Bilkent University

Department of Computer Engineering

CS 319
Object Oriented Software Engineering Project

Project name: Dribble and Score!

Analysis Report
25/02/2017

Group name:
2C

Group Members:

Batıkan HAYTA 21301382
Berke SOYSAL 21400908
Boran YILDIRIM 21401947
Sencer Umut BALKAN 21401911

Instructor:

Prof. Dr. Uğur DOĞRUSÖZ

Table of Contents

Analysis Report

CS319 Project: Dribble and Score

1. Introduction

The “Dribble and Score” is a game that is the combination of road-running and football-shooting games. The aim of the game is to successfully deliver the ball to near the goal by passing the obstacles in the way of character, and then shoot the ball accurately to score. This is the analysis report that includes basic gameplay elements, rules, features. Functional and nonfunctional requirements, use case models and use-case diagrams will be described.

2. Game Overview

The game will have 5 levels, each of them consisting of dribbling and shooting parts. The dribbling part of the game will be 2D-platform type. The character, which is controlled by user, will move autonomously, but the user may give these instructions by keyboard buttons: jump, move left, move right. There will be three possible paths which character can dribble. By moving left or right, character will be escaped from obstacles and by jumping character will be escaped from mud.

The second part of the game, the player adjusts the speed and direction of the ball so that he can beat the goalkeeper and score. Firstly, the horizontal target roll will move right and left without stopping and the user will try to stop it in the right place. Then, the vertical target roll will move up and down without stopping and the user will try to stop it in the right place. The intersection of those two points will show where the ball will go. Secondly, the power option will move up and down without stopping and the user will be able to stop it in the right place and score a goal.

2.1. List of Bonuses

Doping: When player gets this bonus. It enables character to gain extreme speed and also character becomes untouchable until the bonus ends. That means the obstacles won't have any effect on the character for 3 seconds.

Image 1: Doping's appearance[1]

Extra shot chance: For this bonus, player gains an extra shot chance for the second part of the game. This bonus will be visualized as a fan of the character's team.

Image 2: Extra Life's appearance[2]

Football Shoe: When player acquire this bonus, character becomes faster for 3 seconds. This will increase his score.

Image 3: Football shoes's appearance[3]

2.2 List of Obstacles

Mud: This obstacle slow down the speed of the player. Player can dodge this obstacle, if they jump , going left or right.

Referee: When player hits the referees, they get 1 yellow card. If they hit referees again they will get another yellow card and lose their life's. Player can dodge referees going left or right.

Opponent team Fans: If player encounter the opponent team fans, player has to jump to dodge them otherwise they will lose some points.

Defense players: In order to dodge defense players, player need to go left or right. If player decided to jump to dodge them, defense player will get them.

2.2.1 List of Goalkeepers

Standing Goalkeeper: This type of goalkeeper will be standing on the middle of the goal and only save the shoot if it comes exactly towards him.

Lentous Goalkeeper: Lentous Goalkeeper will start from middle and go right until the end of goal, then return and will go left, and vice versa.

Drunk Goalkeeper: This goalkeeper moves randomly.

Master Goalkeeper: This goalkeeper will be focused to the shoot and try to move to the shot's direction to catch it.

2.2.2 Soccer Player

Soccer player is the source of the action in this game. Players can move soccer player using the arrow in the keyboard or 'WASD'. Using keyboard, soccer player can jump, crouch, go left and right to avoid to obstacles and also get bonuses. Later on he has to shoot the ball accurately.

3. Requirements

3.1. Functional Requirements

3.1.1 Game Control

In dribble part, player will use W key to jump over the obstacle, s to slow down, A to go left , and D to go right. In second part of the game, player may shoot the ball by pressing at the right time to SPACE to adjust power, and direction. The bars will appear in game for shooting.

3.1.2 Score

Score system will be included and be given directly to user according to running time performance and shooting. If user used slow down option a lot in the parkour, time performance of this user will be reduced and score of the user will be decreased. If the user hits the ball at the corners, the highest score will be given. After each level is completed, the game will ask for player to input his name. Then if the player made a better score than he did before, the points he get will being add to his general score.

3.1.3 Settings

The player will be able to change the settings listed below:

1. Changing the main character's appearance.
2. Changing the input keys.
3. Resetting the highscores.

3.1.4 Help Menu

User will be able to access the help menu. Help menu contains information about game dynamics and game elements such as purpose of the game bonus descriptions and default controls.

3.1.5 Autosave

When a level is completed, the autosave function runs and saves all the user's data to the game's local database. Later, when the user re-enters the game, user can resume where left off.

3.1.6 Pause/Resume

The game can be paused at any time and can be resumed afterwards.

3.1.7 Characters

Game will contain open source 8-bit pictures. Main character, bonuses, goalkeepers and the whole level environment is going to be visualised in a retrospective way.

3.1.8 Map/Level

Maps are going to be generated randomly according to specified difficulty of the current level. Thus, each level is going to be unique and computer generated.

3.2. Non-Functional Requirements

- It will be a tile-based video game basically which will minimize the space requirements for our game “Dribble and Score”.

3.3. Pseudo Functional Requirements

- The Game will be implemented in Java.
- Some of the graphic objects will be designed using Adobe® Photoshop CS6.

