**Average Othello**

**Tony Nguyen**

**Johnathan Kneice**

**Luis Garcia**

**Henry Dyer**

**Table of Contents**

[**1. Introduction**](#_epdi30d2ath4) **3**

[1.1. Purpose](#_teathxjrz06u) 3

[1.2. Intended Audience](#_17hz8hmwbxep) 4

[1.3. Scope](#_91hpz4y95t1y) 4

[1.4. Definition](#_ycv2l6d6po6n) 4

[1.5. Reference](#_9rww1cxjob17) 5

[**2. Overall description**](#_nao4fiabutiw) **5**

[2.1. Game Rules](#_54qxiqs5wjs7) 6

[2.2. Product function](#_wn807hu0uf9i) 6

[2.3. User characteristics](#_hj5whf53k7) 6

[2.4. Dependencies on other systems or software](#_4736uwbz4xbp) 7

[**3. Specific requirements**](#_rvdh9dad26f5) **7**

[3.1. User interfaces](#_x0w32cj1nfbx) 7

[3.2. Hardware interfaces](#_k4jcwtvrga4) 7

[3.3. Software interfaces](#_l89n6ep3843x) 8

[3.4. Communication interfaces](#_acwvwl43hytn) 8

[**4. Functional Requirements**](#_mu9r85ud3ot7) **8**

[4.1. Start and End of Game](#_n65onm4mwnx6) 8

[4.2. Input and Output Processes](#_x2rkie9oe1yl) 9

[4.3 Playing Against an AI Opponent](#_e1athjj3j806) 9

[4.4 Playing Against a Local Opponent](#_ufini4d8z959) 10

[4.5 Playing Against a Network Opponent](#_kyz4e2rpyvji) 10

[4.6 Appearance](#_5e8pjugr1iv6) 10

[4.7 Menus](#_fx76wpu8ypl7) 11

[**5. Non-functional Requirements**](#_horx84nbw5yd) **11**

[5.1. Performance requirements](#_1h7g58e8iq7b) 11

[5.2. Safety Requirements](#_knftbtmqc0ti) 12

[5.3. Security Requirements](#_c5f9hsp94et8) 12

[5.4. Software quality requirements](#_snekyu92yzgg) 12

# 1. Introduction

This introduction should cover the overview of this SRS document. Furthermore, this document has a table of definitions and abbreviations.

## 1.1. Purpose

The purpose of this document is to provide a detailed description of the requirements for the “Average Othello” application. It will demonstrate the development of the program and will also show the constraints, interface, and interactions.

## 1.2. Intended Audience

This document is intended for people with a basic knowledge of Java and a basic knowledge of software engineering.

## 1.3. Scope

“Average Othello” is a desktop application that allows the user to play a virtual version of the board game Othello (and Reversi). The application should emulate the experience of playing Othello using the modern ruleset created by Goro Hasegawa in 1971. It should also provide the option to play with the older Reversi ruleset if the user so chooses.

Othello is a game played by two players on an 8 × 8 grid-based board. Each player is assigned a color, typically either black or white. Players take turns placing one piece of their color on the board in a spot on the grid adjacent to another piece of either color that has already been placed. If any piece of an opponent’s color is placed between two pieces of a player’s color–that is, a straight line can be drawn between two pieces of a player’s color, and there are pieces of an opponent’s color that fall on that line–then all pieces of the opponent’s color should be replaced with pieces of the player’s color. The objective of the game is to have more pieces of the board in your color than your opponent once all the spots on the board are filled with pieces.

The application should display an 8 × 8 grid-based board similar to that used in real-world games of Othello, where users should be able to place their pieces to play the game. The game should be playable using the mouse and left-click button to place pieces on the virtual game board. Upon the start of a game, the user should be given the option to play against an AI opponent, a real opponent using a singular instance of the application, and a real opponent with a second instance of the application running, where the two applications communicate via a network connection.

