



Bilkent University

Department of Computer Engineering

CS319-OBJECT ORIENTED SOFTWARE ENGINEERING

Q-Bitz: Q-Bitz

Iteration 1 - Final Report

Sait Aktürk, Zafer Tan Çankırı, Berkin İnan, Halil Şahiner, Abdullah Talayhan

Supervisor: Eray Tüzün

1. Introduction	3
2. Design Changes	3
3. Lessons Learnt	3
4. User's Guide	3
4.1 System Requirements	3
4.2 How to Use	3

1. Introduction

We have started the development process after the design report. Since the game has both singleplayer and multiplayer modes, we wanted to work on both of the modes at the same time. For multiplayer mode, we have implemented the basic functionalities using a local server and database. These functionalities include registration to the system using email verification and logging in to an account. A game with multiplayer players have not been implemented yet. For single player mode, we have implemented the game itself with the most fundamental mode which is race mode.

The 3D mechanics are implemented using JavaFX[1] without using third party 3D engines. Rest of the user interface components are also implemented using JavaFX and JavaFXML.

2. Design Changes

Although there are not much changes in the classes, there are significant changes in the game mechanics like the following:

- We changed the cube control mechanics from keyboard to mouse because in a game where time is a constraint, using keyboards for rotating the cubes was not efficient in terms of time.
- Instead of moving the cube from focused area to the board, we put a preview of the selected cube face on the board before placing it.
- The user is able to rotate the placed cube face on the board without the need of removing it, this functionality was necessary because there is no need to select a face again only for the sake of changing the rotation of it.

We had several changes for Client Server mechanics as well:

- There was a missing connection between the ServerSocket and the main server class, this connection was necessary because only the main server class can access the database using the database connector.
- We added a SceneController to the Client, the reason of this change is that, for each message that comes from the server, we need to notify different scenes in the client. This class controls the information for deciding which scene to send the message from the server. Since this is the only additional class, we haven't included the updated class diagram.

3. Lessons Learnt

3.1 Technical

- Since the 3D functionalities of JavaFX is fairly limited, the actual game mechanics took more time than it was expected. JavaFX does not include Quaternion rotations in 3D objects, it uses Euler angles resulting in a famous computational problem called Gimball Lock[2]. We have learned that it would be better to examine the functionalities of the system that we are using. We could have started the implementation of game mechanics earlier if we knew the capabilities of JavaFX better.
- We learned about splitting Client threads from javafx threads in order to resolve the error related to illegal threads.

3.2 Teamwork

- We learned the importance of using github, it enhanced the workflow of the project monumentally. Everyone in the team were able to use different development environments and still we were able to merge and pull the codebase efficiently.
- We learned the importance of splitting the work between team members, this was useful for implementing the project faster because different team members built different parts of the project. Still different parts were easy to integrate because of the project design phase.
- Other than splitting the work, team work was useful for solving difficult problems. For figuring out the rotational mechanics and pattern checking mechanics, multiple people worked on the problem and proposed different solutions.

4. User's Guide

4.1 System Requirements

- Java 8 in order to support JavaFX 3D properties.

