Overview

We designed a Game Launch Center that lets users sigh up/sign in, start new Minesweeper, Memory, and Sliding Tiles games with different complexities, and view their Scores. We have a scoreboard that displays scores of a user's previous games. An autosave feature that saves the game every few moves that is implemented in Sliding tiles and Minesweeper. An undo feature that allows the user to undo previous moves that is implemented in Sliding tiles and Memory.

- What is your unit test coverage?

Our Unit-Tests cover the three games. Each game has tests that check each method within each of the games Board, BoardManager, and Tile classes. In addition, There are unit tests to cover the other non-activity classes such as step saver, Score, and Scoreboard.

- What are the most important classes in your program?

The most important class in our program is BoardManager. BoardManager is used to manage each of our three games, it handles the saves, and it stores the scores.

- What design patterns did you use? What problems do each of them solve?

We used the iterator design pattern in the BoardManager parent class so that we can iterate through the tiles easily in the same way for all games. There is also a Score iterator which allows us to iterate through the scores.

We used the observer design pattern in Game Activity to notify observers when things needed to be changed. It's also used in MinesweeperBoard to notify observers that the board has been generated, and whenever a tile has been tapped.

We followed the Model-View-Controller design pattern. We kept the view (Activity) classes separate from the model (logic) classes so it would be more organized, and it would be easier to spot bugs when they popped up.

We followed the Low-Coupling, High-Cohesion design pattern. We wanted to keep coupling low so that edits to one class wouldn't have large effects on other classes.

- How did you design your scoreboard? Where are high scores stored? How do they get displayed?

Individual scores are stored with a user's name, game, and points. Scores are then stored in an Arraylist. It gets updated by each of the GameActivity classes. To get the Users high score, sort the array and remove the first score for that specific user. To get the high scores of a specific game, we sort the score list and add the top to a new list to be removed. They are displayed in an activity class. We iterate through and split over each score in the scoreboard so that we can display them separately.