## 1.4. Definitions

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Reversi | Another name for Othello that uses an 8 × 8 board and 32 pieces for each player.  Reversi is the original game on which Othello is based. Othello is a specific ruleset variant of Reversi, though Reversi and Othello are often used as synonyms of each other. |
| User | Someone who is operating the game piece and the one who is interacting with the software  Someone who is interacting with the software. A user is sometimes also a player, but not always. For example, a user who is clicking a button on the application’s main menu is not a player because a game for the user to play has not yet been started. |
| Player | Someone who is interacting with the software in order to play a game of Othello. |
| AI (Artificial Intelligence) | Computer technology that tries to mimic human intelligence. In this document, it refers to the way the software simulates a player’s decision-making by placing pieces on the board in an algorithmically-decided manner in order to allow human players to play the game without another human present. |
| Network | A server that can connect to different devices. |
| Simultaneous multiplayer | Two or more players/users using the software on one device simultaneously. |
| Multiplayer | Two or more players/users using the software. |
| Single player | One player/user using the software. |
| OS (Operating System) | System software providing services for the computer program. |

## 1.5. Reference

**[1] World Othello Federation. “OFFICIAL RULES FOR THE GAME OTHELLO” *World Othello Federation.* 1 Sep 2022.**

**[2] Susan Weber. “Behind the Othello Board Game: History and Gameplay” *LOVE to KNOW*. 9 Sep 2022**

**[3] Home Rec World Editorial Team. “Othello vs. Reversi: Are They the Same?” *Home Rec World.*  8 Sep 2022.**

# 2. Overall description

This section will give an overview of the general software. This overview will also go into detail on the functions and interactions.

## 2.1. Game Rules

The rules for playing Othello start with the four main pieces in the middle. The player who goes first plays the black pieces, while the second player plays the white pieces. The main objective is to fill the board with your pieces and have the most pieces of your color in order to win. On a player’s turn, whenever they play a piece, any pieces of the opposing color between that played piece and another piece of the player’s color are turned into a piece of the player’s color, or “overtaken.” When it comes to the legal moves the user is allowed to do is ‘outflanking’ your opponent, meaning overtaking the pieces. Legal moves take pieces, and illegal moves are those that do not take a piece. In other words, if there is a straight line between the piece being played and another existing piece of that color, and there is at least one or more pieces of the opposing color in between those two pieces on that line, then the piece being played is legal. Moves are allowed forming horizontal, vertical, and diagonal straight lines. However, the user isn’t allowed to put a piece in an area with no piece is adjacent to it.

Please note that although the names Othello and Reversi are used interchangeably and have similar gameplay, both have rulesets that differ in key areas. For example, given a game state where a player is unable to play a piece during their turn, in Othello the turn would be passed to the other player. In Reversi, if no legal moves can be played during a player’s turn, then the game ends immediately, and the player with the most pieces on the board wins. Furthermore, the state at the start of the game is different for each ruleset. In Othello, it is mandatory to have the four starting pieces in the middle, two black and two white, with pieces of the same color diagonal from each other. In Reversi, the players can place their first piece on any space within the center four squares of the board, where the starting pieces would normally be placed for a game of Othello.

## 2.2. Product function

With this program running on computers, the users should be able to operate their game pieces by mouse. Please note that the user should be able to place their pieces in a free space adjacent to any other piece. After placing your piece, the user should no longer be able to affect the board until the opponent makes their move. Please note that the player using black pieces will always go first.

## 2.3. User characteristics

Each instance of the application will be used by either a single user, or two users simultaneously. Each user will have the same capabilities in any instance of the application. Users will be able to start a new game of Othello and select whether they want to play a singleplayer, multiplayer, or simultaneous multiplayer game. If a user selects to play a simultaneous multiplayer game, then a second user can play a game as an opponent against the first user, where each user takes turns operating the application using the same computer, mouse, and application instance.