4.2 How to Use

4.2.1 Main Menu

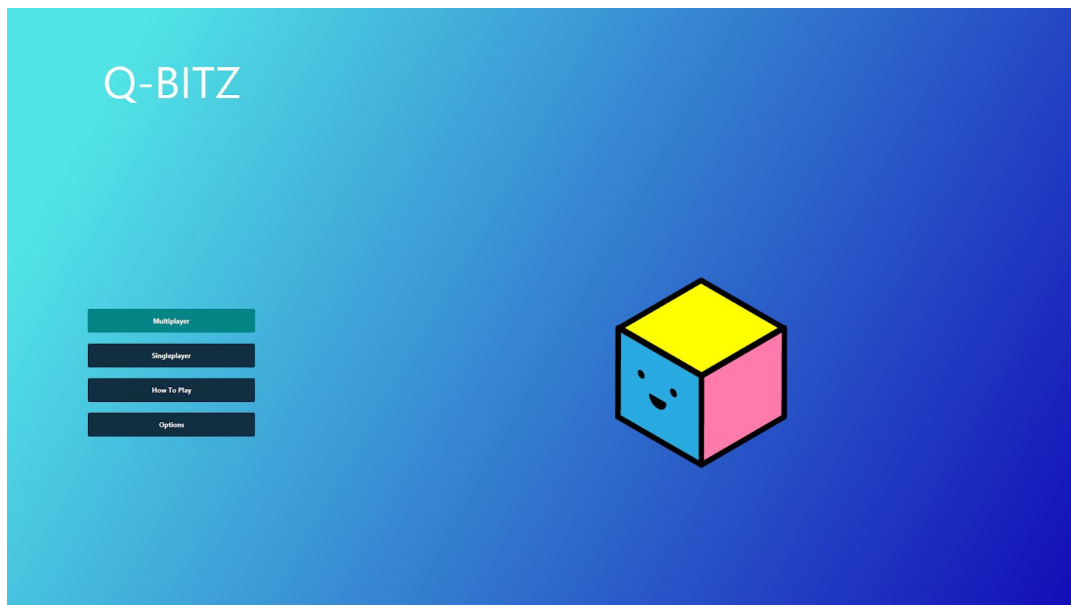


Figure 1. Main Menu

4.2.2 Login and Register for Multiplayer

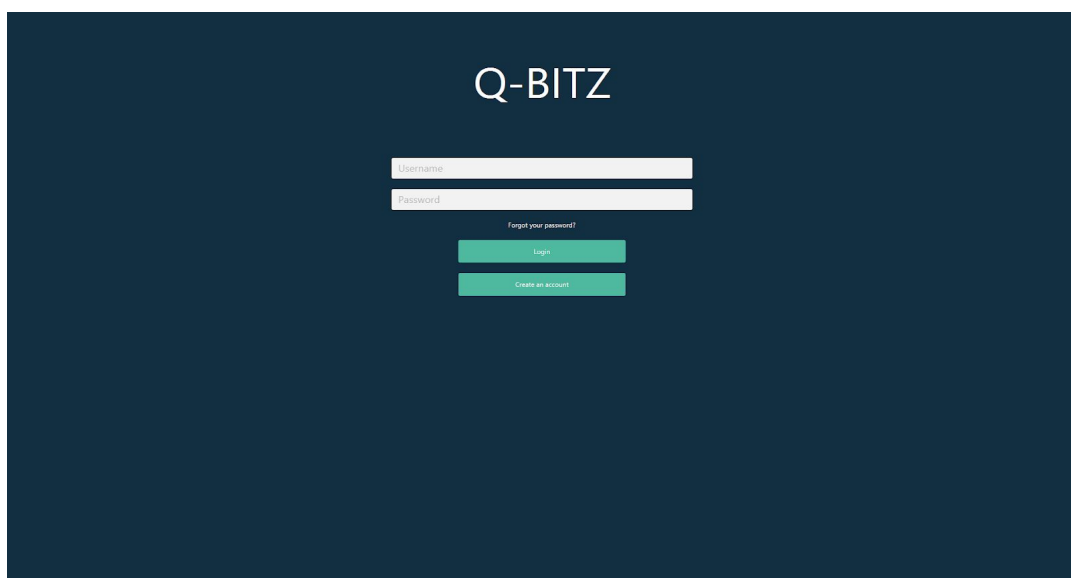
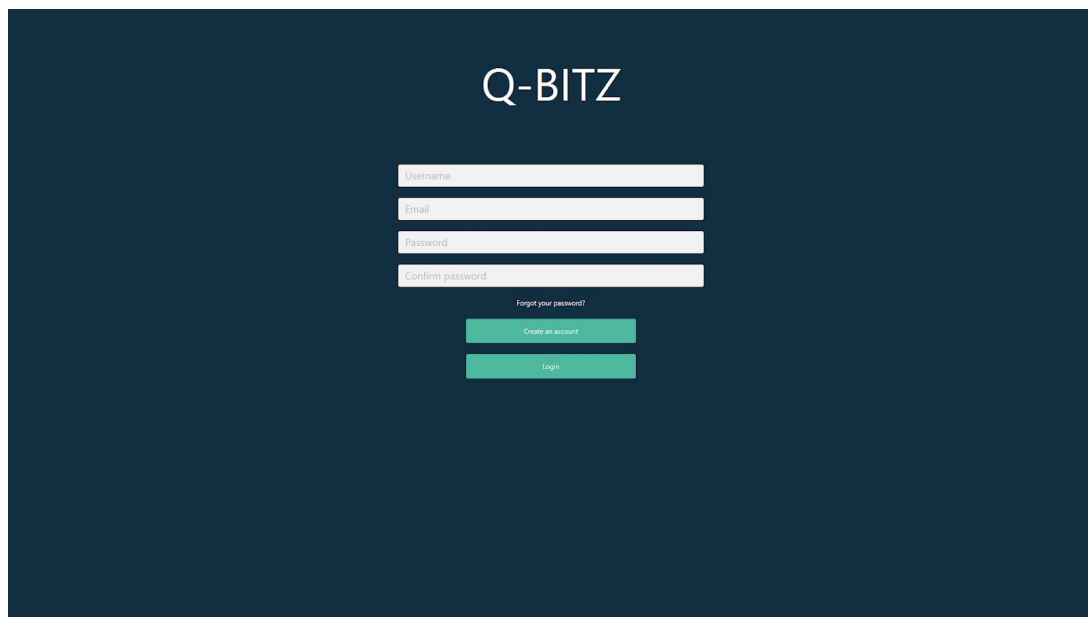