4. System Models

4.1. Use Case Model

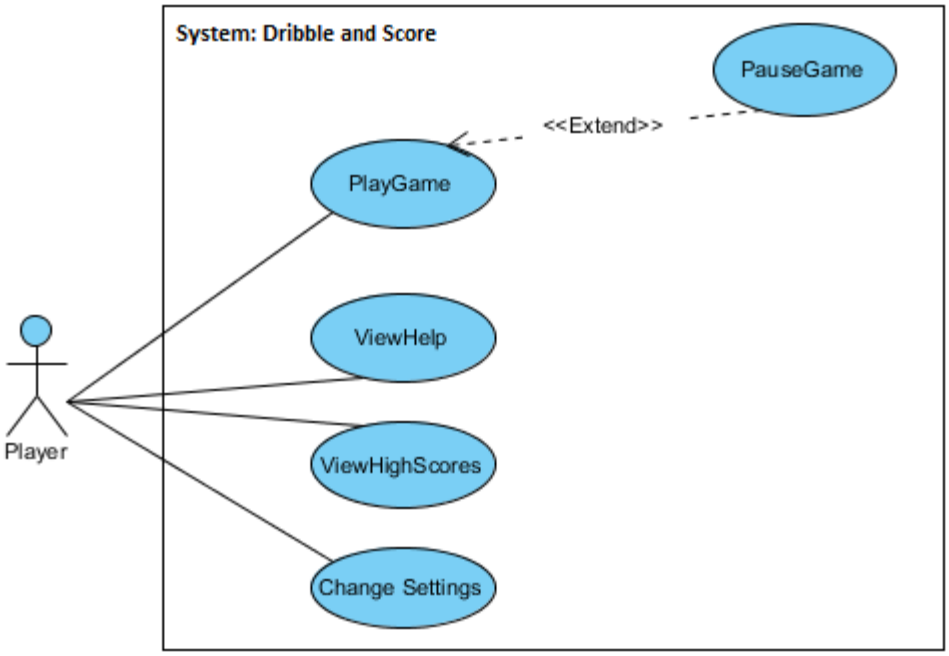


Figure 1- Use Case Diagram

4.1.1. Use Case Descriptions

Use case #1

Use case name: PlayGame

Participating actors: Player

Entry Condition: Player selects level and press “Start game” button from main menu

Exit condition:

- Player decided to choose exit game via pausing, and selecting “quit option.” from menu. OR
- Player is hit by a defender. OR
- Player overcomes the first part and finished his shootout.

Main Flow of Events:

1. Player starts the game
2. Player selects any of the levels which are unlocked.
3. Player plays the level until beats it.
4. System displays player’s score to the screen.
5. Next level unlocked.

   Player repeats step 3 and 5 until all levels are finished.
6. System returns initial page of the game.

Alternative Flow of Event

1. Player starts the game.
2. Player is getting hit by a defender. (Display game-over message, go step 6.)
3. Player presses exit game button from menu. (Go step 6)

Use case #2

Use case name: ViewInfo

Participating actors: Player

Entry Condition: Player opens the game and pauses game.

Exit Condition: Player returns to main menu

Main Flow of Events

1. Player choses to view help menu on main screen.
2. The game controllers and high score is displayed on menu. .
3. Player choses to return the menu, and then play may resume or quit.

Use case #3

Use case name: ViewHighScore

Participating actors: Player

Entry condition: Player opens the game and pauses game.

Exit condition: Player returns to main menu.

Main Flow of Events:

1. Player presses “Show High Scores” button.
2. The system displays high score of player for each levels.
3. Player returns main screen menu.

Alternative Flow of Event:

1. Player is getting hit by defender and loses game.
2. System displays high score of this level.
3. Player chooses either retry or go back to the main menu.

Use case #4

Use case name: ChangeSettings

Participating actors: Player

Entry condition: Player opens the game and is on main menu

Exit condition: Player returns to main menu.

Main Flow of Events:

1. On main screen menu, player presses settings menu
2. System displays different adjustments, change character, control, reset high score.
3. Player chooses the settings which are better for him/her.

Use case #5

Use case name: PauseGame

Participating actors: Player

Entry condition: Player is playing the game.

Exit condition:
-Player returns to main menu.
-Player chooses to exit the whole game

Main Flow of Events:

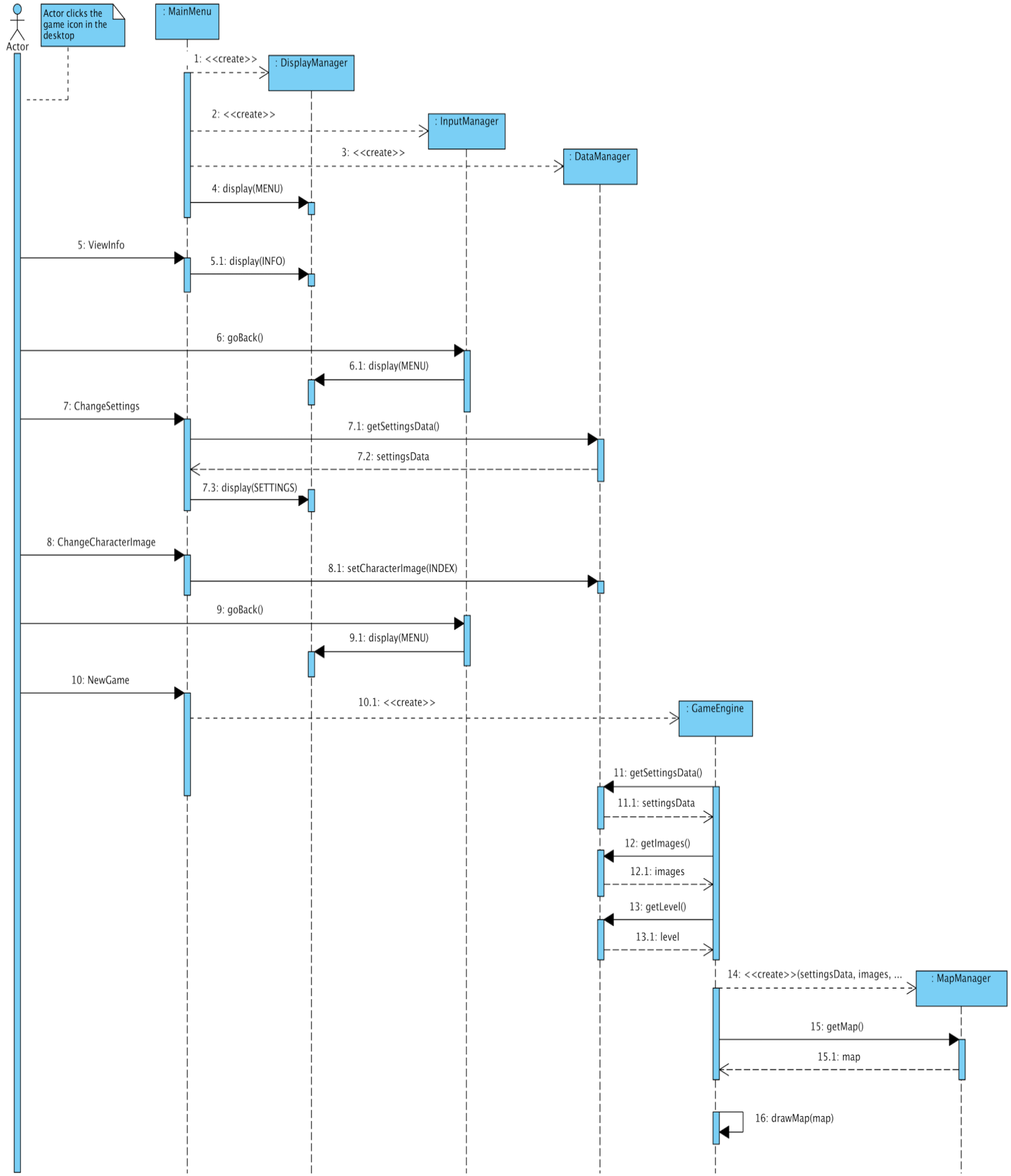
1. During the game, player presses the pause button.
2. While player is not pressed continue, game freezes.
3. Player presses continue.
4. Player continues the game

