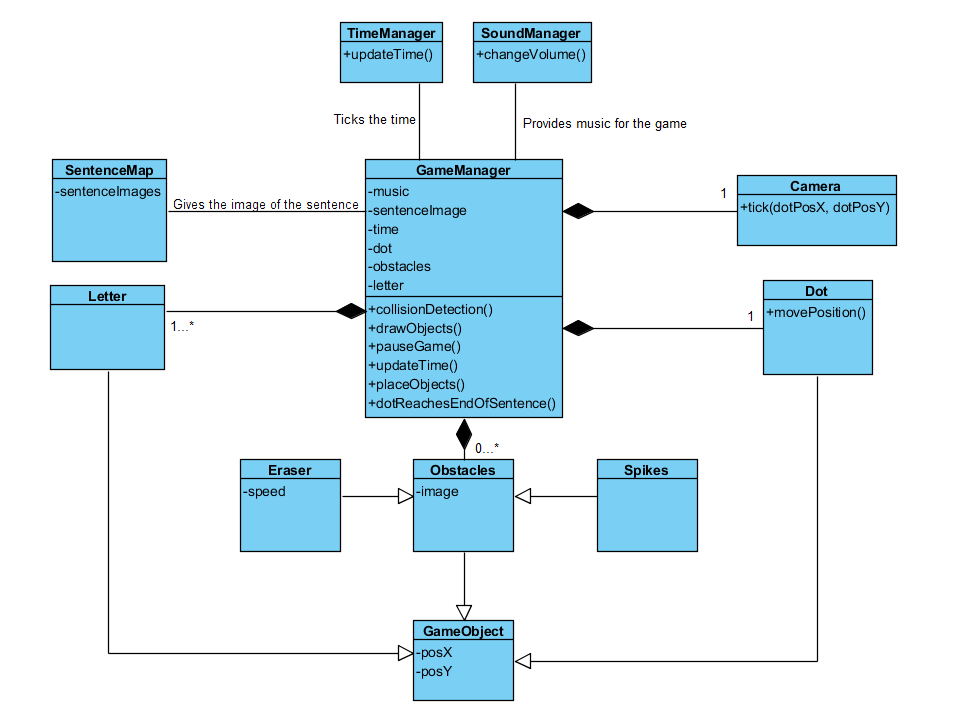
**Figure: Sequence diagram for the Play Game Scenario**

* + 1. **Activity Diagram**



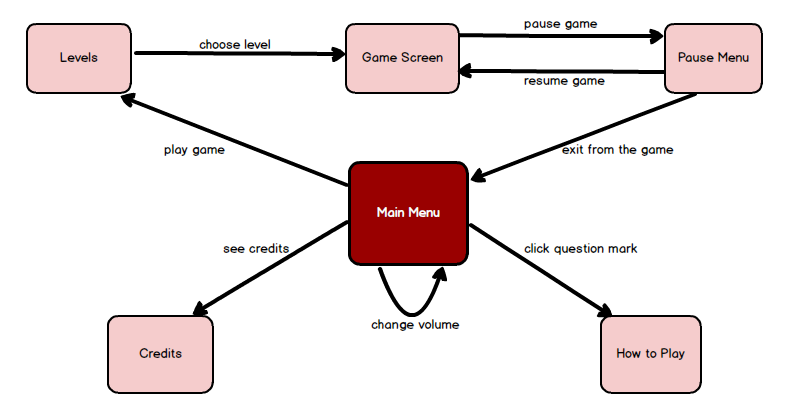
After the game starts the letters, obstacles and the Dot is placed to their initial positions. When the game starts the time starts ticking, so firstly the remaining time is checked. If the time is up the game will be over, if the time is not up the game will continue. Then the input of the user is waited. According to the input that comes from the user, the Dot is moved. After the Dot’s position is changed the collision it could be making with the ground and the obstacles are checked. When the collision with the ground is checked, in other words checking whether the Dot has fallen to the ground, if collision is not being made then the player could continue playing. So the program goes back to checking the time and continues with the loop. If collision with the ground is made, then it is checked whether the Dot has fallen to the ground at the end of the sentence. If collision to the ground is made when the end of the sentence is reached the game is won, thus the game ends. If it’s not the end of the sentence, then this means that the Dot has fallen to the ground, thus the position of the Dot will be set to the beginning point and the game will continue on until the game is won or the time is up. If collision of Dot and an obstacle is made, then the Dot will be repositioned to the beginning point. If a collision is not detected with the obstacle, the game will continue on.

