CIS 163
Project 3 – 1024/2048 Game

Due Date: At the beginning of the lab on Tuesday 3/14.

Before Starting the Project

- Review array, and Interfaces (Chapters 6, 7)
- Read this entire project description before starting

Learning Objectives

- Write an event-driven application program
- Develop a GUI program that handles mouse and keyboard events
- Use widgets for showing data graphically
- Arrange widgets using layout managers
- Show program feedback using simple dialog

Introduction

Upon completion of game engine implementation in project 2, you will now write a GUI for playing the game. You should be able to complete the GUI in project 3, **without any modifications** to the game engine completed in Project 2

Program Specifications

Your GUI should include the following minimal set of features:

1. Show the game board and numbered tiles in a grid. Tiles with zero value should be rendered a blank tile
2. Allow the player to use **four different keys (KeyListener)** to move the tiles (-5 pts if you use JButtons)
3. A single GUI run shall allow the user to play several sessions of the game.
4. Show the statistics of the current game and overall games: number of slides made so far, the highest score achieve in past games, number of game sessions played so far
5. Provide JButtons items for the following actions
   - Exit the program
   - Reset the current game (and reset the game board)
   - Resize the board to any reasonable size (rather than the default 4x4). Your GUI should work correctly for rectangular (non-square) board sizes (Without exiting the program!)
6. Provide a mechanism to undo the most recent step. You may implement this feature using a key, menu item, or button
7. Provide sufficient feedback (like a dialog) on the following occasions (using a
   - When the player won/lost the game

When the game engine throws an exception (for instance attempting to undo beyond the initial configuration)

Programming Resources

Study the structure of the text mode UI (TUI) code and learn how the program interacts with the game engine. Your GUI program shall be designed using the same pattern. The main exception to the GUI code: it replaces the main loop in the TUI that reads the user input from the keyboard with an event listener.

Recommended Steps

The following steps assume you write the GUI in Swing.

Step 1: Setup the GUI Program
Under the same project created in Project 2, create a new package for the GUI.

Step 2: Render the Game Board

1. Apply BorderLayout to your top-level JFrame object. Use the CENTER compartment for rendering the game board. The other four compartments (NORTH, SOUTH, WEST, EAST) will be used later for additional widgets
2. Setup a container (panel) that uses GridLayout and render widgets of your choice (JLabel). Place this container into the CENTER compartment referred in the previous step.
3. Use a 2D array to hold these JLabel objects
4. Define a private method for updating the text of each JLabel using random values.

Step 3: Keyboard Event Handling

1. Define a KeyListener and attach it to the container created in Step 2.
2. In the keyPressed() method add necessary code to handle four keys of your choice. In response to pressing these keys, invoke the private method created in Step 2

Step 4: Connect The Game Engine
Using a similar approach as demonstrated in the TUI code, connect your GUI code with your implementation of the game engine. Use appropriate try-catch for handling any exceptions and display a dialog.

Step 5: Add Menu Items
Add necessary menu items for handling other user actions (game reset, game board resize, ….)

Step 6: Show Game Statistics
Add another container (panel) for holding several labels for displaying the game statistics. Insert the container into one of the four remaining compartments referred in Step 2 above.

Update the text of these labels in appropriate places of your code.

Extra Credit Options (Extra Credit is only given when ALL other requirements are completed.)

Several possibilities for extra credits:

- Apply time limit to hit the winning value. Use Swing Timer to update the remaining time every second (or your choice of "refresh rate")
- Develop a customized widget for rendering each tile
- Use Swing Timer to add fading in/fading out effect
- Render the transition of the game board from its previous state to the current state

Turn In

- Be sure to add proper **Javadoc comments** throughout your source code
- At the top of the source file, write a comment block that includes the @author and @version tags. Include your name **after** the @author tag.
- At the beginning of each method (public and private), write a comment block that describes what the method does. For each parameter, use the @param tag to explain how the parameter is used by the method. When the method returns a result, explain it using the @return tag.
- Keep each line (statements and comments) not to exceed the right margin at column 75. Break a long statement into several lines when it exceeds this margin.
- Be sure **your source code is properly indented**. Modern IDEs (like Eclipse and IntelliJ) provides a menu command shortcut for "Reformat Code". Learn how to use this editor command and *avoid indenting your source code manually* line-by-line.

Project 3: “1024 Game" Program Rubric.

Student Name	
Due Date	
Date Submitted, Days Late, Late Penalty	

Graded Item	Points	Comments and Points Secured
Javadoc Comments and Coding Style/Technique (http://www.cis.gvsu.edu/studentsupport/javaguide) 1. Code Indentation (auto format source code in IDE) 2. Naming Conventions (see Java style guide) 3. Proper access modifiers for fields and methods 4. Use of helper (private) methods 5. Using good variable names 6. Header/class comments 7. Every method uses @param and @return (1 sentence) 8. Every method uses a /***************** separator 9. Overall layout, readability, No text wrap 10. Using /** … / for each Instance variable 11. Has many inner “inner" comments	10	
Appearance (does it look nice) Undo button. Number of wins, Stats Error checking (e.g., user input) Board size changing with out restarting All other stuff (GUI) KeyListener Misc (Code design, etc)	30 10 10 10 10 10 10 10	
Total	**100**	

Additional Comments: