Project Phase 1 - Requirements

Group: Ringtail

CMPT276 D100: Intro. To Software Engineering

Dr. Tao Wang

June 5th, 2023

Game Introduction

The game being implemented is Othello which is a strategy board game between two players. One player plays black and the other white. There are a total of 32 discs and the black player starts the game. The winning player is the one with the most discs at the end of the game. Starting with four discs, the discs increase and the winning strategy is to maximize the number of discs. The game alternates between black and white discs until either both players have no more moves left, or the board becomes filled.

The game will be implemented in Java as its library is feature-rich and will facilitate development of the game. There will be a multiplayer mode for players to play against each other. There will also be an AI mode where the player will be able to play against the computer with multiple difficulties, and can be potentially determined dynamically by player performance. Players can also play with AI-assistance where it will suggest the best moves based on predetermined criteria.

Accessibility features for color blind individuals, such as changing colors for the game board, the pieces, as well as labeling pieces with shapes to help distinguish piece ownership.

The game will have tutorials available for the player to peruse. For example, corners are worth more than side pieces - text files like these with all the information will be provided, and the player can use this feature to learn and hone their edge.

User stories

Scenario 1: User wants to know how to play the game
The main screen is displayed in front of the player, showing a lot of options: “Start Game”, “How to Play”, “Settings”. (Note: words in brackets might be named again later in real application.)

John would like a rundown on how the game works, thus he uses the tutorial feature to learn the basics by clicking on “How to Play”. Basically, there could be instructions for 2 groups of players: new players and experienced players. For those who are new, there will be a brief instruction which provides basic rules of the game with images (screenshots or animations of the game along with the text instructions for easier understanding), also features of this particular implementation (unique accessibility features in “Settings”). For players with experience, there will be several tutorials of advanced techniques to win the game in difficult levels.

Scenario 2: User wants to change some features/traits of the game (background, tiles)
John is a color-blind person. He wants to change the color of the tiles so that he can be able to distinguish which tile is his and which tile is opponent’s.

In the main screen, if John clicks on “Settings”, he will be able to customize the game in his own way in terms of the display of the game. He can customize the “background colors” of the board, “colors” and “shapes” of the tiles.

Default setting of the game will be: background color of the board (green), color and shape of tiles (black, white and circular 2D tile)

Scenario 3: User wants to change some features/traits of the game (timer)
John is also a careful and chill person, he does not like making quick decisions in any case. He wants to have more time to think so that he can relax, make a good move in his turn and also enjoy the game.

In the “Settings”, there is an option for John to customize his timer, he can either choose to have a timer for each player’s turn or not. If he sets up a timer, he can also adjust the amount of time.

Default setting for the timer will be: each player has 1 minute to make a move in their turn.

Scenario 4: User wants to play the game either alone or with a friend
John is on the main screen of the game now. When he clicks on “Start Game”, two game modes are displayed to him: “Player vs Player”, “Player vs AI”.

John wants to play this game with his friend so that he can have much more fun with them. John chooses “Player vs Player” mode, in this mode, John and his friend sit together in real life, playing the game on the same device.

(Note: this local play is our team’s current assumption, we might let user be able to invite their friend through social media or we can assign the player randomly with another player, those implementations will be decided later)

Scenario 5: User wants to change some features/traits of the game during the game session
John and his friend Jessica are in the game right now. Unlike John, Jessica is a bit indecisive. She now wants to change the background color of the board because she does not like the color that John chooses. They want to be able to make this change during the game session so that they do not have to start a new game.

There will be an option “Settings” in the game screen where adjustable features can be found in “Settings” (in the main screen) can also be found and adjusted here.

Scenario 6: User wants to have variety of difficulty
John wants to have a new, different, exciting but level-appropriate experience whenever he plays the game so that he will not get bored of or discouraged by the game.

In case John chooses to play with AI, three levels of difficulty are provided for him: “Easy”, “Medium”, “Hard”, “Dynamic”, in which “Dynamic” level means AI will adjust the difficulty

differently based on the move and the time (that player used to make the move) in each player's turn.

Scenario 7: User wants to leave during the game session
John is playing the game but he suddenly has a phone call, he needs to stop for a while to make a phone call. When he's done, he wants to continue playing from where he left off so that he does not need to start over a new game.

During a game session, John will have an option to "pause" the game, then he can choose to either "resume the game" or "exit with saving current game" or "exit without saving". If he chooses "exit with saving current game", there will be a check box "autosave for following times".

In case he clicks on the box, the following times when he chooses to "pause" the game, there will be only 2 options: "resume" or "exit". This functionality can also be changed in "Settings".

Scenario 8: User wants to continue the game they left last time
John and Jessica want to load the previous game that they left last time.

Because last time they did save their game state, so in the main screen, there is one more option called "resume".

Functional and Non-Functional Requirements

Functional Requirements

1. The game must mark and make legal moves visible to the current player on each turn.

Function	Marking legal moves
Description	Compute legal moves for the current player based on the current game state, and make them visible.
Input	Game state, including pieces placed, piece colour, and current player
Output	Markers on the board indicating valid moves.
Action	For each tile: If no adjacent tiles, no action Else: For each cardinal and ordinal direction: If there is another same color tile, and between them exists no empty space and at least one opposite colored tile, mark valid
Requires	Ongoing game, with at least 1 empty tile remaining

2. The game must allow players to choose a colour and display the board in said colour if specified.

Function	Accessibility options
Description	Allows users to change the color of the board background to facilitate gameplay for players with visual impairment.
Input	Color code
Output	Board background displayed in specified color
Action	Overwrite previous colour code in config, and display board background in new color.
Requires	None

3. The game must allow players to save and load any valid game state for the players to pause and resume play.

Function	Save
Description	Saves the current game state to memory for the user to resume gameplay later.
Input	Board state, elapsed time, current player
Output	Save file with valid game state
Action	Writes all aforementioned variables into a formatted file.
Requires	Ongoing game, with writable directory and free drive space

Function	Load
Description	Loads a previous save into the game, then resume play in a state where the game is left off at the time of save.
Input	Save file with valid game state
Output	Board state, elapsed time, current player
Action	Verify the validity of the file, then read game state variables and load into memory.
Requires	Valid, readable save file.

4. The game must play against the player in Player vs AI mode if the player elects to do so.

Function	Player vs AI
Description	Uses an algorithm to make decisions and play against the player.
Input	Difficulty level, player moves
Output	A sequence of decisions
Action	In each turn, decide the next move based on the current game state and the heuristic value of each move. The decision should be stochastic at lower level difficulties, with successively increasing chances of making optimal decisions at higher levels.
Requires	Ongoing game

Non-functional Requirements

1. During player vs AI, the AI shall take into consideration that border and corner tiles are more strategically advantageous than middle tiles.
2. During player vs. AI, if the difficulty is not specified, the game shall determine the difficulty automatically and update accordingly based on player performance throughout the course of each game session.
3. The game shall display the outcome and scores of each player at the end of each game session.
4. The game session shall end immediately if neither player has any legal moves left.

UML Diagram

RingTail

June 4, 2023