DUE DATES:

Monday recitations: 9:00 PM on 11/20
Wednesday recitations: 9:00 PM on 11/22
Thursday recitations: 9:00 PM on 11/23
Friday recitations: 9:00 PM on 11/24
Saturday recitations: 9:00 PM on 11/25

Preliminaries

This lab is the second of three labs (lab 8, lab 10 and lab 11) that in the end will have you build a single-player matching game like Candy Crush.

For this lab you have an option: continue with the code you wrote for lab 8, or use our lab 8 solution code. Note: even if you did not complete lab 8 you can attempt lab 10.

Keep in mind that any work you submit must be your own. Submitting work done by someone else as your own is academically dishonest, and will result in immediate failure in the course. We will use software tools to detect inappropriate collaboration.

Your TAs will give you an overview of the lab and hints of how to proceed in recitation. You should expect to have to put in time outside of recitation to finish this lab. We therefore recommend that you bring your laptop to recitation so you can more easily continue to work outside of Baldy 21.

During office hours TAs will give priority to students who have attended recitation. It is not acceptable to skip recitation and then expect one-on-one assistance in office hours. Some aspects of this lab will ONLY be discussed in recitation, and TAs will NOT answer basic questions about these topics during office hours.

Because this lab gives you a great deal of freedom in how you write the code there is NO AUTOMATED GRADING. When you submit to Web-CAT the only score you will see is your early bonus or late penalty. The functionality of this lab will be manually graded by the TAs.

Overview

For this lab your program will display a grid of colored tiles. Adjacent tiles can be interchanged. Non-adjacent tiles cannot be interchanged. A *match* is defined as three adjacent tiles of the same color (either horizontally or vertically). If there is a match on the board after two tiles are interchanged the program must print out the message “The board has a match”, otherwise it must print out “The board has no match”.

Remaining gameplay functionality, including some extra credit, will be added to lab 11.

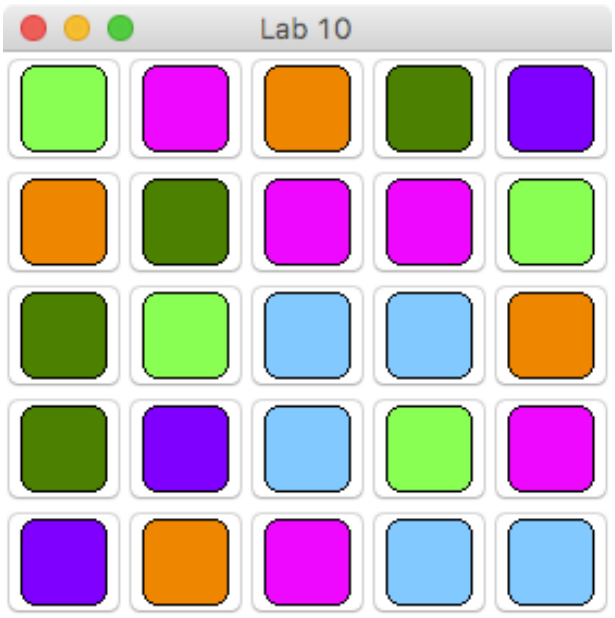
Requirements

[20 points] TILE BOARD DISPLAY

Display a JFrame whose title is your first and last name, followed by "'s Lab 10". For example, the title of my JFrame must be "Carl Alphonce's Lab 10". (The screenshots provided do NOT show the correct title.)

Display the board of tiles as a 5 by 5 grid of JButtons, each displaying one of the supplied images, selected at random. Keep in mind that your code MUST maintain the user interface (view+controller) and model separation. The view displays the board of tiles, based on the data maintained by the model. The controller accepts input from the user, and communicates it to the model. There must be no "business logic" embedded within the user interface. Likewise, there must be no view or controller functions in the model (i.e. nothing visual/graphical: no images, no JLabels, no JButtons, etc).