Alternative Flow of Event:
1. Player pauses the game.
2. System displays continue and go back buttons.
3. Player chooses the go back.
4. In main menu, Player selects another level to play.

4.2 Dynamic Models

Scenario Name: Execute Game

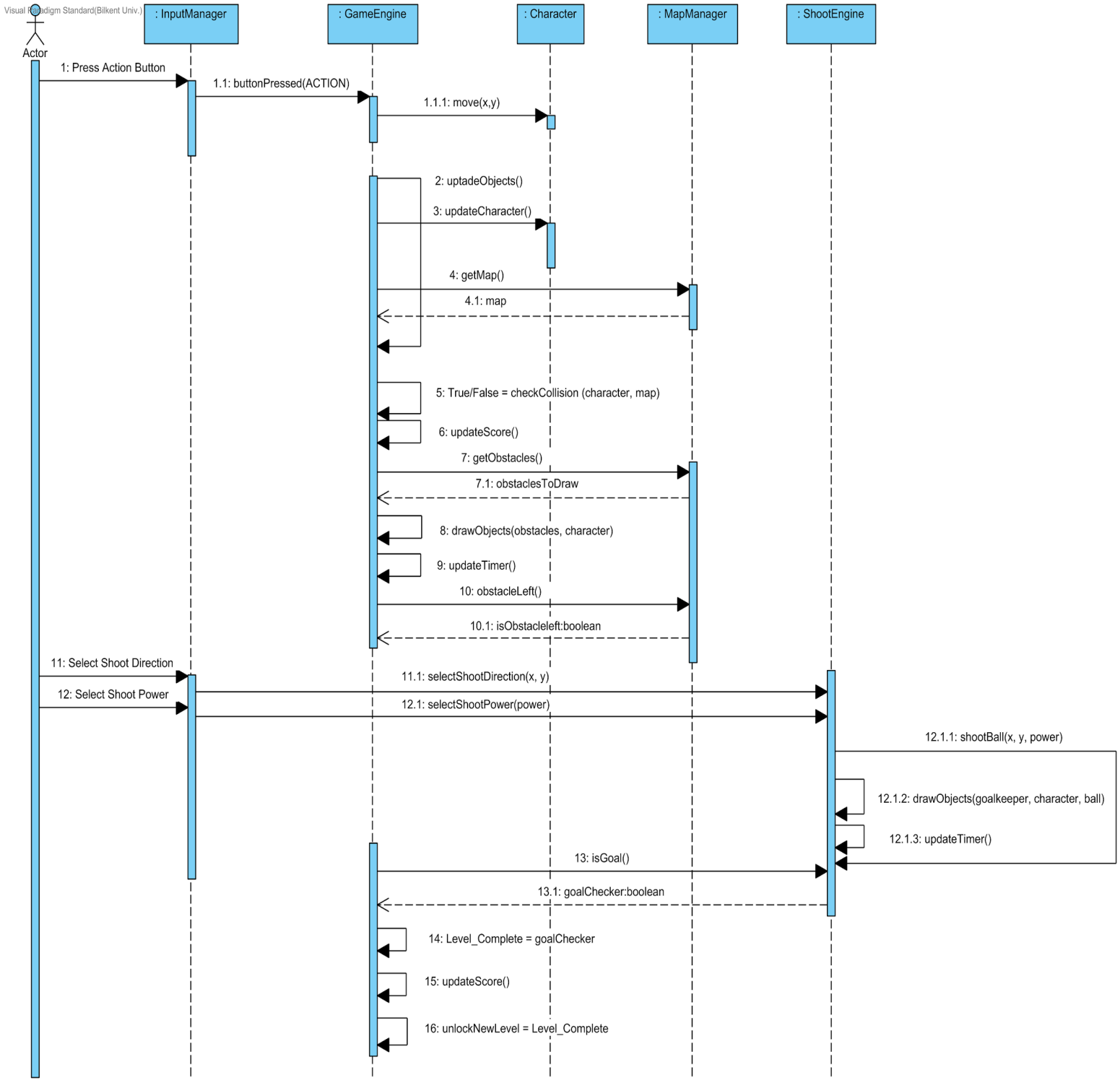
Scenario: When the game is opened first time, player sees the Main Menu and does not know how to play the game. Thus, firstly he looks to Info. After player learns how to play, he returns to Main Menu. Player wants to look which Settings options are available and then he change the character according to his supporting team. Afterwards, player return back to Main Menu and clicked to Start Game.



