CS319 – Object Oriented Software Engineering

Final Report-2

Run Dot Run!

Group 2-H

Beyza Tuğçe BİLGİÇ

Gökalp KÖKSAL

Gökçe ÖZKAN

Emine Ayşe SUNAR

Contents

1. Implementation Flow

We have mentioned that the game will be implemented with a 3-tier architecture in the design report, thus we implemented the code using this architecture. Since we have a 3-tier architecture, we have mainly 3 packages, which are Game Objects, Game Control and GUI. We have implemented the GUI package and the Game Control and Game Objects packages separately, and then combined them both. Since the Game Control and the Game Objects packages were tightly communicating, both of them are implemented together. We have first started by creating the Game Object package. The super class GameObject was firstly created and the subclasses were then implemented. After this, the Game Control package was created. The classes of the Game Object package, which are the main objects in the game, were used in the Game Control package. The Game Control mainly uses the Game Objects and controls them by updating and drawing them. After the main game play was implemented, the GUI was added on top of this; in other words the GUI was attached to the game to create a fully functional game.

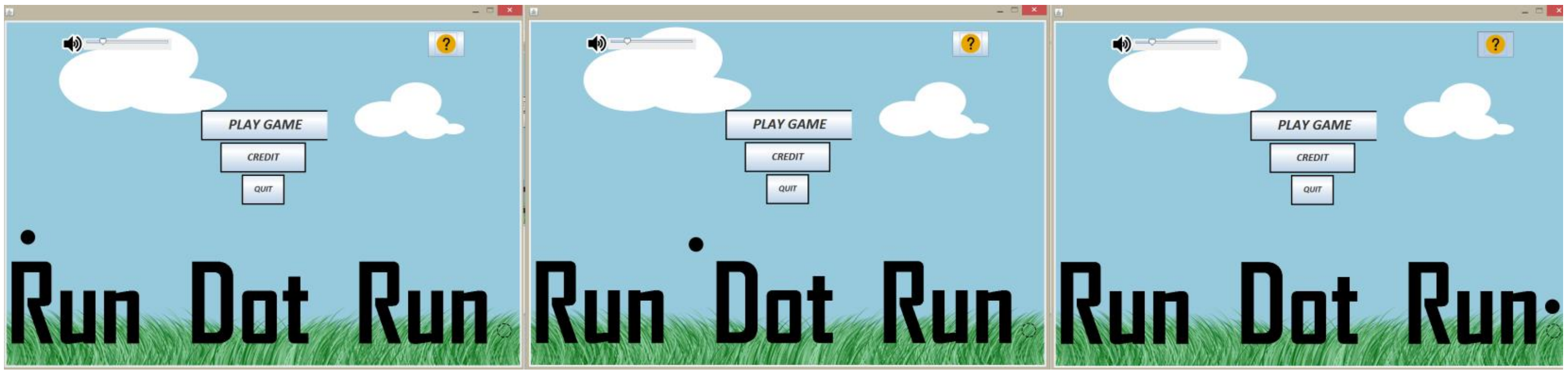
2. Changes in the Design

- In our design report, we have planned that the GameManager class, which is the façade class of the Game Logic Subsystem, would extend JPanel, however, after doing some research we have found out that Canvases work more efficiently in 2D games like ours. So, eventually we decided that our GameManager class would extend Canvas. After extending Canvas, a buffer strategy was added to the class to increase efficiency. The buffer strategy is a class which creates the mechanism to organize complex memory. Since our game has to handle many objects at the same time, we decided to add a buffer strategy.
- We have mentioned that that the collision detection would be done in the GameManager class, however to create a more organized code, we decided to add a new class named CollisionDetection which has a method named detectCollision. This method takes the linked list of objects and first specifies the Dot object, which is the main player. Later on it checks whether the other objects in the linked list collide with the Dot. An instance of

this class is created in the Dot class and the detectCollision method is being called in the update method of the Dot class.