## 2.4. Dependencies on other systems or software

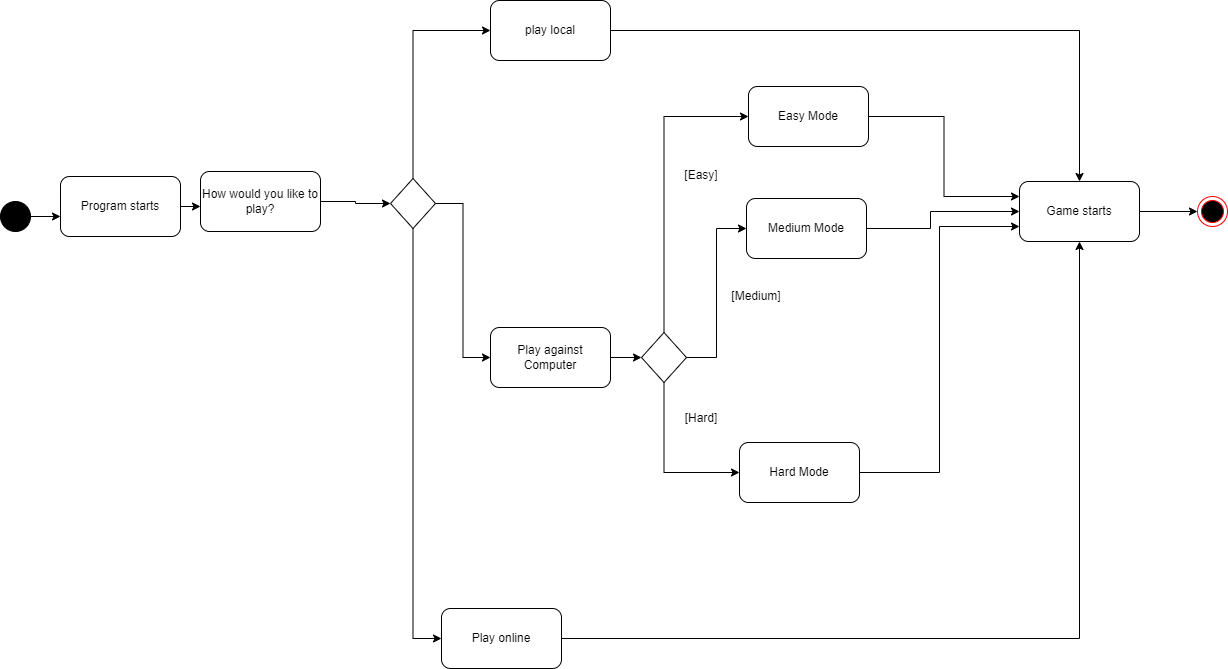
We will be using Windows OS as the main framework for this software.

# 3. Specific requirements

This section contains the functionality and requirements of our program. This section also gives an idea of what the interface will look like.

## 3.1. User interfaces

When the user first starts the program, it should show the user an 8 × 8 light green background and a dark green line grid board with two white pieces in the coordinates, (4,4),(5,5), and two black pieces in the coordinates (5,4), (4,5). The users would have a transparent gray circle in legal areas where they can place their pieces. Next, a pop-up will ask if they want to play local multiplayer, play online, or play against the computer. Selecting one of these will start the game. You can place pieces by clicking with your mouse on a location on the board. When no more moves can be made, a pop-up will announce the winner.



## 3.2. Hardware interfaces

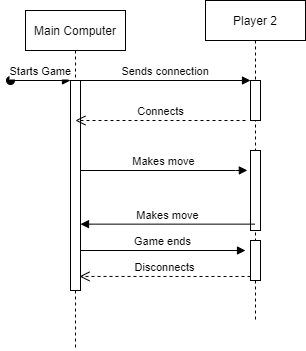
Our program requires a monitor, keyboard, mouse, and speaker. We hope to one day implement a touch screen.

## 3.3. Software interfaces

Our program will be supported on Windows/7/8/8.1/10/11, Mac, Linux, and Java.

## 3.4. Communication interfaces

We will have an online mode for two people to play if they both have the program running on their computers. One computer will act as the server allowing the other computer to connect.



# 4. Functional Requirements

This section contains the functional requirements of the software.

## 4.1. Start and End of Game

4.1.1. The program shall ask the user whether they want to play against an AI, against a human on the same computer, or against a human on another computer.

4.1.2. The program shall ask the user whether they would like to play with the Othello or Reversi ruleset.

4.1.3. The program shall end the game when all places on the board have been filled with pieces if the user has chosen the Othello ruleset.

