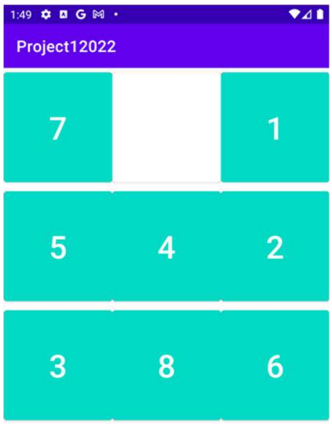
CS 477 Programming Assignment #1: A simple game

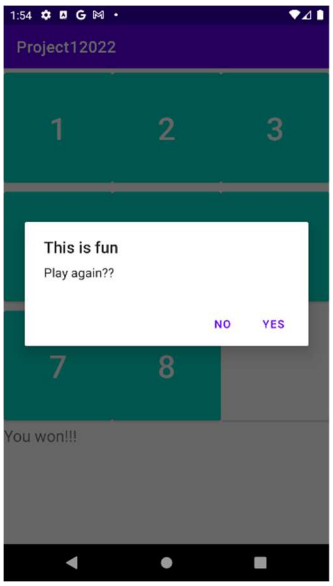
Due: on Blackboard



Touch a tile next to the empty space and that tile will move there. Once you have the tiles ordered 1-8 and the empty tile in the corner, you win!

The purpose of this assignment is to have you use the techniques we have talked about in class and used in lab to create a larger app. This document contains information about the functionality that your app needs to include. I will also demo my app in class. Your version can have a different look and feel but needs to serve the same purpose with similar features. Extra functionality is fine (and even encouraged, particularly for those of you who have played with app development before this class!) as long as it doesn't make the app more difficult to use to use and evaluate.

For this project, you are going to create a version of the simple 3x3 (and 4x4) game where you have to move the scrambled tiles to get them back in order. For example, to the right is a scrambled 3 x 3 grid of values 1-8. I generate the scrambled puzzle using a random number generator to move the pieces. You will want to do something like this to make sure the puzzle is solvable.



To play, the user touches a block next to the empty square. That action moves the block into the empty position (leaving an empty position). Once you get the blocks 1-8 with the empty block in lower right corner, you need to recognize that the puzzle is done and use an alert to see if the user wants to play again or quit, as you can see in the app snapshot to the right. The Tic-Tac-Toe app on blackboard has an example of this.

You are going to implement both the 3x3 and the 4x4 game. There is a intro screen (next page) that lets the user pick which to play. There is an example of a scrambled 4x4 screen also shown on the next page.

It is ok if your app looks a little different than mine but you really want to do everything you can to make sure the puzzles look ok on the screen. I used linear layouts for the puzzles so that I didn't end up with something that was unplayable. The linearlayout example on blackboard might give you some good ideas how to space things evenly across the screen. In addition, getting the button colors was a little bit of a pain – you will want to modify the 'tint' not the 'background' of your buttons. For example, in XML you would use something like

android:backgroundTint="#ff0000".

To do this in Java code, I used this:

```
setBackgroundTintList(ColorStateList.valueOf(getResources().getColor(R
.color.white)));
```

but there is probably a better way.

Notes:

- Assume the grading phone will be in portrait orientation.
- To generate random numbers, use java.util.random – this lets you generate a random number between 0 and N-1.

Grading:

- Achieving the basic functionality and requirements – correct behavior, correct overall flow, no runtime errors: 75 pts
- Overall Experience – making sure interface looks good, works well and makes sense: 25 pts
- Be sure to test your app. There is a mandatory 10 point deduction for apps that we can make crash.

Submitting:

1. Name your .apk file as project1-youruserid. Remember to use 'Build APK'
2. Submit a zip file that contains your Java and XML documents (just the ones you wrote or modified) and separately your .apk file via blackboard. **Do not** submit your entire Android Studio project – this is a really large file!