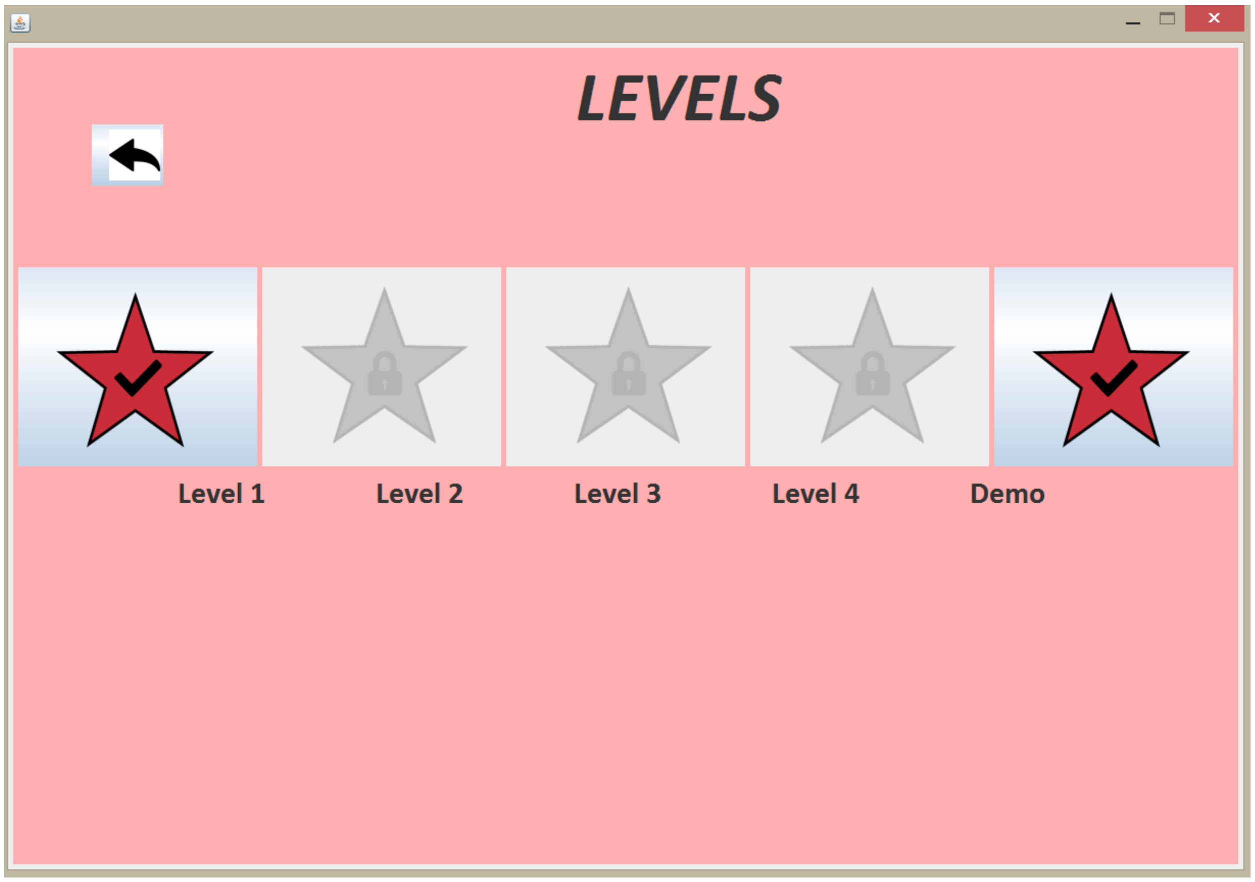
- For our FadingLetterBox class, we mentioned in the analysis report that once the dot collides with these objects, the letter will slowly disappear. Instead of making the letter boxes disappear, we have decided to give them a velocity, so once the dot touches these objects they will start falling down. This change was made because it creates a better effect in the game and thus makes it more enjoyable to play.
- In the design report we haven't mentioned adding ActionListener's to the GameObjects, however in order to present a game with better visual effects, some animations were added to some of the GameObjects. In order to add animations ActionListeners were added to some of the GameObjects. For example, the CheckPoint GameObject has an animation which plays an animation of a turning picture of "C", which represents the checkpoint.
- An animation of a simple game play was added to the main panel of our game. This wasn't mentioned in the design report, however in order to add animations to the main panel, an ActionListener was added to the MainMenuPanel class.

