Software architecture overview

Design Patterns - MT803

MTU Kerry

11/3/2022

Table of Contents

[Background Information 2](#_Toc118384739)

[Existing structure 2](#_Toc118384740)

[Requirements 2](#_Toc118384741)

[Functional 2](#_Toc118384742)

[Non functional 2](#_Toc118384743)

[Use Cases 2](#_Toc118384744)

[Change graphics options 2](#_Toc118384745)

[Load board from file 3](#_Toc118384746)

[Plays a valid move 3](#_Toc118384747)

[Board is saved 4](#_Toc118384748)

[Components 4](#_Toc118384749)

[UML 5](#_Toc118384750)

# Background Information

Chess is an old game enjoyed by many This initial implementation is one executable, that requires all players to be on one computer screen. The aim of this this project will be to alter the prototype into a product that can, more easily, be maintained, expanded to multiple platforms, and be given online play.

# Existing structure

## Requirements

### Functional

* Players must be able to set the size of their board.
* Players must be able to set the colour of both teams.
* Players must be able to load a board state from a file before the game.
* Players must be able to save the current board state to a file during the game.
* Players must be able to move their pieces in valid moves on their turn during the game and have their turn end when a valid move is made.

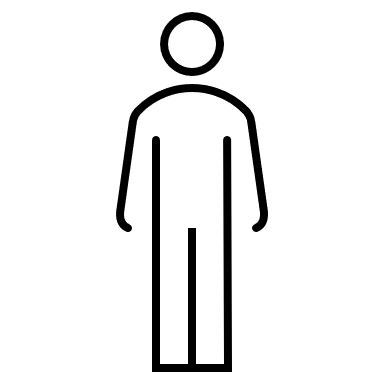
### Non functional

* Players valid moves must be made visible in real time (less than 1/3 seconds).
* Chess game must be portable to other platforms
* Players must not be able to cheat by taking invalid chess moves.

## Use Cases

### Change graphics options

User



Primary actor: User

Brief: The user changes the size of the board and/or the colour of any team.

Postconditions: The size of the board will be the user’s choice when the game is started. The colour of both teams will be what the user chooses when the game starts. The colour previews of the team are what the user choose.

Preconditions: application was started but no game was yet started.

Triggers: The user clicked the board size drop down or teams’ colour.

Basic flow:

Change screen size

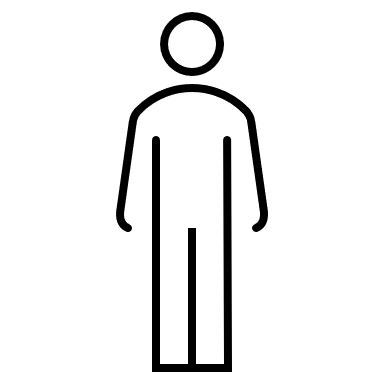
1. The system displays the options of board size: small, medium, or large.
2. The user chooses one of the options.
3. The system stores the selected board size.
4. The system hides the board size options.

Change team colour

1. The user presses a team colour.
2. The system displays a colour selector window.
3. The user selects a colour and confirms.
4. The system stores the selected colour.
5. The system hides the colour selector window.

### Load board from file

User



Primary actor: User

Brief: The user loads a board state from a .brd file.

Postconditions: The state of the board will be as describe by the file when the game is started.

Preconditions: application was started but no game was yet started.

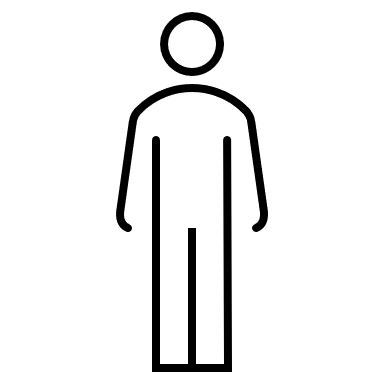
Triggers: The user click the load board button.

Basic flow:

1. The system displays a file select window.
2. The user selects a file.
3. The system hides the file select window.
4. The file is stored to the board state.

### Plays a valid move

User



Primary actor: User

Brief: The user moves a piece on their turn.

Postconditions: The user’s piece will have been moved and it will be the other user’s turn.

Preconditions: The game must have started, and it must be the user’s turn.

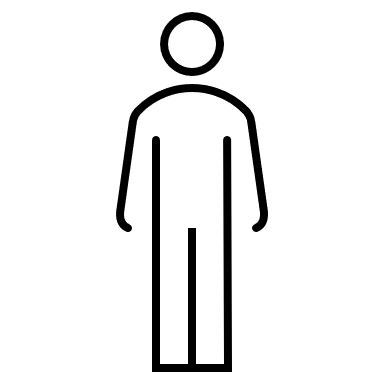
Triggers: The user clicks down on their coloured piece during their turn.

Basic flow:

1. The user drags their piece
2. The system detects the user’s cursor is over a different grid tile from where it started
3. The system validates the move.
4. If the more is valid, the system changes the position of the visual of the selected piece.
5. When the user lets go and the piece is in a valid move, the piece is moved.
   1. If the piece is in a invalid position or if the piece was not moved, the selected piece is not moved and the graphic is reset.

### Board is saved

User



Primary actor: User

Brief: The user saves the board state to a file

Postconditions: The user will have a file on their computer containing the board state.

Preconditions: The game must have started.

Triggers: The user pressed ‘s’ on their keyboard.

Basic flow:

1. The system displays a menu.
2. They user enters a name for the board state and confirms.
3. The board state is saved under a file called the user’s inputted name followed by ‘.brd’.

## Components

## UML

