Java Project - Tile Memory Game

Dylan Liu, Pratiksha Saravanan, Saanvi Neema
Period 3

Project Description

The project is to create a single-person tile memory game using JavaFX. The player must replicate a sequence of tiles displayed by the game (must click the same tile in the same order). Every time the player gets a sequence right, the game adds one more tile to the sequence. When the player gets the sequence wrong, the player loses a life. When the player loses all their lives, the game ends, and they get a choice to play again or quit.

Detailed Description

1. Set-up:
    - Get the name of the player
    - Ask them to pick the size of their board
        - Minimum is 2 rows and 2 columns, maximum is 7 rows and 7 columns
    - Ask the player how many lives they want to start with (0 or more)
    - Draw the board and set-up the UI based on their input
2. Play Round:
    - The game starts by playing a sequence of one tile: a randomly selected tile flashes a random color and a random musical note
    - The player replicates the shown sequence by clicking the respective tile. If the user clicks the correct tile, the tile flashes the same color and plays the same musical note as was displayed when the sequence was being shown.
    - If the player gets the tile sequence correct, the game repeats the same sequence but adds one new tile to the sequence. The game adds one point to the player's score for each tile correctly clicked.
    - If the player gets a tile wrong, the game plays a sound to indicate the player made a mistake. One life is deducted from the number of lives the player chose to start with. If they have enough lives, they have multiple opportunities to choose the right tile. The game continues until all lives have been lost.
3. Round Ended:
    - After the player loses all their lives, the game displays a message saying: "Game Over". An alert pops up, which says "Incorrect Tile" and then displays the player name followed by their final score.
    - The alert also displays two buttons:
        - If the user clicks "Play Again", the game resets the board and the score and starts a fresh game with a "Let's Go!" sound.
        - If the user clicks "Quit", the game quits with a "Goodbye!" sound.

Example Play

Our game looks a little different from how we initially planned it. We added another slideshow to show how our game works. It does not show the full game since it will be demonstrated on the final. Link: Memory Tile Game - Final Example Play

Final Classes

- FinalProject.java - launches the game
    - Attributes: None
    - Methods:
        - void start(Stage stage) - creates a new gameUI object and passes Stage object to it
        - public static void main(String[] args) - sets up and runs the game

- GameUI - creates set-up screen and game board layout, handles user click events and game logic
    - Attributes:
        - int rows, columns - the amount of rows and columns for the board
        - int initialLives - number of the lives entered by user
        - int lives - number of lives the user has left
        - TileManager tileManager - object of TileManager class
        - SequenceHandler sequenceHandler - object of SequenceHandler class
        - ScoreHelper scoreHelper - object of ScoreHelper class
        - SoundPlayer soundPlayer - object of SoundPlayer class
        - VBox, GridPane, Label - classes for UI elements
    - Methods:
        - void askGridSize(Stage stage) - asks the user for their name, the number of rows and columns, the number of lives, and then creates a board of that size with those inputs
        - void startGame(Stage stage, String playerName) - initializes the main UI with the user's input and starts the first level
        - void startLevel() - resets all tiles, then generates the new sequence
        - void onSequenceFinished() - asks the user to repeat the sequence that was displayed
        - void onTileClicked(Tile tile) - handles the logic when a tile is clicked by the user.
        - void showGameOverDialog() - displays the "Game Over" message when the user clicks an incorrect tile (and has no lives left).

- SequenceHandler class - manages the sequence logic. Generates the tile sequence and assigns colors and musical notes.
    - Attributes:
        - TileManager tileManager - object of TileManager class
        - List<Tile> tileSequence - ordered list of tile objects
        - List <Color> colorSequence - ordered list of color objects corresponding to the tiles
        - List<String> noteSequence - ordered list of note objects corresponding to the tiles
        - int currentIndex - defines the index of the position in the current sequence
        - GameUI gameUI - reference to user interface
    - Methods:
        - void addToSequence() - add a new tile, color, and sound to the current sequence
        - void playSequence() - plays the current tile sequence with tile flashing and sound
        - boolean verifyClick(Tile clickedTile) - checks if the clicked tile matches the expected tile in the sequence
        - boolean isSequenceComplete() - checks whether the user has completed the current sequence
        - void reset() - clears the tile sequence, color sequence and note sequence
        - String getNoteForTile(Tile tile) - gets the musical note associated with a given tile in the sequence
        - Color getRandomColor() - selects and returns a random color from the list