3. User Guide



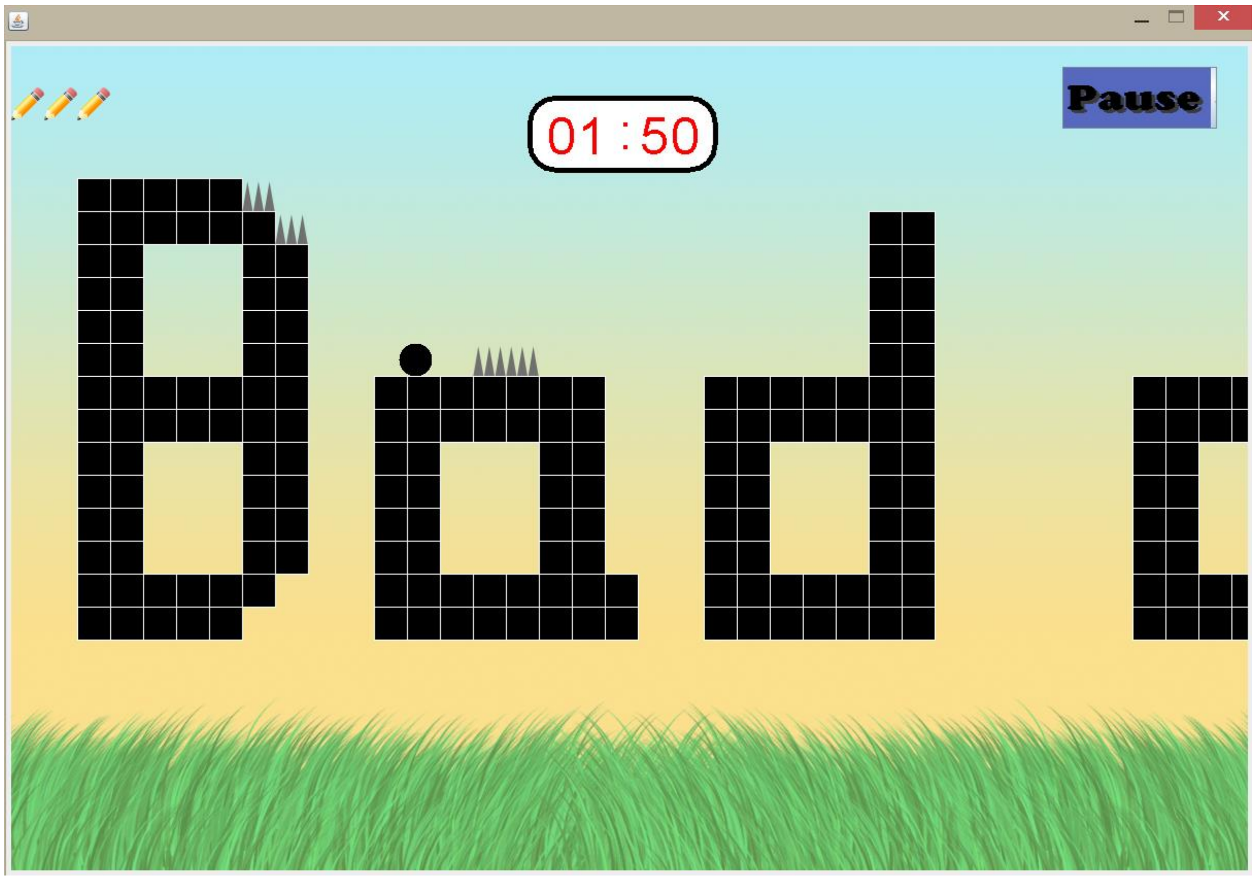
Main Menu Screen (final version)

The game starts with a menu screen offering a variety of options. There is an animation of a simple version of the game play, as it could be seen in the figure (These three pictures are the screen shots of the main menu. Three screen shots were added to the user guide to emphasize that there is an animetion at the background). By clicking the question mark button at the upper right corner of the menu, user can find all necessary information about the rules of the game. Also, credit option provides the user with names of the developers of the game. If the user can exit the game with “Quit” button or display the levels screen with “Play Game” button.