Description
After the game is executed “Display Manager”, “Input Manager” and “Data Manager” are created. “Display Manager” is responsible for loading the user interface views. “Input Manager” is responsible for reading the user inputs. “Data Manager” is responsible for saving the datas of the game. In this scenario, player first choose the Info option, which is loaded by “Display Manager”, for learning the game. Then, user presses the back button for returning back view, “Input Manager” gets the input about going back and orients the “Display Manager” according to input. Settings view is opened by “Display Manager” and fetches the datas from “Data Manager” and display the settings datas to user. After that, user change the setting about character and set character data in “Data Manager”. After “GameEngine” is created, it loads the datas from “DataManager” and pass these values to “MapManager” for creation of the map. Afterwards, “GameEngine” draws the map before the gameplay starts.

Scenario Name: Play Game

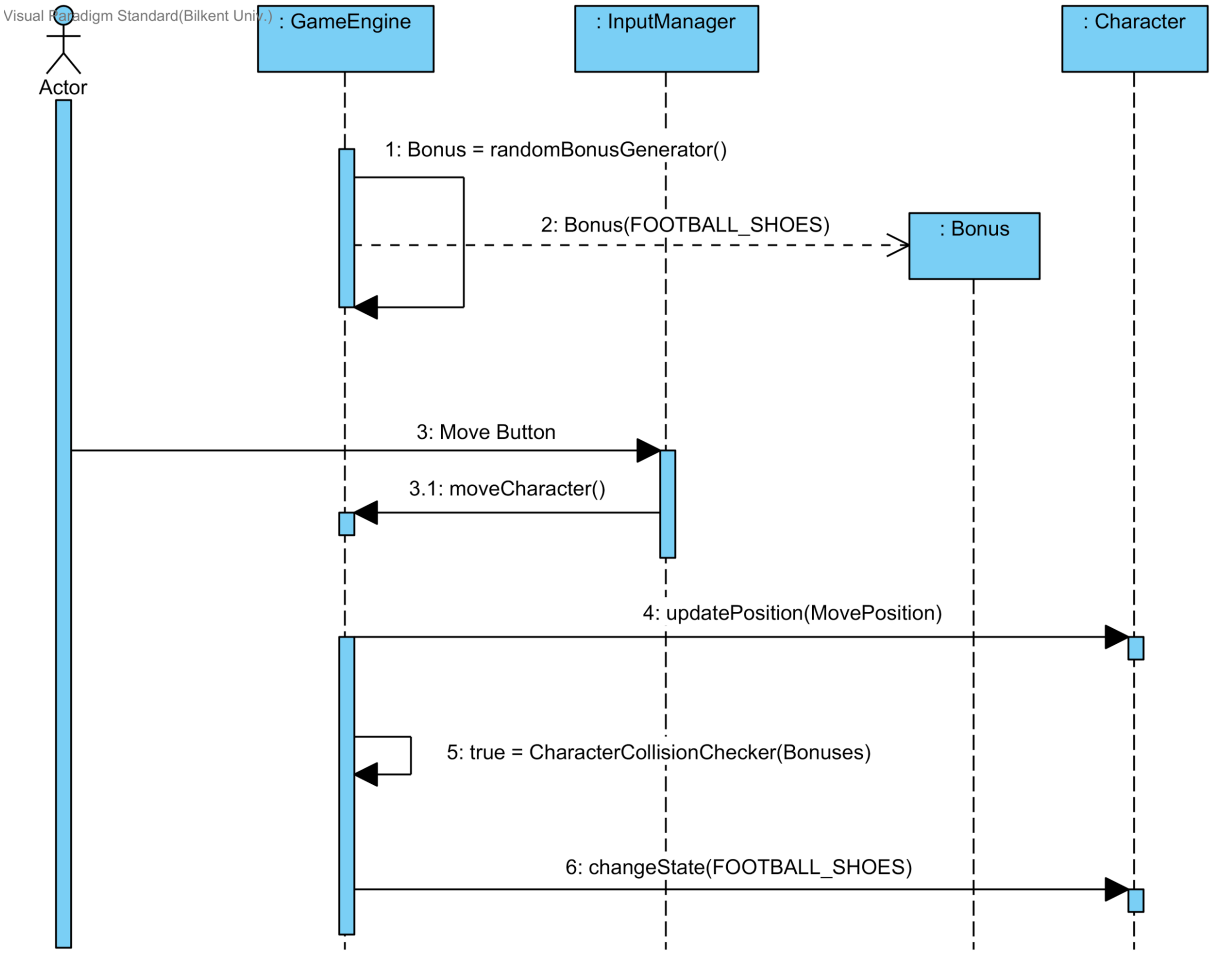
Scenario: Player starts the game. He starts the game which he selected. He successfully dodge the obstacles. After no obstacle left, score parts take in place. Player chooses direction of the ball and power of the shoot. He successfully score and unlocked next level.