- ScoreHelper class - Manages the score and the player's name
    - Attributes:
        - String name - player's name
        - int score - how many tiles of the sequence they clicked correctly (for each tile correctly clicked, score increases by one)
    - Methods:
        - void increment() - adds one point to the players score
        - void reset() - resets score to zero
        - int getScore() - returns the current score of the player
        - String getPlayerName() - returns the player's name

- Tile - represents an individual clickable tile on the board
    - Attributes:
        - int row, col - position of tile
        - Rectangle rect - object of the Rectangle class

    - GameUI gameUI - object of the GameUI class
  - Methods:
    - Rectangle getNode() - returns a node representing a tile
    - void flash(Color color) - changes the tile's color to a specified color, creating a flashing effect
    - void unflash() - returns the color to the default light gray color
    - void enableClick() - enables mouse-clicks on the tile, notifying the gameUI when it is clicked
    - void disableClick() - disables the mouse-click event on the tile
    - void resetColor() - returns the color to the default color
    - void showError() - sets the tile's color to red, indicating an error
    - boolean equals(Object obj) - checks if the tile is equal to another object

- TileManager - manages the grid of tile objects, handles tile creation and the enabling and disabling of user clicks.
  - Attributes:
    - int rows, cols - position of the tiles
    - GameUI gameUI - object of GameUI class
    - List<Tile> tiles - stores all Tile objects
  - Methods:
    - void createTiles(GridPane gridPane) - creates grid of tiles and places them on GridPane
    - void resetTiles() - resets the color of all tiles to their default state
    - void enableClicks() - enables mouse-clicks on all tiles
    - void disableClicks() - disables mouse-clicks on all tiles
    - List<Tile> getTiles() - returns list of all tiles
    - Tile getRandomTile() - returns a random tile from the list of tiles

- SoundPlayer - handles the playing of all sounds
  - Attributes:
    - AudioClip wrongSound - object indicating the wrong sound
    - AudioClip letsgoSound - object indicating the "Let's Go!" sound
    - AudioClip goodbyeSound - object indicating the "Goodbye!" sound
    - Map<String, AudioClip> noteSounds - map of sound name and its corresponding sound file
  - Methods:
    - void playWrong() - plays the sound indicating a wrong move
    - void playLetsGo() - plays the sound indicating start of a fresh game
    - void playGoodbye() - plays the sound indicating the end of the game
    - void playNote(String note) - plays the sound with given note letter

        - String getRandomNote() - returns a random musical note character between a and g.

Roles and Responsibilities

Documentation: Everyone
Google Drive Folder: Saanvi
Submissions: Saanvi
Classes:

- SequenceHandler: Pratiksha
- ScoreHelper: Pratiksha
- Tile: Dylan
- TileManager: Dylan
- SoundPlayer: Saanvi
- GameUI: Saanvi
- FinalProject.java: Everyone

Project Schedule Plan

SequenceHandler Coded - 5/16
ScoreHelper Coded - 5/16
Tile Coded - 5/21
TileManager Coded - 5/21
SoundPlayer Coded -5/21
GameUI Coded - 5/23
FinalProject.java Coded - 5/21
Testing & Final Code Done - 5/30
Presentation Ready - 6/1

<u>EXECUTION INSTRUCTIONS (PLEASE READ BEFORE RUNNING)</u>
* This code only works with Java version 18 because we couldn't get JavaFX to work with later versions of Java.
* This code uses maven to get JavaFX libraries.
* This code has some set-up issues on VSCode, and works when run using IntelliJ IDE.