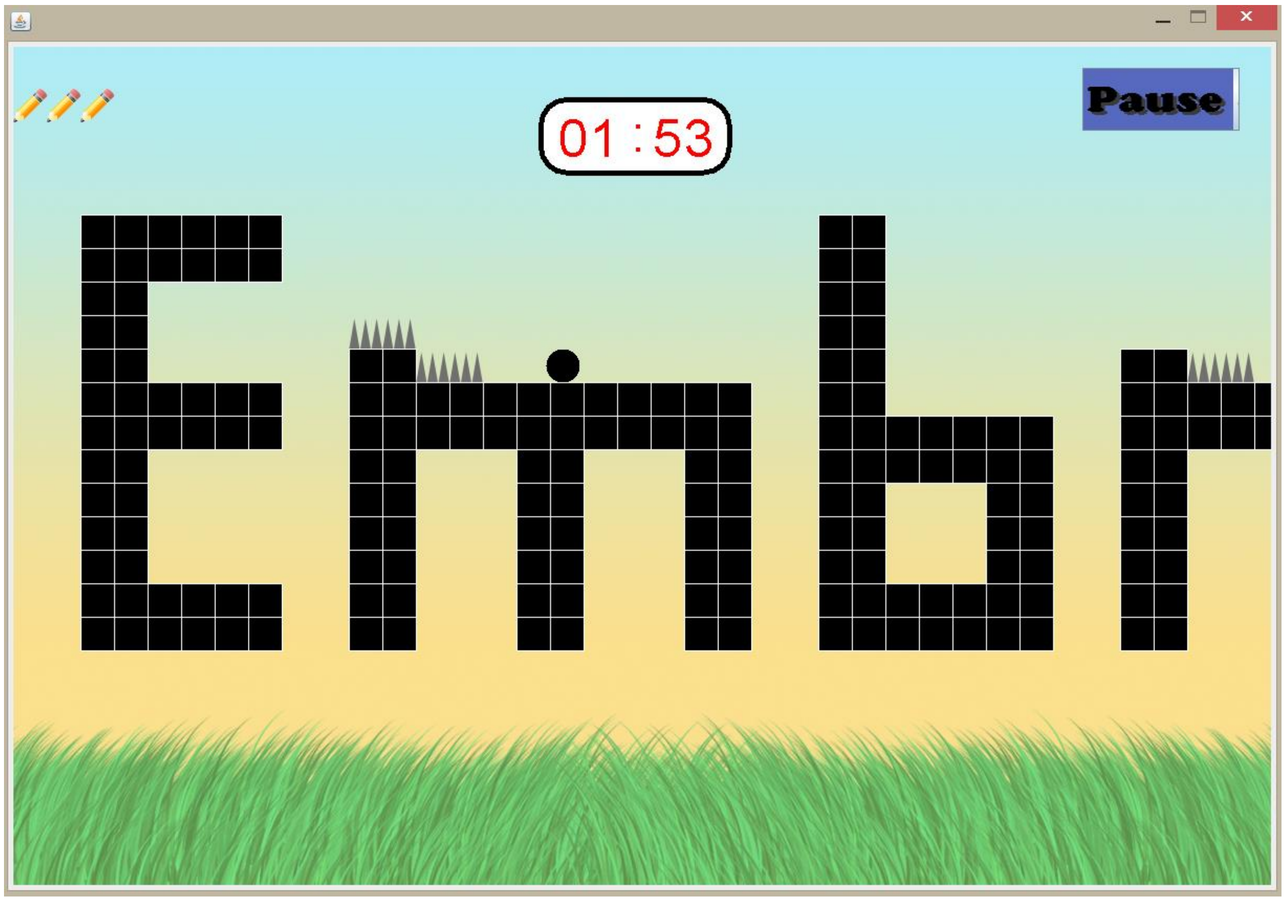
Levels Screen

After selecting the play game button, the levels including different types of difficulties such as spikes or erasers are displayed and the user has to select an unlocked level. As it could be seen in the picture, level 1 is unlocked and the other levels until level 4 are locked. As the user wins the games, the levels will become unlocked. There is a special level called demo. This level is created special for our demo for this project (This level is created to represent all functionalities of the game, so it consists of every GameObject, thus every obstacle the user may come across). After the game starts, there is a counter on the screen then, the user can check the time remaining for finishing the level.