* 1. **Object and Class Model**

The main class of our design is the GameManager class. This class gets the inputs of the user and moves the dot according to the input. The Dot, Letter and Obstacles are the entity objects used in the GameManager, since they are the main objects of out game.

The SoundManager class is called by the GameManager to get music for the game. The GameManager also calls the TimeManager class to update the time of the game. The SentenceMap class provides images for the GameManager and the GameManager uses these images to specify the locations of the Dot, Obstacle and Letter objects. All the Letter, Dot and Obstacles classes are the children of the GameObject class.

1. **User Interface**
   1. **Navigational Path**



* 1. **Screen Mockups**

When the game is opened, the user first sees the main menu of the game. In the main menu, there are “Play Game”, “Credits”, “How to Play” and “Quit” options.

The volume of the music which will be playing on the backround of the game can be set from the bar at the upper left corner of the main menu screen.



As the bar is moved left side, the volume will decrease. If it is moved to the most left side, the game will play in silent mode.

**Main Menu Screen**

**How to Play:**

If the use clicks the question mark button at the upper right corner of the main menu page, “How to Play” screen, where the rules of the game are explained, will open.

If the user wants to return main menu, s/he should click the arrow button at the upper left corner of the “How to Play” page. The button is shown below:



**“How to Play” Page**

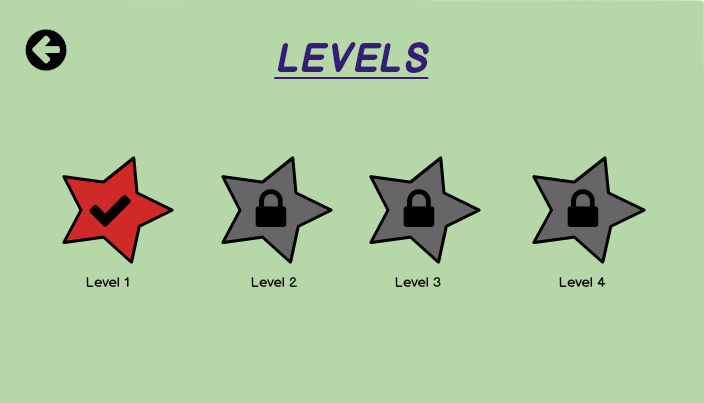


**Play Game:**

When the player clicks “Play Game” button, firstly the “Levels” screen will be shown. At the beginning, player can play only level 1 and will not be able to click the buttons of other levels. As the levels are unlocked, they will also be clickable.

If the level button’s color is red and there is a tick icon on the button, this indicates that level is unlocked.

If the level’s button is grey and there is a lock icon on it, this means that level is still locked and the button cannot be clicked.



**“Levels” page**

**Game Screen:**

After the user selects an unlocked level, the game starts. The time starts to pass immediately after the game starts.

Player can see the counter on the game screen. 

If s/he wants, s/he can pause the game.

User will control the game and ball with keyboard.



**Game**

**Pause Menu:**

Player can pause the game by pressing “p” letter key on the keyboard.

By clicking “Resume” button, s/he can continue the game where s/he left.

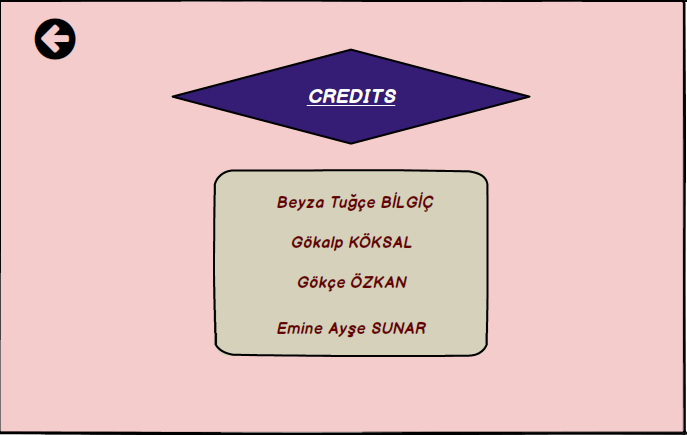
By clicking “Exit” button, s/he returns the main menu.



**Pause Menu**

**Credits:**

As the player clicks the “Credits” button, the name of the developers of the “Run Dot Run” game will be shown on the screen. In order to turn back to main menu, player should click the left arrow button.



**“Credits” page**

**Quit:** User should click on “Quit” button to quit the game.