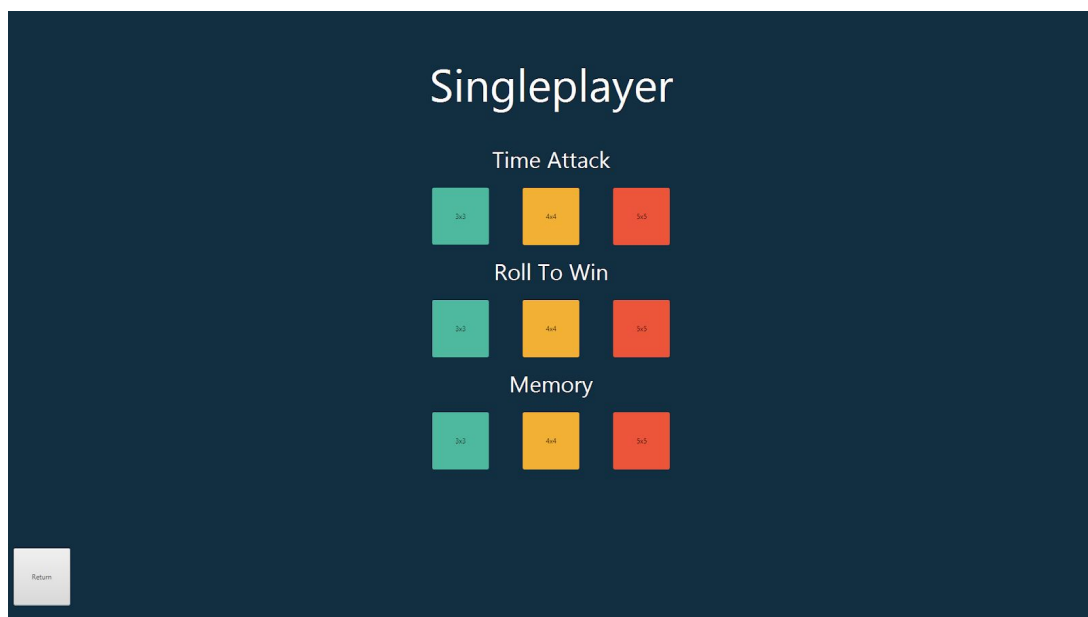
Figure 2: Login Menu



The registration menu for Q-BITZ features a dark blue background. At the top center, the text "Q-BITZ" is displayed in a large, white, sans-serif font. Below the title, there are four white input fields stacked vertically, labeled "Username", "Email", "Password", and "Confirm password". To the right of the "Password" field, there is a link that says "Forgot your password?". Below the input fields, there are two green buttons: "Create an account" and "Login".

Figure 3: Registration Menu

4.2.3 Level Selection Menu



The level selection menu for Singleplayer has a dark blue background. At the top center, the text "Singleplayer" is displayed in a large, white, sans-serif font. Below the title, there are three game modes listed: "Time Attack", "Roll To Win", and "Memory". Each mode has three colored buttons below it: a green button labeled "3x3", a yellow button labeled "4x4", and a red button labeled "5x5". In the bottom left corner, there is a small, light gray button labeled "Return".

Figure 4: Level Selection Menu

4.2.4 Game Screen

5. References

[1] <https://www.oracle.com/technetwork/java/javafx/overview/index.html>

[2] <http://www.okanagan.ieee.ca/wp-content/uploads/2016/02/GimbalLockPresentationJan2016.pdf>