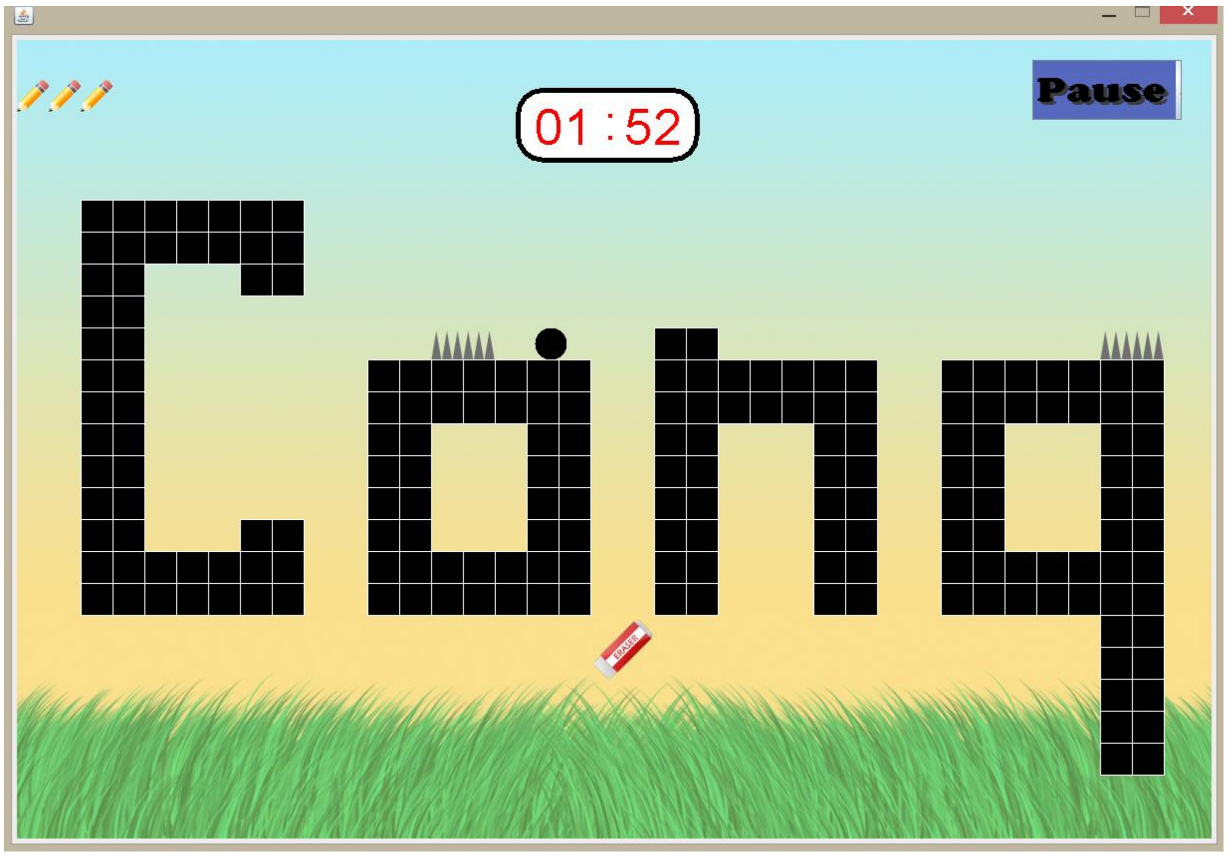
Main Game Screen with Pause Button

By using keyboard, the user controls his/her movements during the game. The purpose of user is to reach the end of the sentence and to finish the game without falling down. At the same time, the user has to track how much time is left through checking the counter. While playing the game, the user may pause the level and return to menu by selecting exit button or continue with playing this level by selecting resume button.



