Supreme Checkers v1

Context

Drexel University | **SE-181: Intro to Software Engineering**
Fall 2020

Project Members

Christopher Clifford	crc339@drexel.edu
Henry Dang	hd349@drexel.edu
Peter Mangelsdorf	pjm349@drexel.edu
Conner Pierce	ccp49@drexel.edu
Thomas Trimbur	tft27@drexel.edu

Faculty

Filippos I. Vokolos	fvokolos@drexel.edu

1. Contents

2. Introduction

2.1 Purpose of Document

This document will provide all of the requirements for the game Overlord-Supreme Checkers. It will serve as a reference for the developers and the users for the development of the final product.

2.2 Scope of Document

This document will contain enough information such that a developer would be able to easily translate the code requirements.

2.3 Overview of Document

This document will contain function and nonfunctional requirements for the game Overlord-Supreme Checkers, as well as use cases, diagrams, and UI mock-ups. The game will run on PC for both the Windows and Linux operating systems. Both versions will be developed simultaneously using the Unity game engine and will integrate with the PUN networking asset for multiplayer support. The requirements will be provided in the document and will also contain mock-ups for the graphical user interface.

3. Description

Overlord-Supreme checkers is a recreation of the classic board game checkers that is intended to be played with 2 players per session. When each player launches the game, they will start in the main menu. From the main menu, they can either open a lobby or join an existing one. Once in the lobby, the player will load into the game scene where there will be a checkers board and will be presented with turn order and other information through the ui.

Figure 1: Player's View of the main menu screen (mockup)

The game itself will be 2-D, and the camera will show a top-down view of the board. The board will consist of the standard 8x8 square grid with offset colored tiles. The theme will be futuristic with hologram looking assets (similar to Tron).

Figure 2: style mockup

The game will play the same as classic chess where each player will have the goal of eliminating the other's pieces in order to win.

3.1 Interface

The interface while in the menu will contain buttons for creating/joining lobbies and exiting the game.
The lobby interface will have a textbox for changing the Player's name, a button for creating a new lobby, and a list of currently open lobbies.

3.2 Functionality

The game will game the following functionality:

- Networked multiplayer with the ability to either host or join a game
- Ability to run the game and restart when over
- Interactable game synced across players

3.3 User Description

The necessary number of players for Overlord-Supreme Checkers is 2 players

3.4 Assumptions and Dependencies

3.4.1 Unity

Unity is a game development engine that allows users to create video games that can be exported across multiple platforms with relative ease. This project is extremely dependent on Unity functioning and working properly. If Unity ceases support or breaks then the team would be unable to continue. There are other engines that could be used to work on the project but would require a complete rework of almost every single asset in the game. The team is assuming that Unity is not planning on dropping support in the near or distant future.

3.4.2 Photon Networking (PUN)

Networking is required for this project and will be using the Photon Networking asset, specifically the PUN protocols. If this functionality breaks, the project would not be able to be finished in time for the end of class. There are other networking solutions but they would require a large amount of work the the Unity Prefabs and structure in order to get working. We assume that Photon is not planning on dropping suppport in the near or distant future.

3.4.3 Art Assets

The art assets for the game will be Two-Dimensional and will be either free use from the web or generated by the team. We do not anticipate any issue with developing the assets required for the game.

3.5 Requirements Apportioning

Priority Level	Description
1	**Priority 1** requirements are essential to the correct functionality of the system. These requirements must be extensively tested and verified to ensure the system fundamentally works.
2	**Priority 2** requirements are non-essential, but may be considered in the design. These requirements are not guaranteed to be in the final build of the system. The system should function as expected without fulfilling these requirements. If all requirements of priority 1 are fulfilled, the team will consider fulfilling requirements of priority 2.
3	**Priority 3** requirements are non-essential and will not be considered in the design. These requirements are not likely to be in the final build of the system. The system should function as expected without fulfilling these requirements. If all requirements of priority 1 and 2 are fulfilled, the team will consider fulfilling requirements of priority 3.

4. Functional Requirements

4.1 Definitions

Tile A spot on the board upon which a piece could be placed or moved.

Man A regular checkers piece.

King A piece that can move diagonally backward.

Player A user who has connected to an opponent in the game and is about to begin the game, in the process of playing the game, or has finished the game. A player either controls the light or dark pieces. This term will be used to refer to the current player (the player whose turn it is).

Opponent A user who has connected to a game against the player. The opponent is also a player.

Move An action either player can take. This action involves moving a piece from one tile to another at least once, and may involve capturing another tile.

Turn A time frame in which either the player or the opponent makes one or more moves.

Pile A player's collection of captured opponent pieces.

4.2 Matchmaking

R4.2 Player Identification

- **R4.2.1** Upon starting the game, the player shall be prompted to enter a username. **Priority 1**
- **R4.2.2** The player's username shall persist across game sessions. **Priority 3**

R4.2 Menu

- **R4.2.3** After entering a username, the player shall be directed to a menu. **Priority 1**
- **R4.2.4** On the menu, the player shall be able to create a lobby or join an existing lobby. **Priority 1**
- **R4.2.5** On the menu, the player shall be able to exit the game. **Priority 1**

R4.2 Lobbies

- **R4.2.6** A lobby shall be identified by the player who started it. **Priority 1**
- **R4.2.7** A lobby shall be able to be private (password-protected) or public. **Priority 4**

- **R4.2.8** Once two players are in the lobby, both players shall need to "ready-up" before the match is started. **Priority 2**
- **R4.2.9** After a match is completed, both players shall return to the lobby. **Priority 2**

4.3 Environment

R4.3 Checkers Board

- **R4.3.1** The board shall consist of 64 tiles in an 8x8 layout. **Priority 1**
- **R4.3.2** The board shall have 32 dark and 32 light tiles in an alternating pattern. **Priority 1**
- **R4.3.3** The bottom right-most tile shall be a light tile for both players. **Priority 1**
- **R4.3.4** The board shall have 12 men of each color at start of game. **Priority 1**

R4.3 Players

- **R4.3.5** A match shall consist of two players. **Priority 1**
- **R4.3.6** One player shall control the dark pieces, and the other shall control the light pieces. **Priority 1**

4.4 Start of Game

R4.4 Piece Placement

- **R4.4.1** At start of game, the men for each player will be placed on the dark tiles in the first three rows of their respective sides. **Priority 1**
- **R4.4.2** The middle two rows shall remain empty until a player makes a move. **Priority 1**

R4.4 Match Initialization

- **R4.4.3** A player shall randomly be chosen to take the first turn. **Priority 2**

4.5 Gameplay

R4.5 Taking Turns

- **R4.5.1** Turns shall be in alternating order. **Priority 1**
- **R4.5.2** A player shall not be able to choose to pass if there is a move available. **Priority 2**
- **R4.5.3** If a player fails to take a turn after a long amount of time, they shall pass their turn. **Priority 3**
- **R4.5.4** If all of a player's available moves are blocked, they shall pass their turn without making any moves. **Priority 3**

R4.5 Piece Movement

- **R4.5.5** A man shall always be moved diagonally forward. **Priority 1**
- **R4.5.6** A king shall be able to be moved diagonally backward or forward. **Priority 1**

- **R4.5.7** A piece shall either be moved one tile into an empty tile, or capture an opposing piece. **Priority 1**
- **R4.5.8** A piece can be moved again on the same turn if it captured a piece. This can continue until there are no pieces left to capture or the player chooses to end their turn. **Priority 2**