Visual Paradigm Standard(Bilkent Univ.)
Actor
: InputManager
: GameEngine
: Character
: MapManager
: ShootEngine
1: Press Action Button
1.1: buttonPressed(ACTION)
1.1.1: move(x,y)
2: uptadeObjects()
3: updateCharacter()
4: getMap()
4.1: map
5: True/False = checkCollision (character, map)
6: updateScore()
7: getObstacles()
7.1: obstaclesToDraw
8: drawObjects(obstacles, character)
9: updateTimer()
10: obstacleLeft()
10.1: isObstacleleft:boolean
11: Select Shoot Direction
12: Select Shoot Power
11.1: selectShootDirection(x, y)
12.1: selectShootPower(power)
12.1.1: shootBall(x, y, power)
12.1.2: drawObjects(goalkeeper, character, ball)
12.1.3: updateTimer()
13: isGoal()
13.1: goalChecker:boolean
14: Level_Complete = goalChecker
15: updateScore()
16: unlockNewLevel = Level_Complete

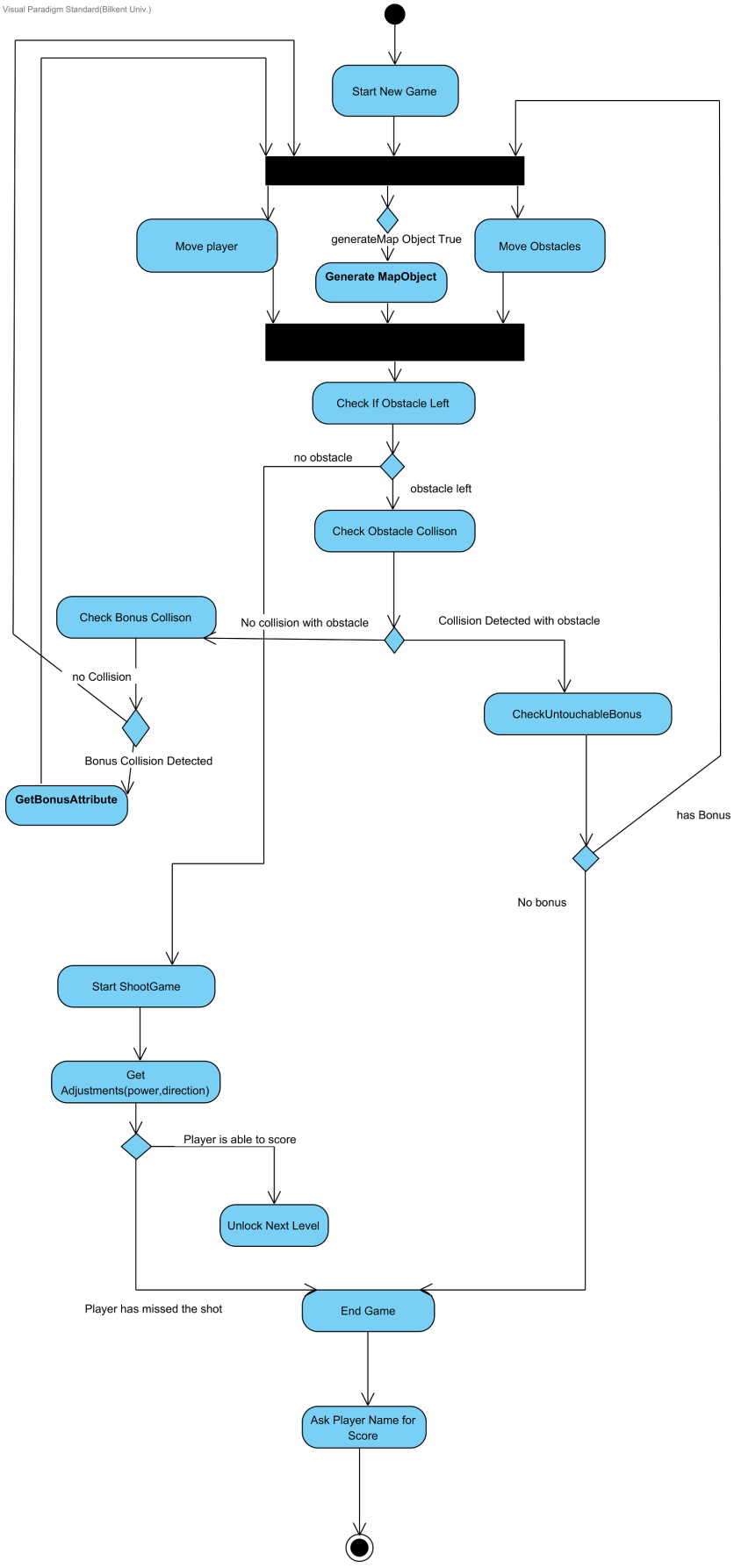
Description
When player pressed any control keys, character starts to move. “GameEngine” responsibles for the updating the position of the “Character” when players moves in other direction. “GameEngine” takes map from “MapManager” to creates a map and also check whether character hits the obstacles in the map or not. If not “GameEngine” continues to draw obstacles and character. When there is no obstacle left, shoot engine starts. Before the “ShootEngine” part, all of events are in the loop. Player chooses direction of the ball and those informations sends to “InputManager”. “InputManager” sends those datas to “ShootEngine”. “ShootEngine” draws goalkeeper, “Character” and ball. By the help of given shoot direction and the power, “ShootEngine” calculates whether player able to goal or not. It sends this data to game engine. If player goals, score will be updated and next level will unlocked.

Scenario Name: Acquire Bonuses

Scenario: Player has already started playing. At some point of the map, player saw a “football shoes” bonus. Player moves character into the bonus and got the bonus. This bonus enables character to run fast.