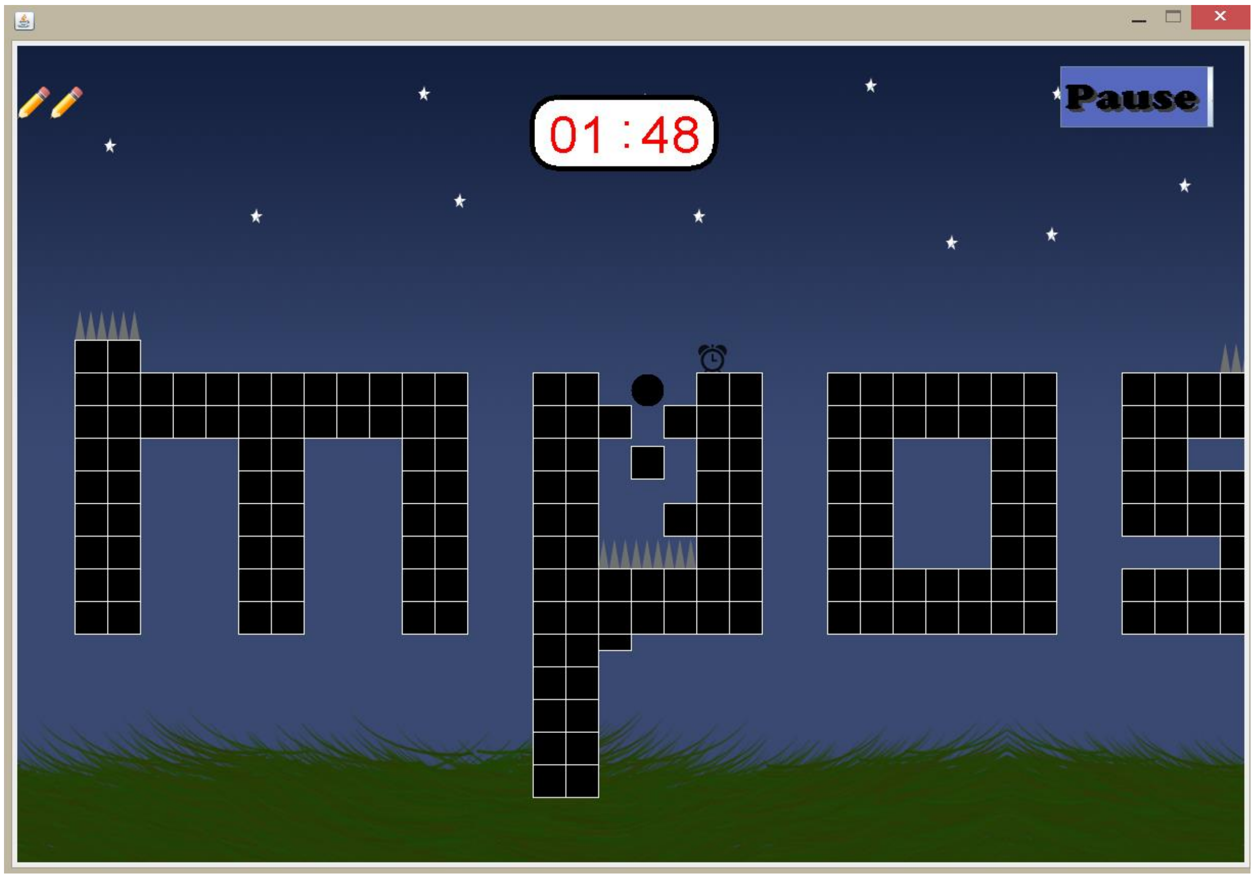
Game Screen with Spike

The user encounter diverse obstacles during the game. For instance, the user should not hit the spikes because there will be a decrease in the number of “lives” and the user will return to the beginning of the level in case of hitting spikes.



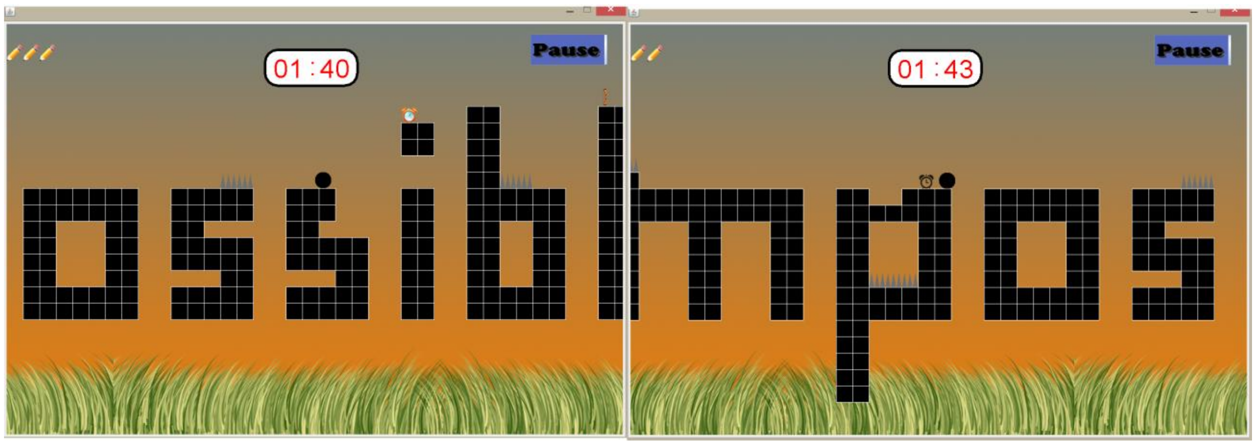
Game Screen with Eraser

Additionally, the user has to face with “erasers” which is another obstacle of the game after he/she manages to reach level 3. These eraser objects comes from the sky or the ground.



Game Screen with Fading Letters

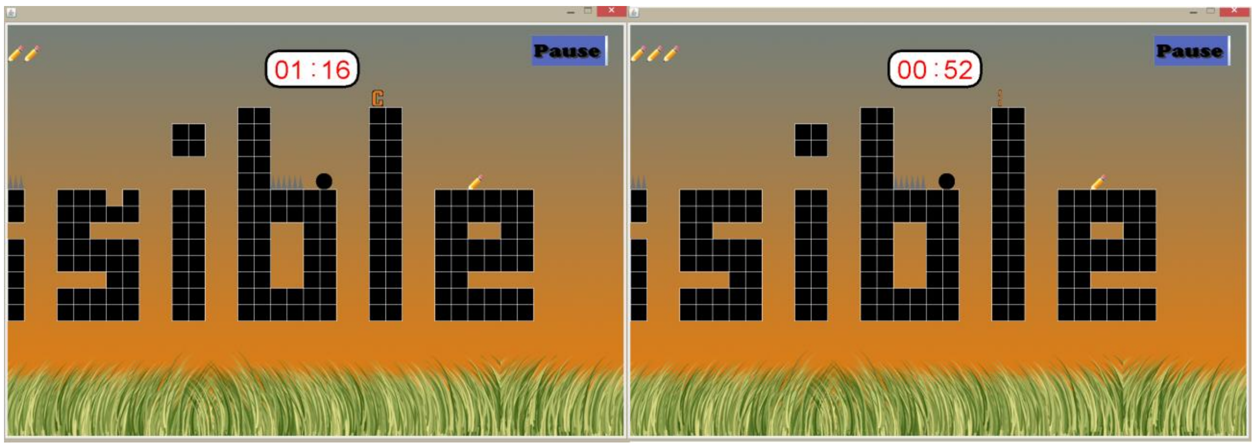
If the user can unlock the level 2, the game starts with "fading letters" which are able to disappear in 1-2 seconds when the dot hit these letters.



