Matt's Brickus UI Use Cases

A New Game is Created

Primary Actor: Model

Goal: Have a new Brickus Game to play

Main Path

1. Create a new, empty boardGrid in the Model
2. Create two players, and give them each the set of 21 pieces in the tray
3. Assign the first move to the player designated as Player1

Player Selects a Piece

Primary Actor: Player

Secondary Actor: Piece

Pre-condition: It is this Player's turn, and he has not yet played the Piece s/he hopes to select.

Goal: For the active Player to have a selected Piece

Main Path

1. Player decides which piece s/he would like to play
2. Player clicks on this Piece in their tray of pieces
3. The Piece is highlighted in the tray, indicating it has been selected.

Player Rotates a Piece

Primary Actor: Player

Secondary Actor: Piece

Pre-condition: It is this Player's turn, and s/he has selected a Piece (see *Player Selects a Piece)*

Goal: For the active Player's selected Piece to be rotated in the desired direction

Main Path

1. Player chooses their direction of rotation (Down = Counter-clockwise, Up = Clockwise)
2. Player scrolls their mouse wheel one "tick" in this direction
3. The Piece is rotated 90 degrees based on the chosen direction of rotation, which is reflected by the updated appearance in the tray

Player Flips a Piece

Primary Actor: Player

Secondary Actor: Piece

Pre-condition: It is this Player's turn, and s/he has selected a Piece (see *Player Selects a Piece)*

Goal: For the active Player's selected Piece to be flipped in the desired direction

Main Path

1. Player chooses their direction of flip (Horizontal = Right click, Vertical = Shift + Right click)
2. Player executes the key-press based on desired direction of flip
3. The Piece is flipped based on the chosen direction of flip, which is reflected by the Piece's updated appearance in the tray

Player Plays a Piece

Primary Actor: Player

Secondary Actors: Piece, Model

Pre-condition: It is this Player's turn, and s/he has selected a Piece and performed all necessary transformations (see *Player Selects a Piece, Player Flips a Piece, and Player Rotates a Piece*)

Goal: The Player desires to place his selected piece on to the grid

Main Path

1. Player drags the piece onto the playable space (ie, the board).
2. The board reflects this by coloring the tiles which the selected piece will occupy in the player's color.
3. The player moves the piece on the board to their desired location.
4. The player clicks to finalize a move. If the desired location is valid (see *Model Checks for Legal Move)*, the board will reflect this move by marking the locations on the grid which this Piece occupied as owned by this Player.

Model Checks for Legal Move
Primary Actor: Model
Secondary Actors: Player, Piece
Pre-condition: The Player has clicked to make the move (see step 4 of *Player Plays a Piece)*
Goal: Ensure the Player's desired move is a legal move
Main Path

1. The Model ensures the Piece is entirely on the Model's grid.
2. The Model ensures that the Piece is not overlapping any other played pieces (by checking the grid to see if the tiles the Piece hopes to occupy on the grid are not already occupied).
3. The Model ensures that the Piece is touching another piece played by this Player diagonally.
4. The Model ensures that the Piece is not touching another piece played by this Player orthogonally.
5. If all the conditions are met, then the move is a valid move.

Game Calculates Player's Score
Primary Actor: Model
Secondary Actor: Player
Pre-condition: None
Goal: To get the Player's score based on the current Model
Main Path

1. Iterate through all of the tiles on the Model's grid. If the tile contains a Piece played by this Player, this Player's score is incremented.

Player Passes a Turn
Primary Actor: Player
Secondary Actor: Model
Pre-condition: None
Goal: For this Player to skip his/her turn
Main Path

1. If the most recent move was a pass (stored in the Model), end the game (see *Game Over)*.
2. Otherwise, mark the most recent move as a pass, and switch to the other Player's turn.

Game Over
Primary Actor: Model
Pre-condition: Two consecutive moves were passes
Goal: To end the current Game
Main Path

1. Freeze the game state so that no more pieces can be played by either Player.
2. Calculate each player's score to determine a winner (see *Game Calculates Player's Score)*.
3. Congratulate the winning Player.

Player Hits New Game Button
Primary Actor: Model
Pre-condition: Player hit the 'New Game' button
Goal: To end the current Game and start a new one
Main Path

1. End the game (see *Game Over*).
2. Initialize a New Game (see *New Game is Created)*.