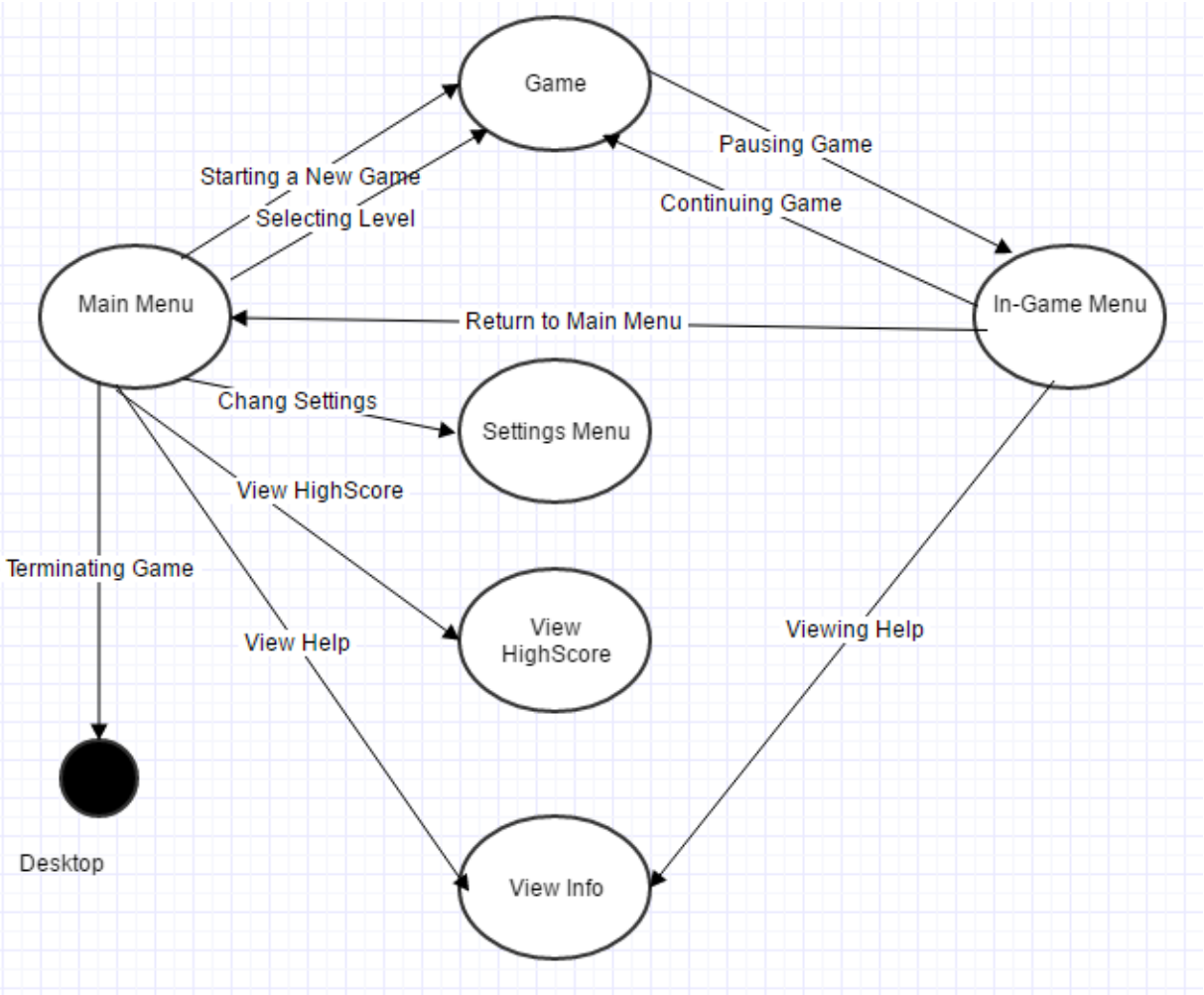


Description
“Game Engine” generates random bonus and random position in the map. When character gets bonus, corresponding changes applies to the character.