Game Screen with TiemBonus(on the left) and

time punichemtn (on the right)

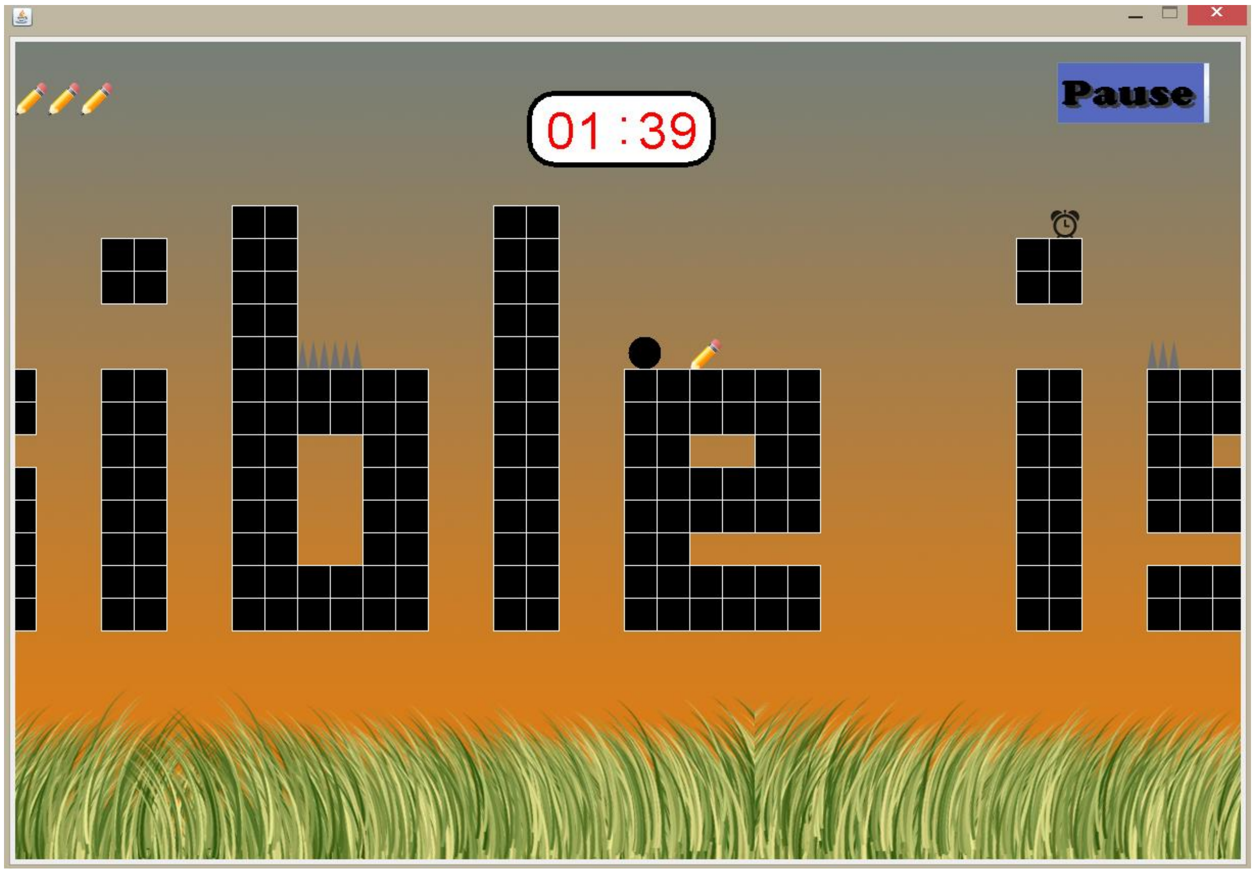
There will be a time bonus object which will increase the time by 10 seconds if the dot gets it. Also there will be another clock, in black and white, which will decrease the time by 10 seconds if the user gets it.