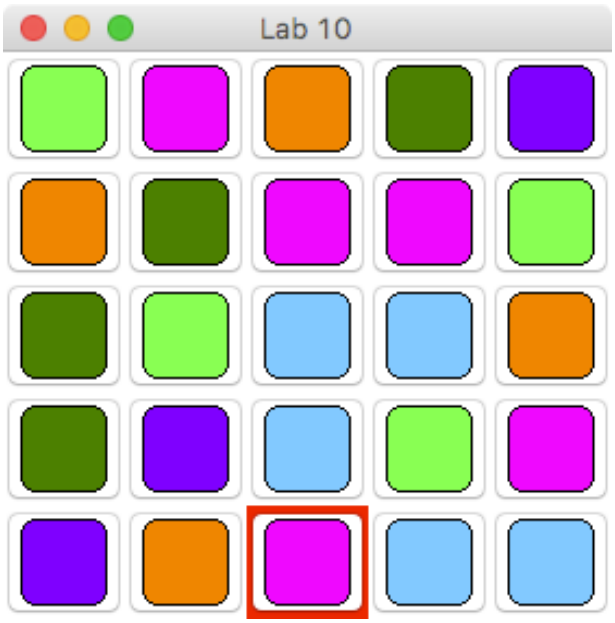
Your program might display something similar to this:



(continued on the next page)

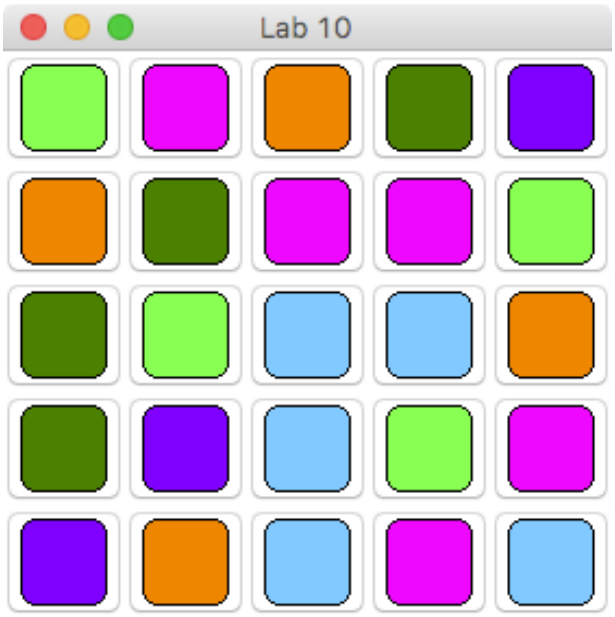
[20 points] TILE SELECTION

Clicking on a JButton selects that tile. A selected tile must have a red border displayed.



[20 points] TILE SWAP

Clicking a JButton adjacent to the selected one interchanges the tiles. Note that the interchange happens in the model. The user interface simply updates to show the new arrangement of tiles. This screenshot shows what the board should look like if the tile to right of the selected one is clicked:



(Clicking on a JButton that is not adjacent to the selected one deselects all tiles.)

[20 points] MATCH DETERMINATION

A *match* is defined as three adjacent tiles of the same color (either horizontally or vertically). If there is a match on the board after two tiles are interchanged the program must print out the message

"The board has a match."

and

"The board has no match."

otherwise.

[20 points] DESIGN

The code must exhibit good design, including implementation of model-view-controller. Each class must have a well-defined role to play in the overall system. Naming conventions must be followed. Keep data (instance variables) private. Use local variables unless instance variables are justified. Avoid declaring static variables and methods. You should be able to run multiple instances of your lab 10 code at the same time (i.e. launch it several times and run each one independently of the others). Minimize code duplication. Indent your code properly. Do NOT use arrays in your code – use an appropriate kind of Collection class instead. In general, apply the principles of good software design that we've talked about throughout the semester.

Submitting your project to Web-CAT

Make sure you submit your work on time; due dates are listed at the beginning of this lab description. There is NO automated grading for this lab. You may submit as many times as you wish. Your last submission is the one that counts (so consider carefully whether you want to make any late submissions, as the late penalty is 20 points per day or portion thereof late).