R4.5 Capturing

- **R4.5.9** The following criteria shall be fulfilled for a capture to take place: **Priority 1**
    - The capturing piece must be one tile away on the diagonal from an opponent piece (the captured piece). If the capturing piece is a king, the captured piece can be behind the king.
    - The player must be able to move the capturing piece two tiles diagonally such that it ends up on the opposite side of the captured piece.
    - There must be an empty tile on the opposite side of the captured piece.
- **R4.5.10** After capturing once, the capturing piece can continue capturing other pieces. The capturing piece shall be able to change direction while capturing multiple pieces. **Priority 2**
- **R4.5.11** After a capture, only the capturing pieces may be moved again. **Priority 1**
- **R4.5.12** Any captured pieces shall be removed from the board at the end of the turn and be added to the player's pile. **Priority 1**
- **R4.5.13** Since captured pieces remain until the end of the turn, a king shall be able to capture a piece twice. A man shall not be able to do this, since it can only move forward. Capturing a piece twice gives no extra scoring benefit. **Priority 3**

R4.5 Kinging

- **R4.5.14** A man that has reached the opponents first row of the board becomes a king on that same turn. **Priority 1**
- **R4.5.15** A king shall be able to move diagonally backward. **Priority 1**

4.6 End of Match

- **R4.6.1** A match shall end when a player has no remaining pieces on the board. **Priority 1**
- **R4.6.2** At the end of a match, the player with pieces remaining on the board is the winner. **Priority 1**
- **R4.6.3** The player that is not the winner loses the match. **Priority 1**
- **R4.6.4** Either player shall be able to choose to leave the match at any point during the match. This player is the loser, and the opposing player wins the match. **Priority 2**

5. Non-Functional Requirements

5.1 Network Performance

R5.1.1 Proton Reliability

Due to our networking choice using a dedicated server, we believe that any network lag in the game should be fairly minimal, as the only networked interaction will occur when passing the control back and forth between the players. Also, referring back to the dedicated server, if lag does occur, it should only affect the lagging player, creating an immediately more stable experience than peer-to-peer networking. Regardless, To ensure game quality, we will be playtesting the game before it's released multiple times, and we'll survey the playtesters to guarantee a more than acceptable quality for the game. **Priority 1**

R5.1.2 Proton Scalability

We want our game to have a healthy option with its scalability, and with our current project's free subscription, the game will be able to support 20 concurrent players out of the box. However, thankfully, due to our networking API's flexibility, the player limit can be scaled up to 2000 simultaneous players with just the click of a button. While minor, we believe this scalability provides us a comfortable position where we can practically guarantee that any small userbase we might gather will be satisfied with their quick and reliable online gameplay. **Priority 1**

5.2 Operating System Requirements

R5.2.1 Supported Platforms

The game is expected to support Windows 7, 8.1, and 10 along with Linux. Other Operating Systems will not be tested and built for, so they are not expected to work with Overlord Supreme. **Priority 1**

5.3 Availability

R5.3.1 Client Download

The Overlord Supreme client application will be downloadable from our *Github* releases. **Priority 1**

5.4 Security

R5.4.1 User Privacy

Overlord Supreme will collect no data about the user running the game, as we believe in allowing our players to use our service without getting something back from them. In furthering our considerations about privacy, the networking tool Proton itself collects no data about the user aside from their IP and their unique Proton authentication code. **Priority 1**

R5.4.2 Network Authentication / Encryption

To connect to another player using Proton, all we ask of the player is their authentication key (which should be automatically generated on game boot). From there, that authentication gets passed along to Proton as encrypted data, keeping the details of the connection safe from any potential outside attackers. **Priority 2**

5.5 Usability

R5.5.1 Playtesting

At each stable release of Overlord Supreme, the team will attempt to run a minor playtesting session with between 5 to 10 users. Our goal with these playtesting sessions is to pull players that are mostly unfamiliar with the project to get their unbiased opinion about the game's current functionality and enjoyability. To make their reports more actionable, we will also be providing them with a form to fill out so that we can better analyze any found shortcomings of the product. **Priority 2**

5.6 Maintainability

R5.6.1 Potential Future of Updates

As time goes on, our game will likely begin to have networking issues as the age of the version of Photon we're using increases. However, due to the excellent documentation on the API, future updates to resolve these issues should be incredibly simple, as they seem to provide a per-release update guide, making it very easy to refactor the code and push out the update in lightning speed. **Priority 2**

6. User Interface

6.0 Note

For this section, svg graphics will be used to illustrate a Checkers board and the various hints and effects anticipated on that board.

HUD and menu elements are drawn using bitmap graphics.

We refer to the User as Player (given game terminology), and Indicate Major Concepts or References in Capitals. We refer to the Application as Game.

Pieces and Tiles can provide colors as hints, mostly to show if they are Selected or Available. The following is used for hints:

- **Available:** *Blue*, such as on-mouse-hover, or selecting a path
- **Selected:** *Green*, such as on-mouse-click, or end of path
- **Threatened:** *Red*, such as attacked pieces, or attacking squares

See 6.10.3 Example Highlights for some of these.

6.1 Elements

The following are recognized as part of the user interface:

- The **6.2 Checkers Board**
- The **6.3 Checkers Pieces**
- The **6.4 Menu Flow**
- The **6.5 Heads-Up Display**
- The **6.6 Main Menu**
- The **6.7 Pause Menu**
- The **6.8 Settings Menu**
- The **6.9 Lobby Menu**

Elements will reference diagrams from the 6.10 Diagrams section.

R6.1.0 There is a Board **Priority 1**

R6.1.1 There is a complete set of Red and White Pieces **Priority 1**

R6.1.2 There is a Menu Flow **Priority 1**

R6.1.3 There is a Heads-Up Display **Priority 3**

R6.1.4 There is a Main Menu **Priority 1**

R6.1.5 There is a Pause Menu **Priority 3**

R6.1.6 There is a Settings Menu **Priority 3**

R6.1.7 There is a Lobby Menu **Priority 1**

6.2 Checkers Board

The Board is rendered to the screen as an alternating sequence of red and black squares called Tiles. See 6.10.1 Blank Checkers Board.

Each Tile can give the Player a Hint, similarly to the Pieces.

Pieces occupy Tiles, and indicate such by rendering over them. It is very important that the logical position ("A-2") corresponds to the on-screen position ("256 x 122") when it is rendered.

See 6.10.5 Selecting a Piece and Path for an interaction of the Board hinting to the Player the Path they are taking.

R6.2.0 Empty Tiles do not render any elements within them **Priority 1**

R6.2.1 Pieces render above correct tiles **Priority 1**

R6.2.2 Tiles hint at being Part of Path **Priority 2**

R6.2.3 Tiles hint at being End of Path **Priority 2**

R6.2.4 Tiles hint at being threatened **Priority 3**

R6.2.5 Board is 8 Tiles by 8 Tiles **Priority 1**

R6.2.6 Board is rendered across most of screen **Priority 1**

R6.2.7 No important part of Board is rendered Off-Screen **Priority 1**

R6.2.8 Inactive Tiles do not hint at anything **Priority 2**

6.3 Checkers Pieces

These are the individual tokens that move around on the Board. They are Men and Kings, with particular movement rules covered elsewhere.

