SOFTWARE DEVELOPMENT

OTHELLO PROGRAM

PREPARED BY
Reinaldo Michael Hasian

Sept 09, 2019

CBS Techno Corp
3-10-17 ITmeieki Bld.No.2, Meieki,
Nishi-ku Nagoya-shi, Aichi,
451-0045 Japan

Table of Contents

1. Preliminary

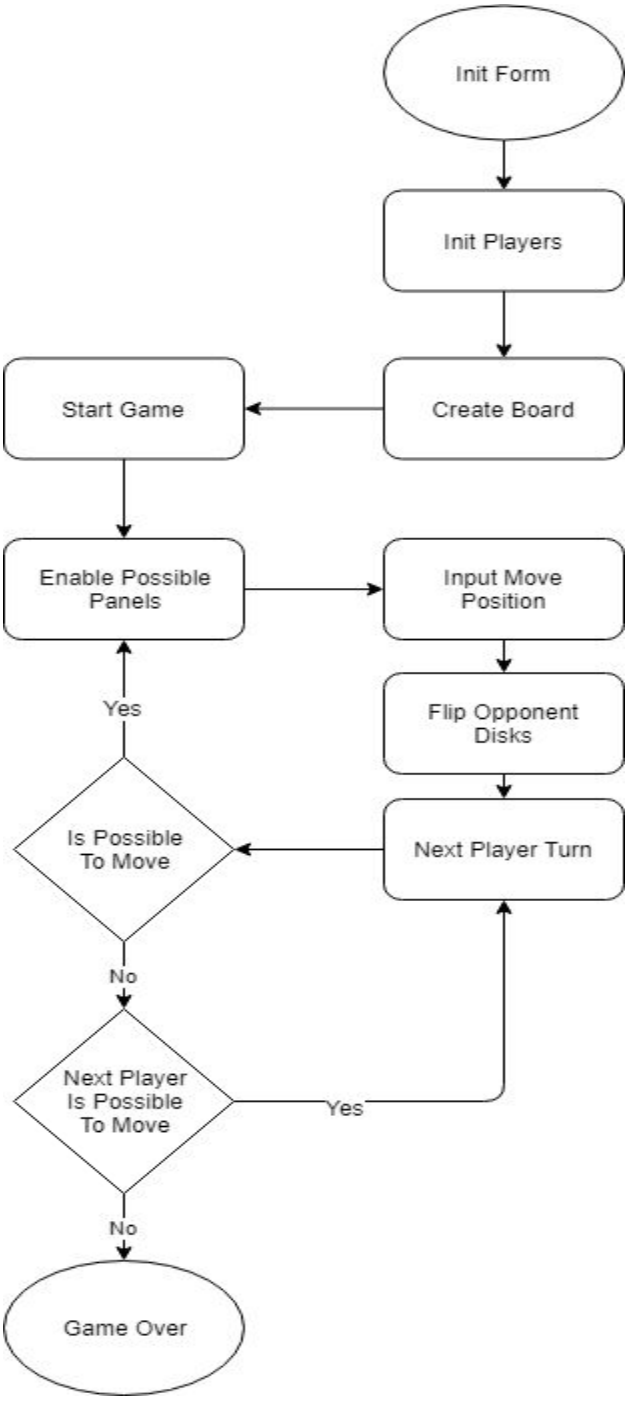
Othello or reversi is a strategy board game for two players, played on an 8×8 uncheckered board. There are sixty-four identical game pieces called *disks* (often spelled "discs"), which are light on one side and dark on the other. Players take turns placing disks on the board with their assigned color facing up. During a play, any disks of the opponent's color that are in a straight line and bounded by the disk just placed and another disk of the current player's color are turned over to the current player's color.

The object of the game is to have the majority of disks turned to display your color when the last playable empty square is filled.

This program is Othello board game implemented as windows form application with C# programming languange.

2. Algorithm Design

Below is flow chart of algorithm design that used to develop this Othello program.



Explanation for each process :

1. Init Form

   Create windows form to be shown on screen.

2. Init Players

   Create White Player and Black Player. Set an AI if user want play versus bot then set Black Player as first turn.

3. Create Board

   Add Othello board to windows from. Board builded with 8x8 panels.

4. Start Game

   Done initialization, execute the game logic.

5. Enable Possible Panels

   Based on current Player turn, enable all panels that possible for player to put disk. For each player acquired panels :

   - Looking for latest north green panel (that separated by enemy disk) and enable it if found
   - Looking for north east green panel (that separated by enemy disk) and enable it if found
   - Looking for east green panel (that separated by enemy disk) and enable it if found
   - Looking for south east green panel (that separated by enemy disk) and enable it if found
   - Looking for south green panel (that separated by enemy disk) and enable it if found
   - Looking for south west green panel (that separated by enemy disk) and enable it if found
   - Looking for west green panel (that separated by enemy disk) and enable it if found
   - Looking for north west green panel (that separated by enemy disk) and enable it if found

6. Input Move Position

   Set chosen panel as Player's panel.

7. Flip Opponent Disks

Flip all enemy disks that flanked by new disk from step 6.

8. Next Player Turn

   Set current turn to the next player then check if that player can move or not.
   If that player can move then back to step 5. If that player can not move, program will check if previous player can move or not. If previous player can move, then current turn will be previous player. But if previous player also can not move then game is over.