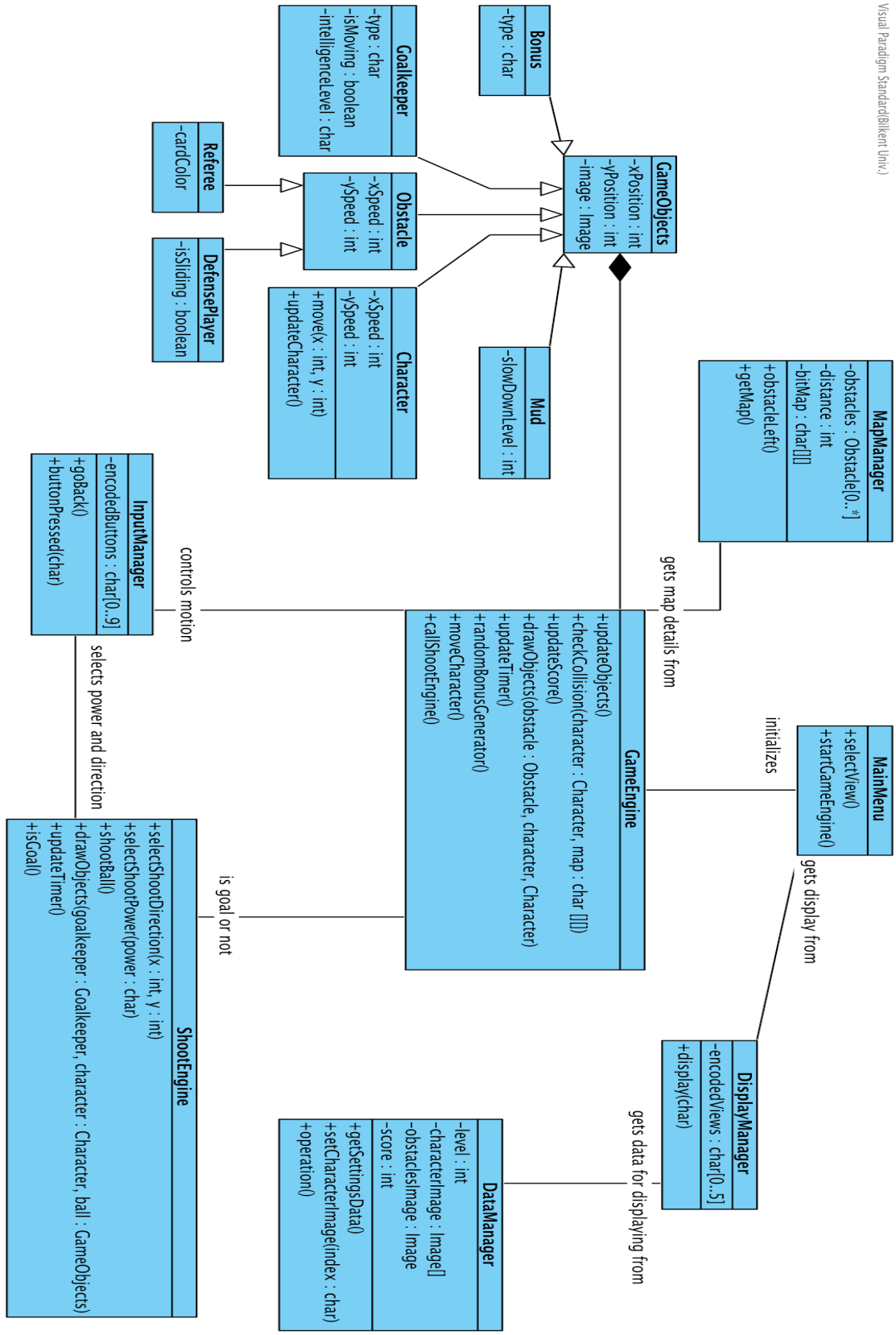


Activity Diagram of the Game

Navigational Paths



4.3 Object and Class Model



In the design of the class diagram, “MainMenu” class can be considered as the main activity class, it is the first view which user interacts. “GameEngine” is responsible for managing all game related issues, especially the first part of the game which is running part. The second part of the game is controlled by “ShootEngine” which is responsible for managing the shootout and checks the shoot is scored or not and pass it to “GameEngine”. During the program's execution, “Input Manager” is responsible for reading the inputs of the user and assigns these inputs to related classes and functions. “DataManager” is responsible for saving and fetching the datas of the game. “DisplayManager” is a simple class which is just responsible for transition of the views.

4.4 User Interface

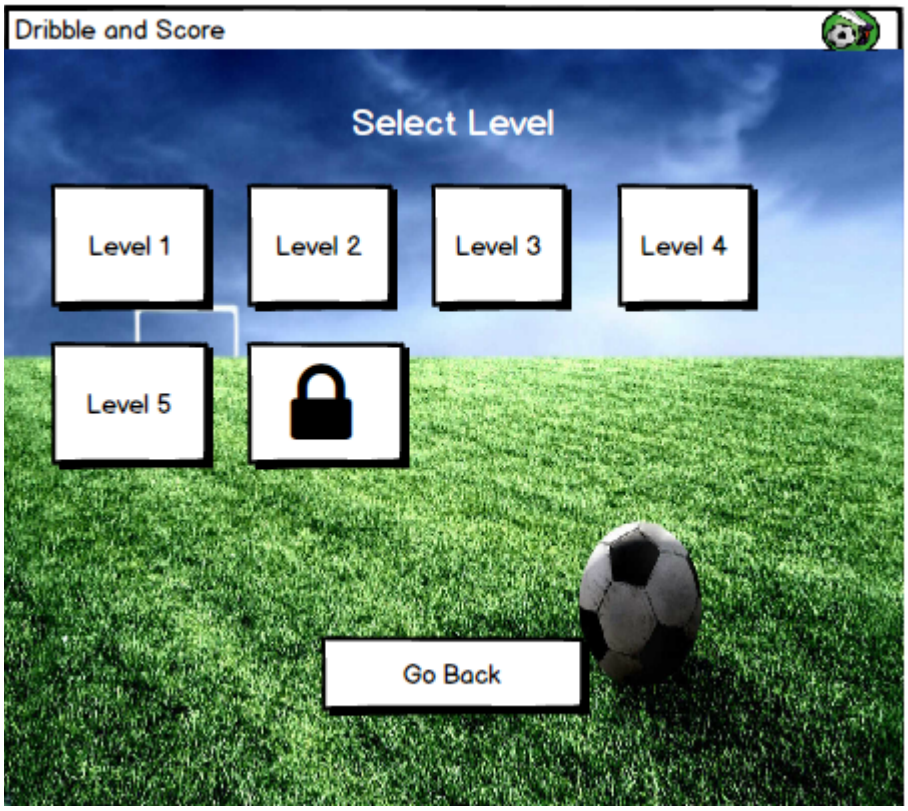
When the user launches the game, main menu is being displayed. The options he may select are Play Game, High Scores, Settings and Credits.

Play Game: User is being forwarded to Select Level Menu, to select an available menu and start playing.

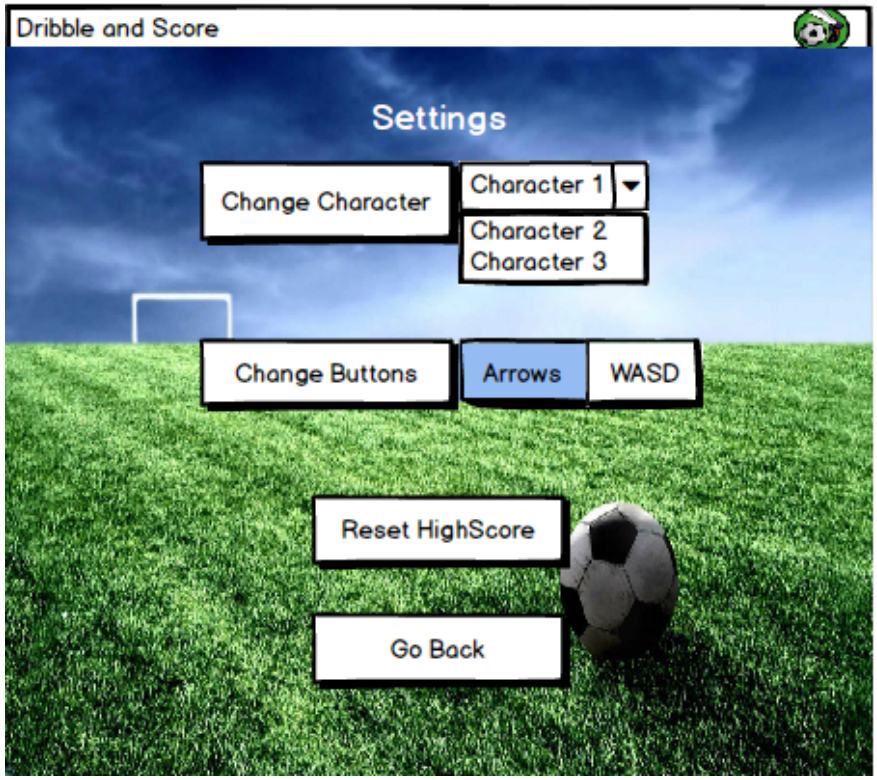
High Score: User can see the high scores which calculated by adding most points a player get in each level he played.