They can be:

- **(None):** No special statuses or effects
- **Hovered-Over:** Inspect their Availability
- **Selected:** By clicking on it
- **Threatened:** By a piece when drawing move path
- **Moved:** To a new location on the Board
- **Removed:** Stop rendering
- **Hinted:** The *Red*, *Green*, and *Blue* colors

See 6.10.4 Hovering Over a Piece for an example interaction with a piece.

See 6.10.2 Initial Setup Positions for an example of logical placement.

R6.3.0 Pieces render to the Board **Priority 1**

R6.3.1 Pieces render to Logical Positions **Priority 1**

R6.3.2 Man Pieces distinct from King Pieces **Priority 1**

R6.3.3 Pieces hint at being Selectable **Priority 2**

R6.3.4 Pieces hint at being Selected **Priority 2**

R6.3.5 Pieces hint at being Threatened **Priority 3**

R6.3.6 White Pieces are distinct from Red Pieces **Priority 1**

R6.3.7 Piece positions render to correct location after Moving **Priority 1**

R6.3.8 Piece positions render to correct location during Game Start **Priority 1**

6.4 Menu Flow

This is how the Player is anticipated to flow between each menu.

See 6.10.0 Menu Flow.

R6.4.0 Player can open Main Menu on Game Start **Priority 1**

R6.4.1 Player can Exit Game from Main Menu **Priority 2**

R6.4.2 Exit Game closes the Game **Priority 2**

R6.4.3 Player can open Settings Menu from Main Menu **Priority 2**

R6.4.4 Player can Return To Last Menu from Settings Menu **Priority 1**

R6.4.5 Player can open Lobby Menu from Main Menu **Priority 1**

R6.4.6 Player can Create Game from Lobby Menu **Priority 1**

R6.4.7 Player can Join Game from Lobby Menu **Priority 1**

R6.4.8 Player can load Main Scene from Creating Game **Priority 1**

R6.4.9 Player can load Main Scene from Joining Game **Priority 1**

R6.4.10 Player can End Game from Main Scene **Priority 2**

R6.4.11 Player can open Pause Menu from Main Scene **Priority 3**

R6.4.12 Player can open Settings Menu from Pause Menu **Priority 3**

R6.4.13 Player can return to Pause Menu from Settings Menu **Priority 3**

R6.4.14 Player can return to Main Scene from Pause Menu **Priority 1**

R6.4.15 Player can return to Main Menu from Pause Menu **Priority 3**

R6.4.16 Player can return to Main Menu from End Game **Priority 3**

6.5 Heads-Up Display

These are on-screen, flat, 2D elements that overlay any other elements. In a 3D game, these are often ammo counters, health meters, and timers.

In *Supreme Checkers*, these are:

- The **Turn Controls**
- The **Game Clock**
- Any **Accessibility Features** (such as an escape-key alternative)

See 6.10.8 Heads Up Display for an example.

R6.5.0 HUD is updated in real-time **Priority 1**

R6.5.1 HUD responds to Player mouse-clicks **Priority 1**

R6.5.2 An "ESC" HUD element exists for mouse-only Players **Priority 3**

R6.5.3 "ESC" opens the Pause Menu **Priority 1**

R6.5.4 An "End Turn" HUD element exists **Priority 3**

R6.5.5 "End Turn" ends the Player's Turn **Priority 1**

6.6 Main Menu

The Main Menu lets the Player know the Game launched successfully and connect to the Lobby Menu.

R6.6.0 Launching Game opens the Main Menu **Priority 1**

R6.6.1 Main Menu has "Exit" Button **Priority 1**

R6.6.2 Main Menu has "Create Game" Button **Priority 1**

R6.6.3 Main Menu has "Join Game" Button **Priority 1**

R6.6.4 "Create Game" launches the Lobby Menu with Creation **Priority 1**

R6.6.5 "Join Game" launches the Lobby Menu with Joining **Priority 1**

R6.6.6 "Exit" closes the Game **Priority 2**

6.7 Pause Menu

The Pause Menu lets Players adjust Settings and exit the Game during play.

See 6.10.6 Pause Menu.

R6.7.0 Pause Menu renders to the screen when open **Priority 1**

R6.7.1 Pause Menu stops rendering to the screen when closed **Priority 1**

R6.7.2 "Exit" Button to Main Menu exists **Priority 1**

R6.7.3 "Settings" Button to Settings Menu exists **Priority 2**

R6.7.4 "Resume" Button to Main Scene exists **Priority 1**

R6.7.5 Selecting "Resume" closes Pause Menu and returns to Main Scene **Priority 1**

R6.7.6 Pressing "Esc" closes Pause Menu and returns to Main Scene **Priority 3**

R6.7.7 Selecting "Exit" closes Pause Menu and Main Scene and returns to Main Menu **Priority 1**

R6.7.8 Selecting "Settings" opens Settings Menu **Priority 1**

R6.7.9 Selecting "Resume" closes the Pause Menu **Priority 1**

6.8 Settings

The Settings Menu include accessibility and debug options, and serves as a space to add future options.

See 6.10.7 Settings Menu for an example layout.

R6.8.0 Closing the Settings Menu stops rendering the Settings Menu, updates the Game with the new values, and returns to the Last Screen **Priority 1**

R6.8.1 "Ok" Button exists **Priority 1**

R6.8.2 "Cancel" Button exists **Priority 1**

R6.8.3 Selecting "Ok" closes the Settings Menu **Priority 1**

R6.8.4 Selecting "Cancel" closes the Settings Menu without Applying Changes **Priority 1**

R6.8.5 Any included Options render correctly and apply effects correctly **Priority 3**

6.9 Lobby Menu

This gives Players a chance to discover over matches and other players. It should be a list of open matches, with names.

See 6.10.9 Lobby Menu for example.

R6.9.0 Lobby Menu updated and re-rendered frequently **Priority 1**

R6.9.1 Joining Lobby Menu does not create a New Game **Priority 1**

R6.9.2 Creating Lobby Menu creates a New Game and add it to the list of Games **Priority 1**