4.1.4. The program shall end the game when one player is unable to make a move, or when all places on the board have been filled with pieces if the user has chosen the Reversi ruleset.

4.1.5. The program shall display the number of pieces of each color on the board and indicate which player has won at the end of the game.

4.1.6. The program shall ask the user whether or not they would like to play another game once the winner of the last game has been displayed.

4.1.7. The program shall start a new game if the user selects to play another game after the end of a game.

## 4.2. Input and Output Processes

4.2.1. The program shall display a placed piece on the board immediately after the player has chosen to place a piece in a permitted space.

4.2.2. The program shall not allow players to place pieces in spaces that are not allowed by the ruleset of the game.

4.2.3. The program shall pass a turn to the other player if the current player has no valid moves if the Othello ruleset has been chosen.

4.2.4. The program shall provide a short visual indication if a player's turn has been passed.

4.2.5. The program shall check which pieces should change color after a piece is placed, and immediately change those colors after a piece is placed.

4.2.6. The opponent should provide a visual indication of which spaces are available for a piece to be placed on during a player's turn.

## 4.3 Playing Against an AI Opponent

4.3.1. The program shall allow the user to choose between an Easy, Normal, and Hard difficulty setting for their AI opponent.

4.3.2. The program shall decide a location and place a piece on the board within one second of a player placing their piece on the board.

4.3.3. The program shall only place pieces in spaces permitted by the ruleset of the game, subject to the same limitations as human players.

4.3.4. The program shall give the player black pieces and thus shall always allow the player to go first.

## 4.4 Playing Against a Local Opponent

4.4.1. The program shall alternate between black and white pieces being placed on the board upon player input.

## 4.5 Playing Against a Network Opponent

4.5.1. The program shall form a network connection with another instance of the program that is running.

4.5.2. The program shall allow one player on one instance to place black pieces on the board, and shall allow the other player on the other instance to place white pieces on the board.

4.5.3. The program shall allow the player to place black pieces to make the first move.

4.5.4. The program shall only allow a player to place a piece after the other player has already placed a piece.

## 4.6 Appearance

4.6.1. The program shall display a green 8 × 8 grid board while in a game.

4.6.2. The program shall provide a visual indication of which moves are permitted to the player.

4.6.3. The program shall display each piece that is currently on the board in its corresponding position on the grid.

4.6.4. The program shall not change the color of pieces on the board unless a permitted piece placement would cause those pieces to change color.

4.6.5. The program shall display a counter of how many pieces of each color are currently on the board.

4.6.6. The program shall update the counter of how many pieces of each color are on the board immediately after a new piece is placed and the color of old pieces is changed as a result.

## 4.7 Menus

4.7.1. Upon the start of the program, the program shall display a "Main Menu" with three options for the user to choose from: Start Game, Join Game, and Options.

4.7.2. The program shall initiate the sequence for starting a game once the user selects "Start Game."

4.7.3. The program shall begin searching for another instance of the program to connect to, with a visual indication that it is doing so, once the user selects "Join Game."

4.7.4. The program shall display a list of options that allow the user to change the color of the player 1 (black) and player 2 (white) pieces and switch between Othello and Reversi rulesets, once the user selects "Options."

4.7.5. The program shall provide an option that allows the user to return to the previously shown menu on any menu that is displayed, except for the Main Menu.

# 5. Non-functional Requirements

This section contains the traits of the software.

## 5.1. Performance requirements

5.1.1 The program shall be run on local Java code.

5.1.2 The program shall depend on the strength of the internet connection for loading times.

## 5.2. Safety Requirements

5.2.1 The program shall have a database to save the game progress in case of crashing.

## 5.3. Security Requirements

5.3.1 This program is not secure.

## 5.4. Software quality requirements

5.4.1 The program shall start in the correct way and end in the correct way.

5.4.2 The program shall have an aesthetically pleasing interface for the user.

5.4.3 The program shall satisfy the user’s expectations for a game of Othello.

5.4.4 The program shall respond to the user’s actions immediately after they’re done.