Settings: Player may change main character, controllers or may reset the high score table.

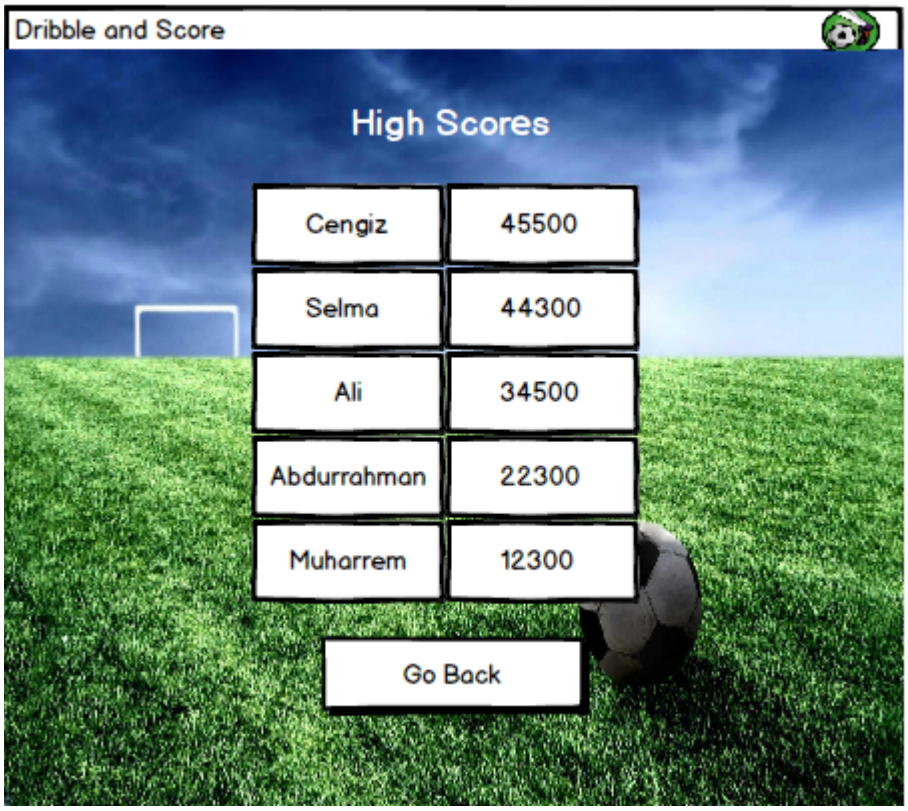
Credits: Player may see the contributors of the game.



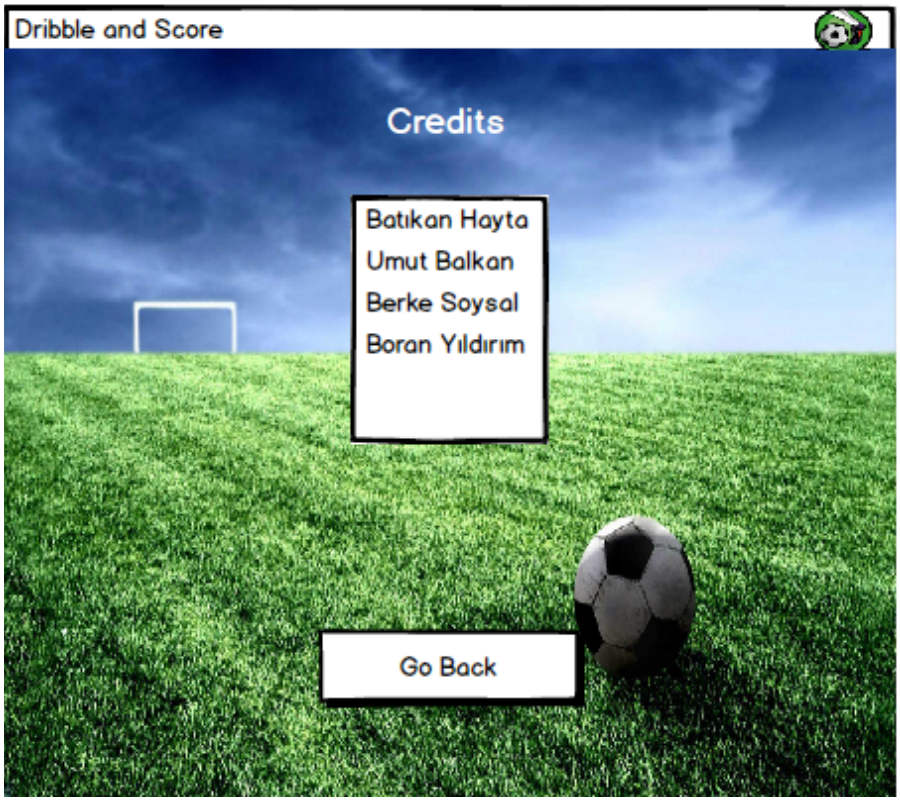
Player may play the previous levels he played to get a higher score or may choose to play the last level to unlock the next level.



Player may change the character appearance, controllers or may reset the high score here.



The high scores are being displayed here.



The developers of the game is being displayed here.

5. References

1. http://www.stockunlimited.com/vector-illustration/capsule_1268415.html
2. http://sociable.co/wp-content/uploads/2013/01/Rugby_Guy.jpg
3. https://www.pixilart.net/art/create-your-own-8-bit-shoe-d0cd572d734ba21