9. Game Over

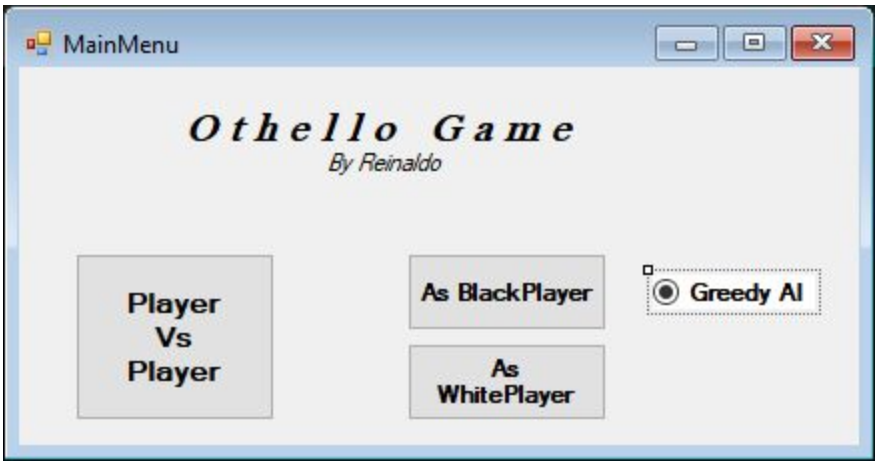
   Calculate each player disk and decide who is the winner. The winner shown at message box with each player score.

3. UI Design

This game application is developed with windows form visual interface. Consist of :

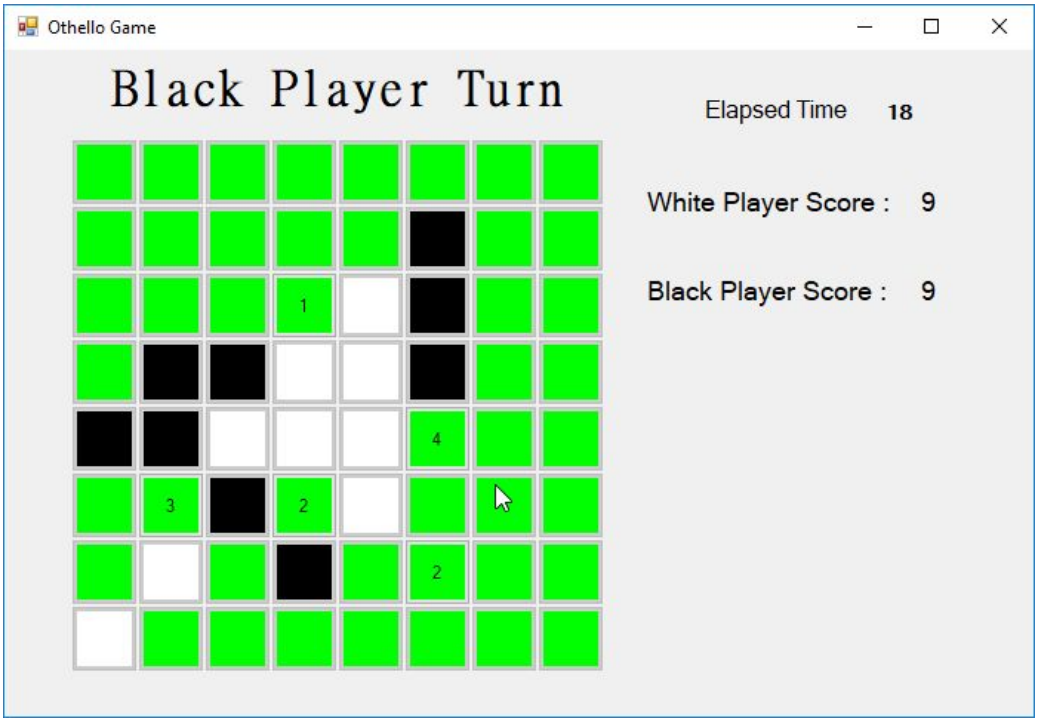
1. Main Menu

   Picture below show application main menu UI.



2. Game Board

   Picture below show application game board UI.



4. Enemy Bots (AI)

This game enable user to play with AI or bots. For now only AI with greedy algorithm available can be choosen as enemy.

Greedy AI iterate all panels in board then choose panel that can flip most enemy disks. Greedy AI read the panel count to knows possible flipped disks. 2 seconds delay added before Greedy AI can move, the purpose is so user can track where the enemy put his disk and give handicap to user to win the game.

5. Man Hour

[Any industry or market-related risks]

Milestone	Tasks	Time Period	Estimate Hours of Work
1 - Design			
1.1	Analysis	01/09/19 - 07/09/19	3
1.2	Architecture design	01/09/19 - 07/09/19	3
1.3	UI Design	01/09/19 - 07/09/19	2
2 - Development			
2.1	Create Classes	08/09/19 - 14/09/19	3
2.2	Create GUI	08/09/19 - 14/09/19	
2.2.1	Main Menu UI		2
2.2.2	Game Board UI		3
2.3	Implement game	08/09/19 - 21/09/19	
2.3.1	Game initialization		3
2.3.2	Game play		8
2.3.3	Game over		3
2.4	Implement AI bot	08/09/19 - 21/09/19	2
3 - Testing			
3.1	Player VS Player game play	15/09/19 - 21/09/19	1

3.2	Player VS Bot game play	15/09/19 - 21/09/19	1
3.3	Bug fixing	08/09/19 - 21/09/19	3
4 - Documentation			
4.1	Write software development documentation	09/09/19 - 21/09/19	3
Total Work Estimation		01/09/19 - 21/09/19	40

6. Possible Additional Features

This game can be more interesting if we add several features like :

- Add more AI algorithm
- Add game history
- Add reply to game history
- Add player maximum time to move (despite allowed or not by game rules)