Game Screen with CheckPoint animation(There are

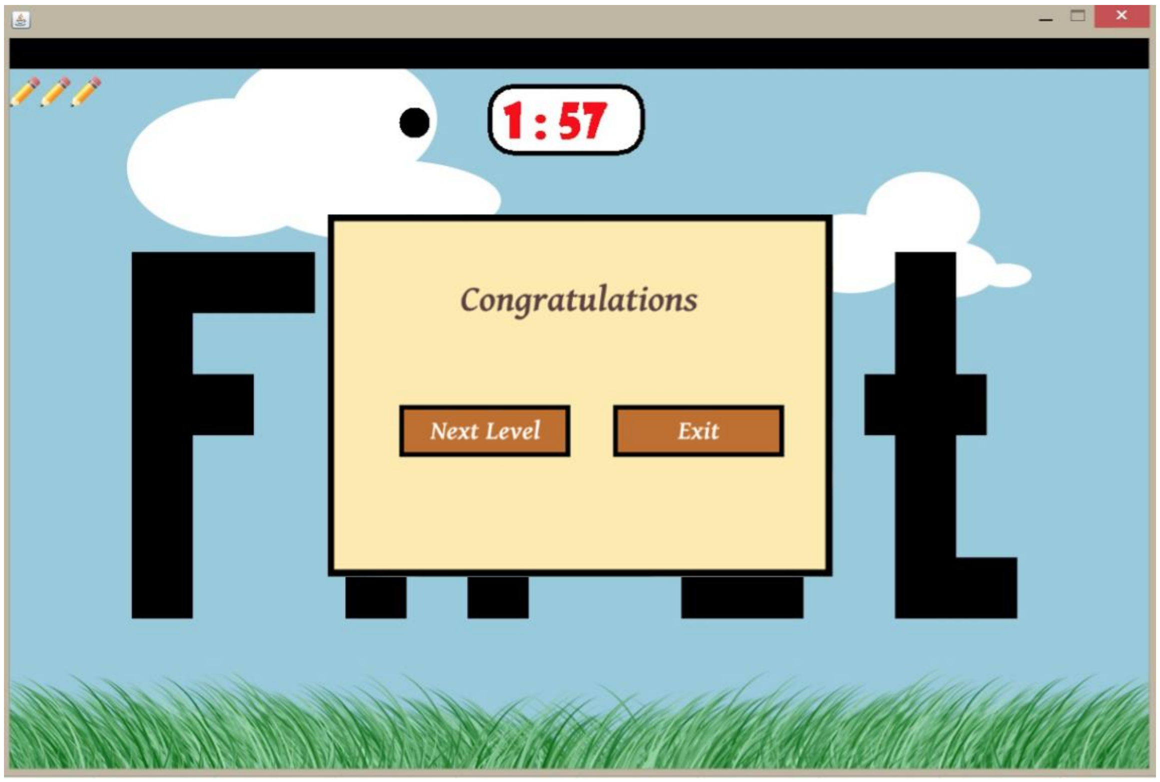
2 pictures to Show the animation)

There will be checkpoints during the game. If the Dot reaches a checkpoint, the next time it dies, it will start from the check point location, instead of the start point of the game.



Game Screen with extra life

There will be extra life's in the shape of a pencil throughout the game. When the user gets one of these extra lif's, it's life will incraese by one. If the Dot already has 3 lives, which is the maximum number of lives, the extra life will not be taken.



Next Lexel Pop-Up

If the user manages to finish any level successfully then, there will be a shown pop-up asking the user for deciding whether he/she wants to play next level or return to main menu. In case of falling down or hitting an obstacle, the user returns the beginning of the level without resetting the counter.

4. What has been left in the implementation?

- Once the main menu is opened, there will be some music and different sound effects as the ball jumps, collides etc. playing in the background. However, the feature of volume control is not finished yet. Therefore, the music plays at the same volume level and cannot be controlled by the player.
- Curvy letters feature could not be implemented yet. The letters the dot has to jump over are letters which have sharp edges, thus the dot does not slide from the edges.

- More animation and music are to be added.

In the next parts, after the necessities which are left are completed and tested, user feedback will be taken into consideration. According to the evaluation of the new opinions and feedbacks, the current features of the game can be developed and new features can be added.

5. Conclusion

In this project there are many things we have learnt to get use to the real life projects since the main aim of this course is to enable us to get some experience on working in a group of people that you do not know in real jobs.

<u>Experiences:</u>

- **Planning, Designing the Project**

We saw that planning and designing the project before getting into coding has huge effects on the quality of the work. It enabled us to see some of the problems that might occur and to correct them beforehand. Also, by looking at the designs it is possible to see everything that are needed in a chart instead of searching in a pile of code.

- **Project Management**

We learned how to use Github to build our project with other teammates. With Github, we can share our work with our team easily. It gives us a place to load our work and save them in instead of having several copies of work done.

- **Time Management**

Thanks to Bilkent's tough conditions of working, we had to learn to deal with the deadlines to experience time management.

- **Team Work**

This project made us to see how the work is being done in real job’s projects since we were assigned randomly in our projects. We had to learn how to work with other people that we do not know to proceed on the project.