R6.9.3 Games can be Joined from the Lobby Menu **Priority 1**

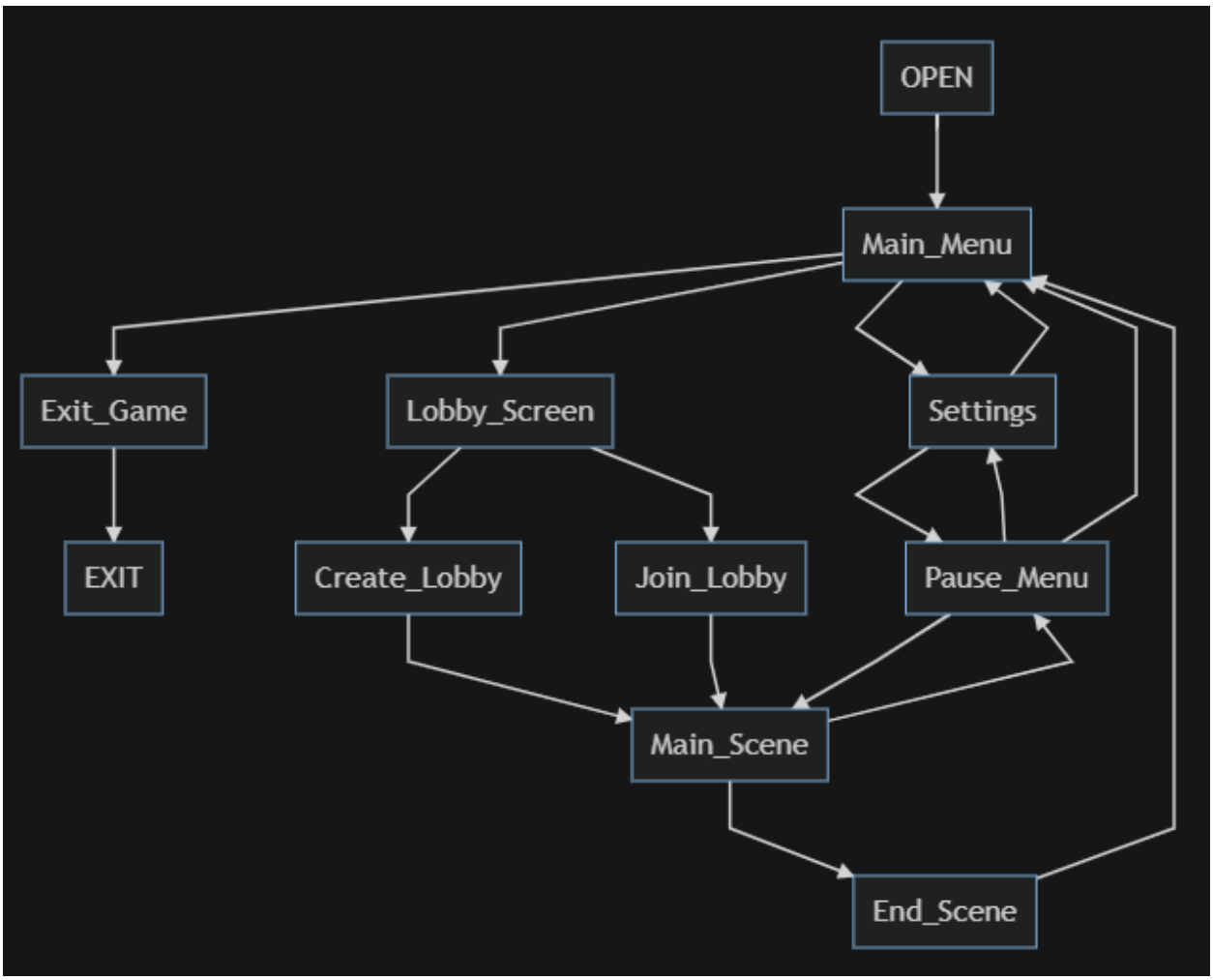
R6.9.4 Games can be Private/Closed **Priority 3**

R6.9.5 Lobby Menu renders to screen on opening **Priority 1**

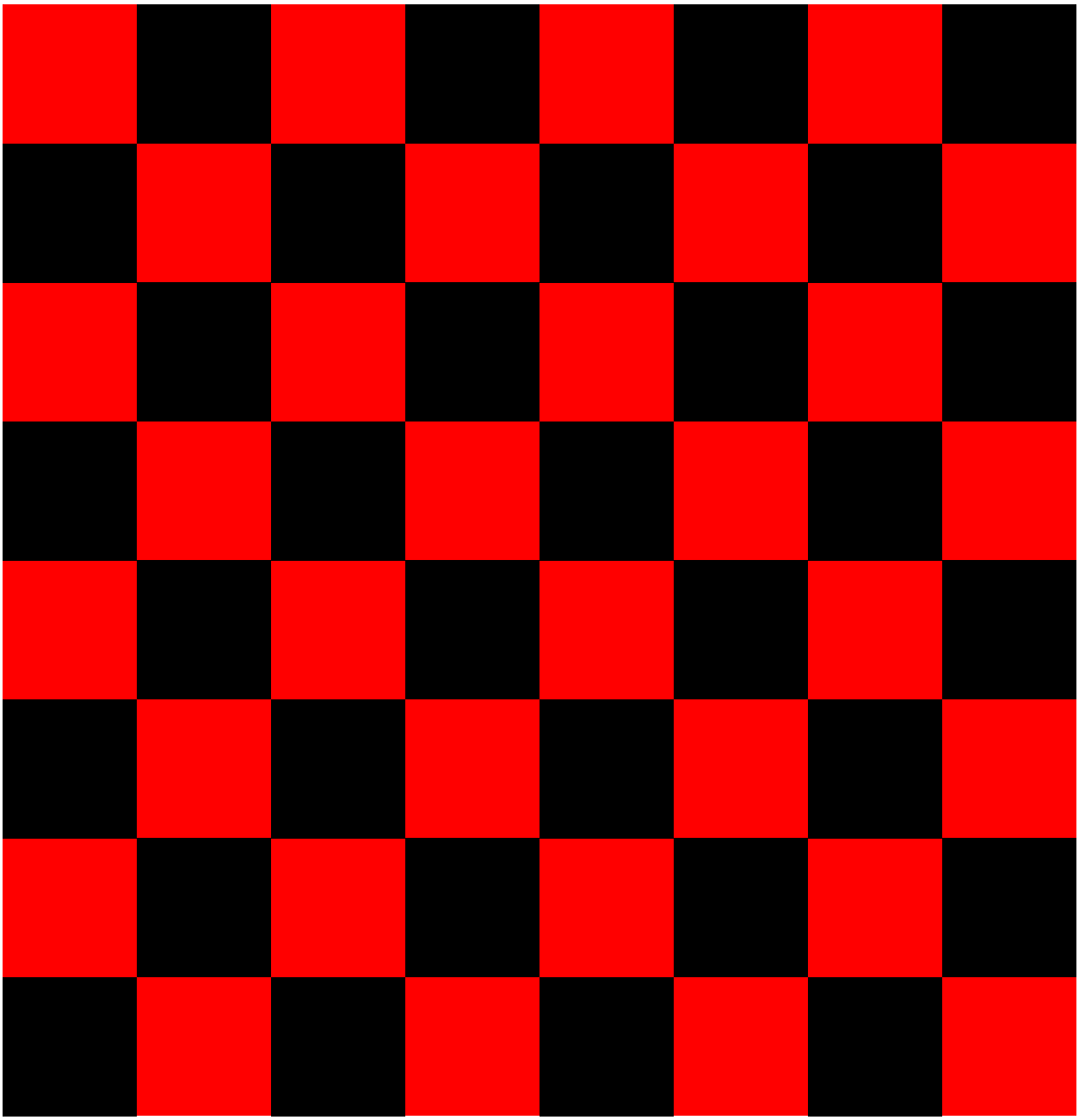
R6.9.6 Lobby Menu stops rendering to screen on close **Priority 1**

6.10 Diagrams

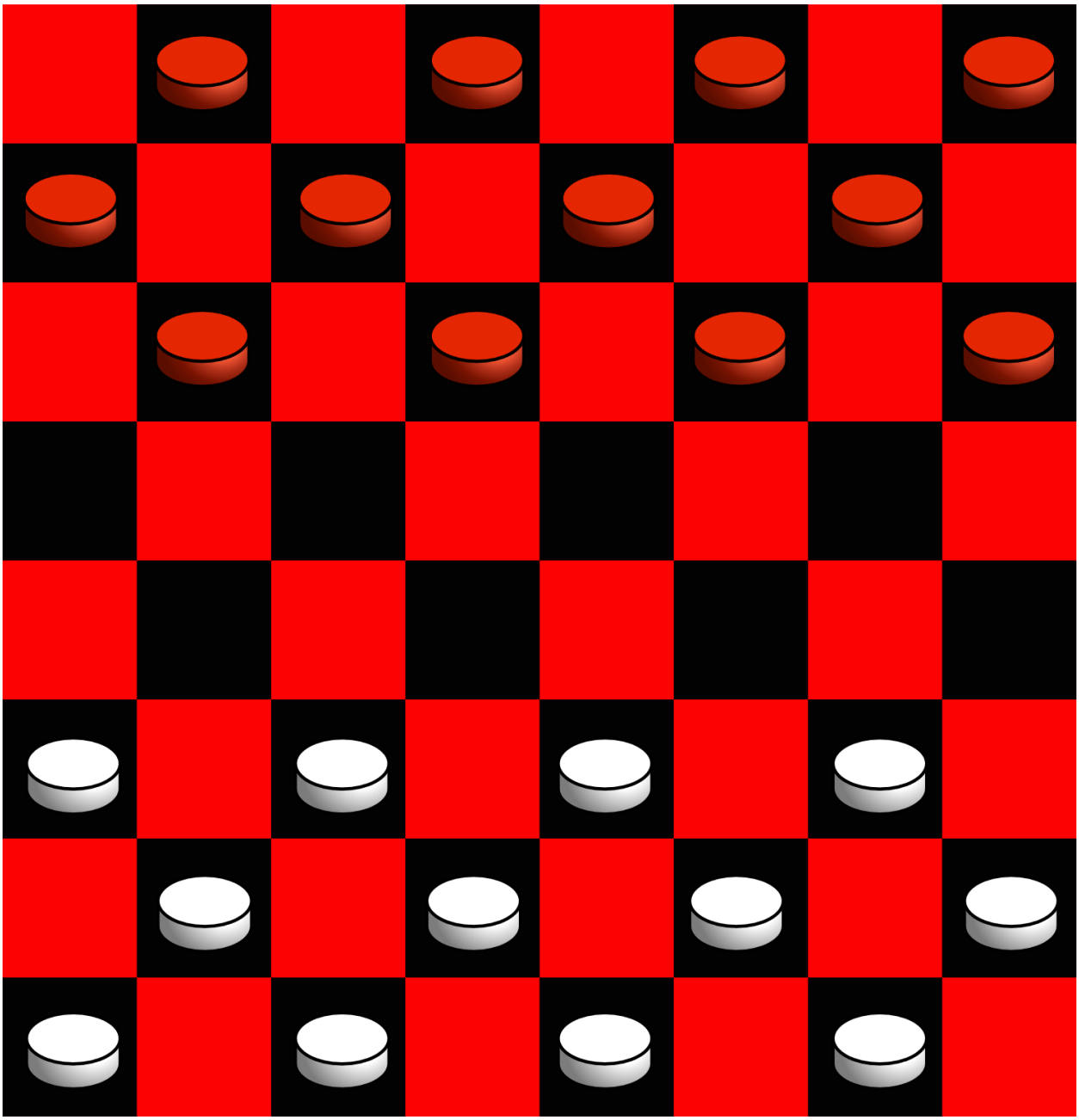
6.10.0 Menu Flow



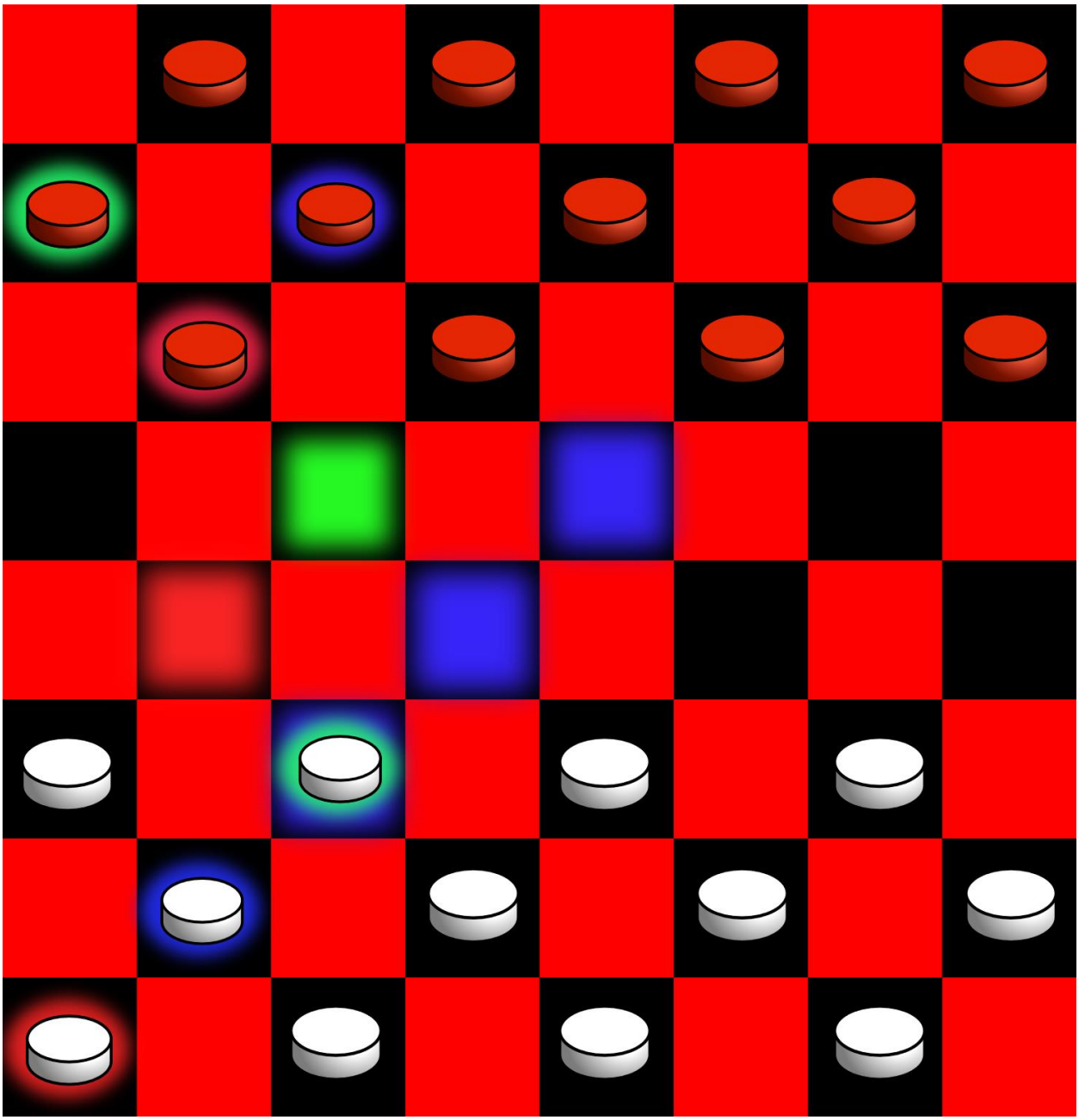
6.10.1 Blank Checkers Board



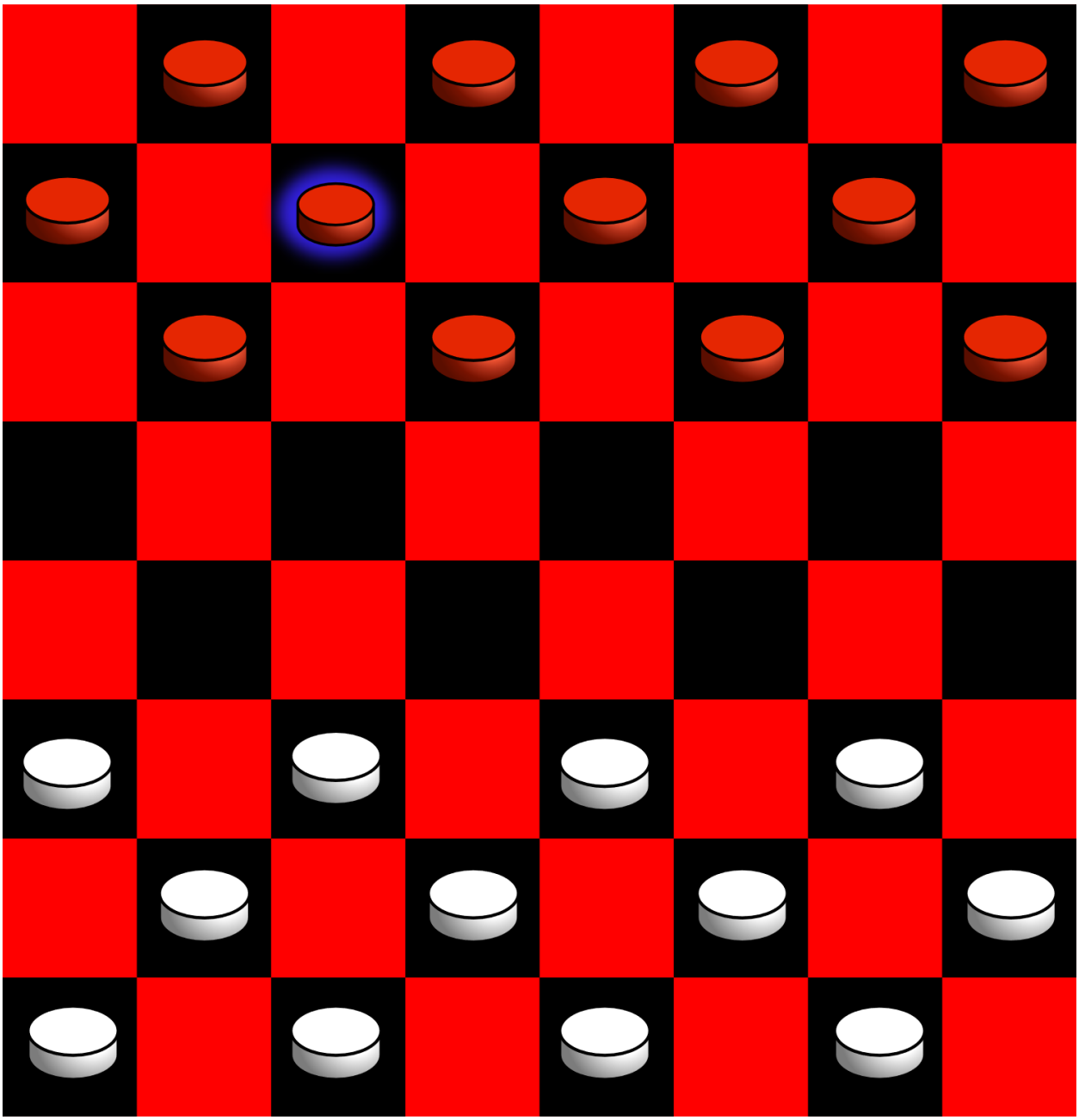
6.10.2 Initial Setup Positions



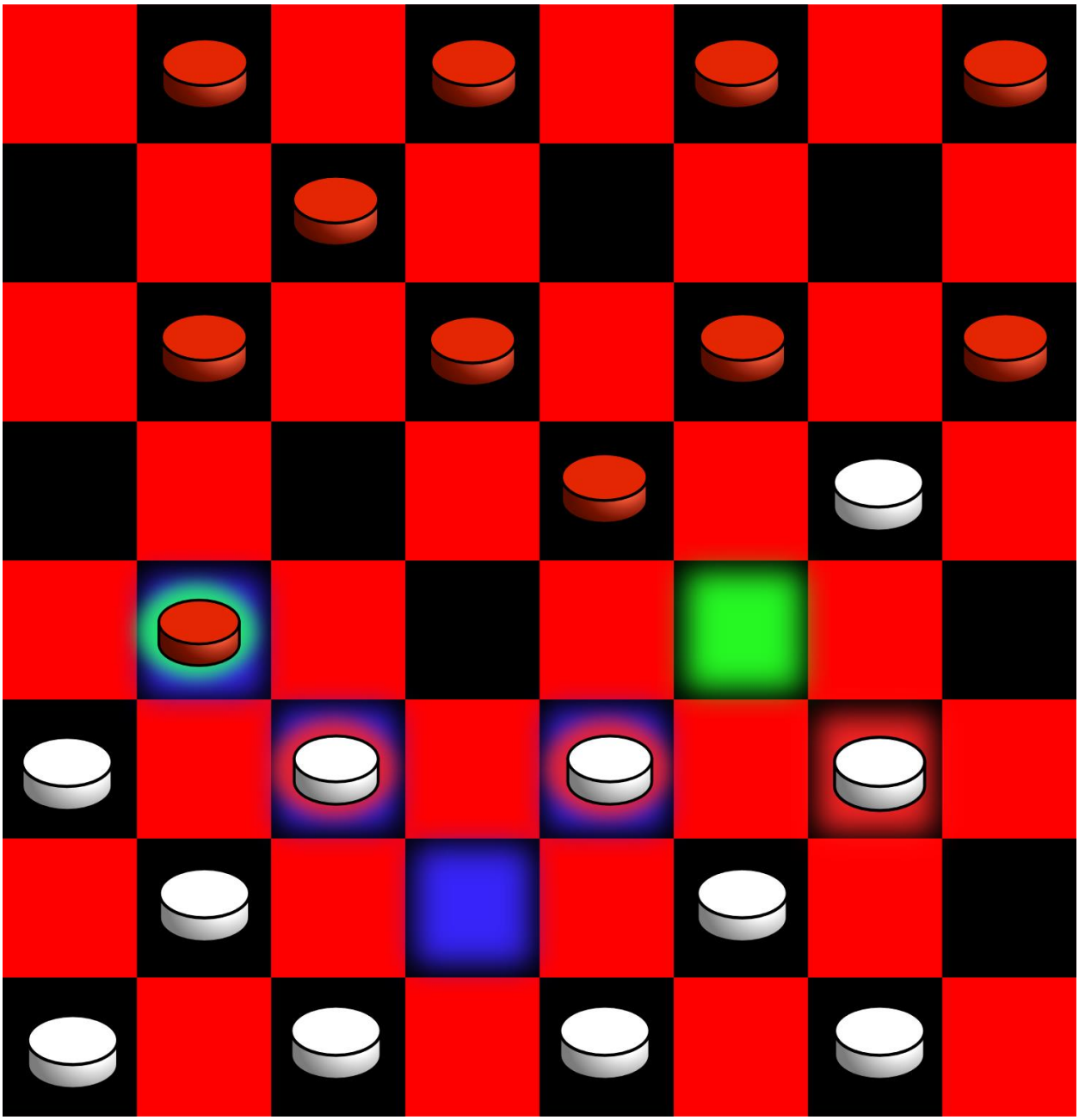
6.10.3 Example Highlights



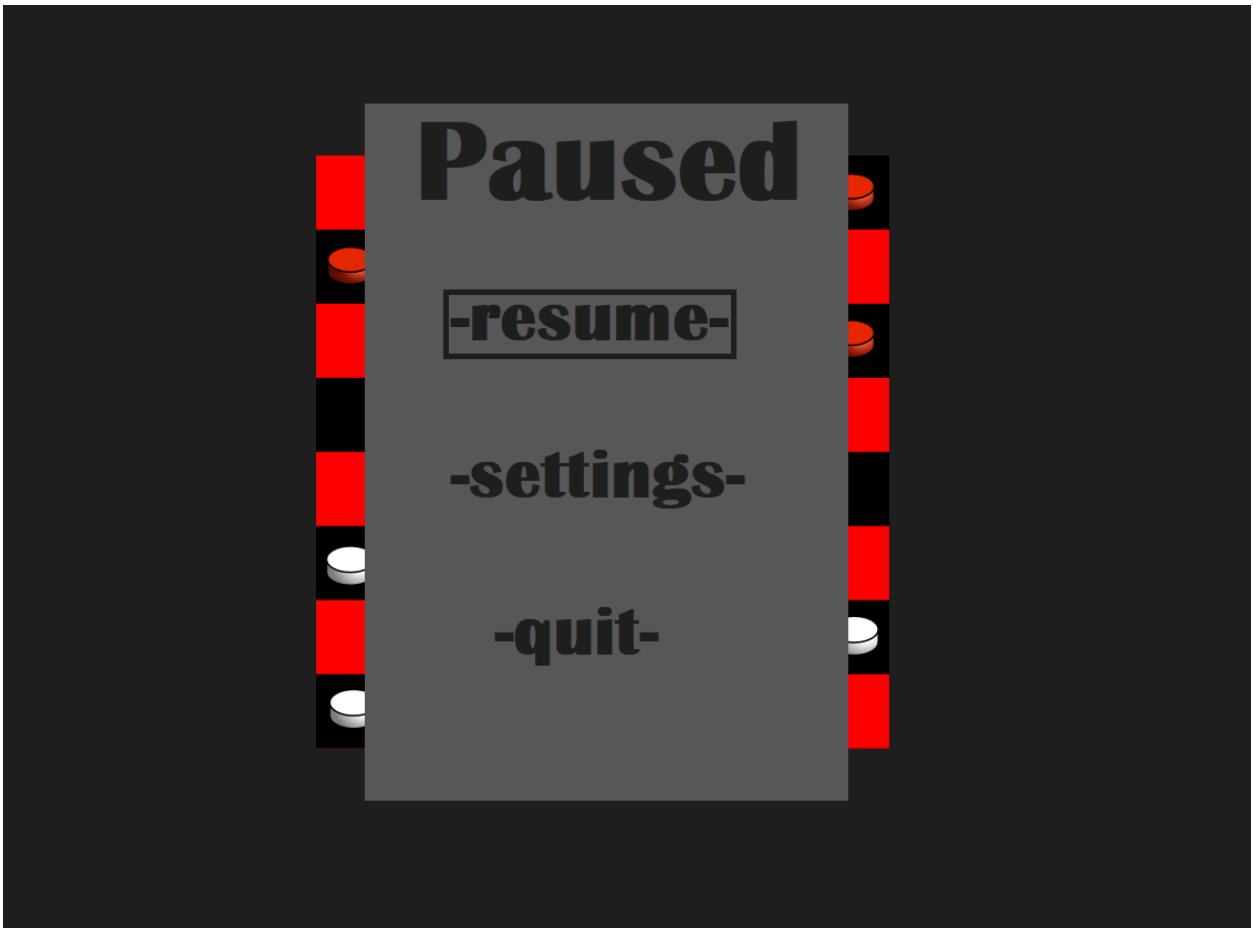
6.10.4 Hovering Over a Piece



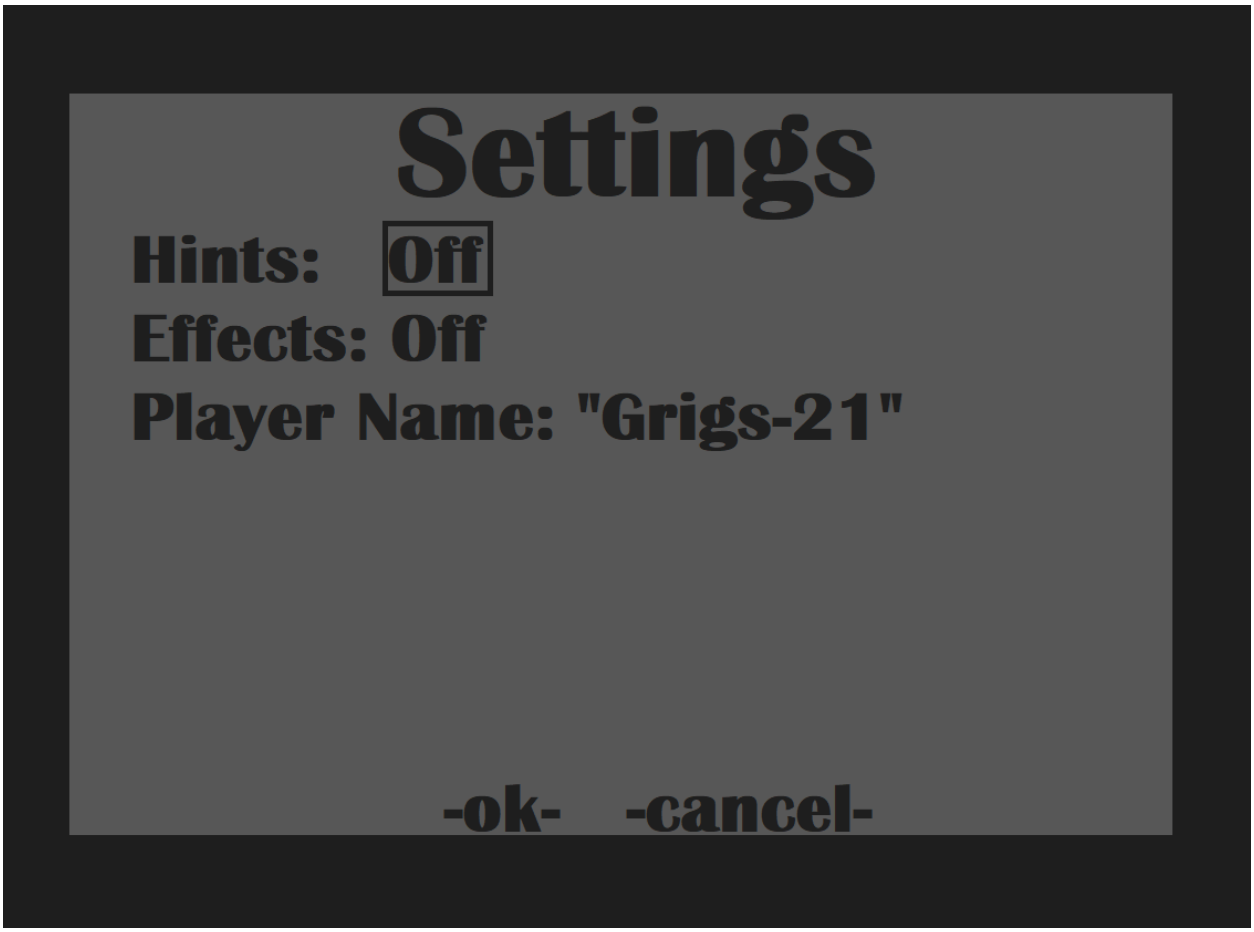
6.10.5 Selecting a Piece and Path



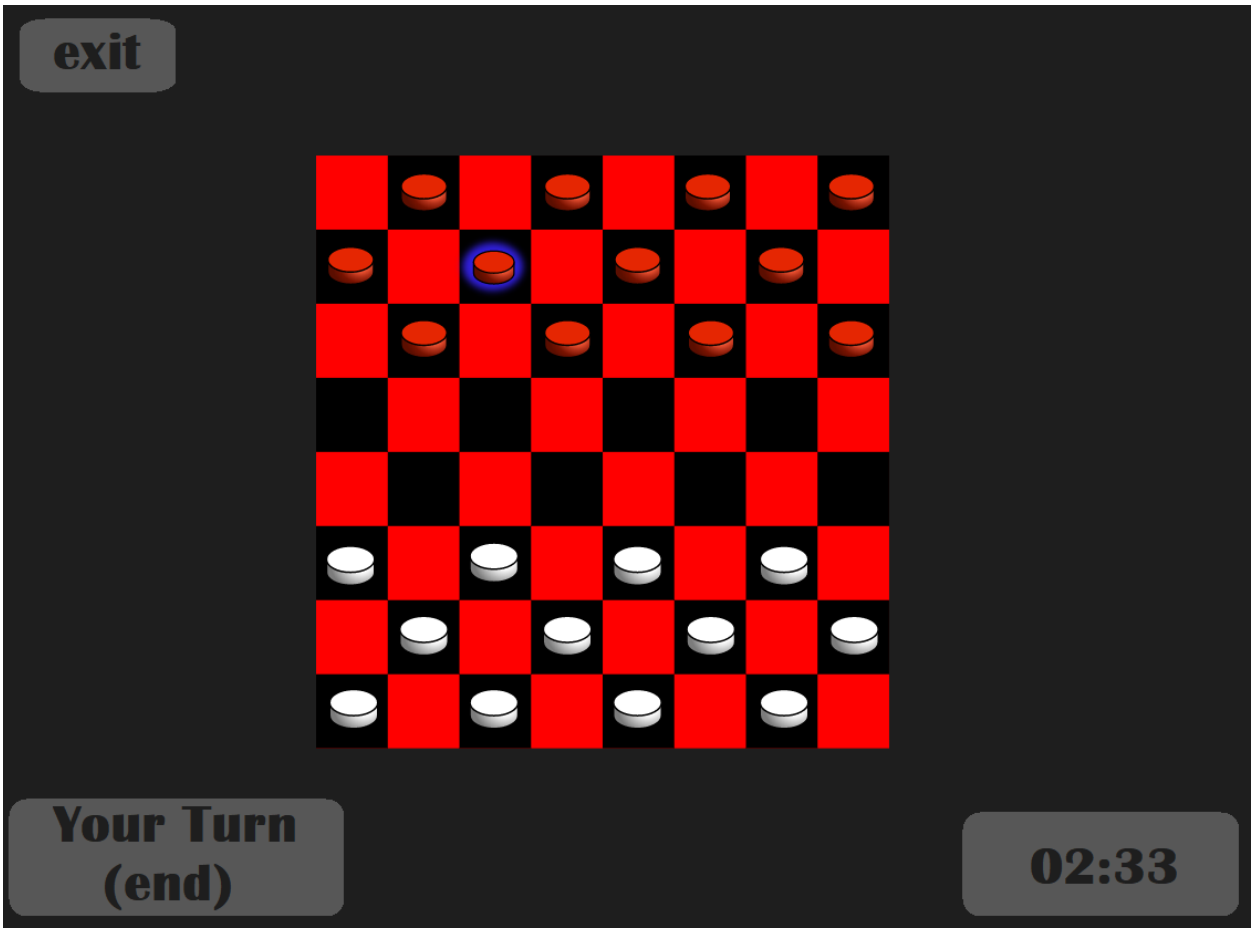
6.10.6 Pause Menu



6.10.7 Settings Menu



6.10.8 Heads Up Display



6.10.9 Lobby Menu

7. Use Cases

7.1 Use Case Flow

7.1.1 Starting Game

- **Precondition**: The player is on the title screen.
- **Action**: The player clicks "New Game". They may optionally set a password for their game lobby.
- **Postcondition**: The player will be added into a new lobby, which can have at max one other player. If a password is inputted, the lobby will be marked as private.

7.1.2 Joining Game (Client)

- **Precondition**: The player is on the title screen.
- **Action**: The player clicks "Join Game" and chooses an available lobby. If it is private, then they must enter the correct password.
- **Postcondition**: The player will join the lobby, and the game will automatically start with one of the players randomly being chosen to go first.

7.1.3 Ending A Checkers Game

- **Precondition**: A game must be in-progress.
- **Action**: Either of the two players exits or disconnects from the game, or one of the players has captured the other player's final piece.
- **Postcondition**: The game will end, and both players will return to the lobby, or just one player if the other player disconnects.

7.1.4 Exiting the Game Application

- **Precondition**: The player is on the title screen.
- **Action**: The player clicks on "Exit".
- **Postcondition**: The game application closes.

7.1.5 Capturing Pieces

- **Precondition**: It is the player's turn to move, and the player can make a valid move to capture one or more pieces using either a Man or King piece.
- **Action**: A player can click on the checkers piece, and click on a empty diagonal square, crossing over an opponent's piece, as long as it is a valid move. If the player is able to repeat this process and capture a second piece with that same checkers piece, the player may do so. If the player does not wish to do so, they can click on the "End Turn" button to end the turn.
- **Postcondition**: The server registers the player's move and updates the other player's board to reflect the current board state.

7.1.6 Moving a Man Piece Without Capturing

- **Precondition**: The player has selected a Man piece.
- **Action**: The player selects an empty forward diagonal square.
- **Precondition**: The Man piece moves to the assigned square and the board state is updated.

7.1.7 Moving a King Piece Without Capturing

- **Precondition**: The player has selected a King piece
- **Action**: The player selects an empty forward or backwards diagonal square.
- **Precondition**: The King piece moves to the assigned square and the board state is updated.

7.1.8 Ending Your Turn

- **Precondition**: The player has made a valid move. A player may not pass their turn without making a valid move.
- **Action**: The player clicks on the "End turn" button.
- **Postcondition**: The server registers the move that the player made, and updates the other player's board to reflect the current board state.

7.1.9 Winning The Game

- **Precondition**: The other player has left or disconnected from the game, or the other player has no remaining pieces, or the other player has no valid moves.
- **Action**: The player wins the game.
- **Postcondition**: Both players will automatically return to the lobby, or just the winning player if the other player has disconnected.

7.1.10 Turning a Man Piece into a King Piece

- **Precondition**: A player's Man piece has moved to the opponent's closest row relative to the opponent's view of the board.
- **Action**: The Man piece turns into a King piece
- **Postcondition**: The newly promoted King piece will now be able to move diagonally backwards.

7.2 Activity Diagram