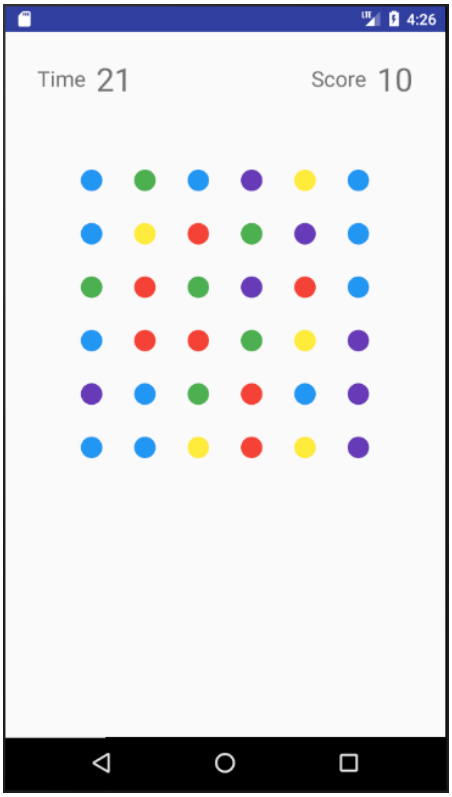
Project 1: Dots
Android App Development
100 points

In teams of twos, you are to create a Dots app similar to the Dots app on Google Play:

https://play.google.com/store/apps/details?id=com.nerdyoctopus.gamedots&hl=en

Rules of the Game

1. The default game grid size is 6 x 6.
2. Each dot is initially assigned one of 5 different colors randomly.
3. The player attempts to connect two or more adjacent dots of the same color (diagonals don't count). After selecting the dots, the selected dots disappear and are replaced with the dots immediately above. Dots that take the old dots' place are randomly colored.
4. If a closed path is formed, all dots of the same color (even those not in the path) disappear immediately.
5. One point is awarded for each dot that disappears.



Requirements

Implementing all the requirements below is necessary to receive 100 points:

- Your app should have 3 activities:
  1. MainActivity – Lets the user select which type of game to be played: timed or moves. Also displays the highest score for a timed or moves game and has a button to change the settings. Finally, the activity should display who created the app.
  2. GameActivity – Displays the game board, time or moves, and the score.
  3. SettingsActivity – Allows the user to change some settings

- Your app should support two types of games:
  1. Timed – A timed game lasts for 30 seconds.
  2. Moves – A move game allows the user to make 15 moves total.

- The number of seconds remaining or moves remaining should be displayed above the game board. MainActivity displays the highest score for both game types. When the app is restarted, the high scores should remain.

- To connect the dots, the user should touch the first dot and hold their finger down over the same colored adjacent dots. A visual indicator should make it clear to the user which dots are selected. When the user removes their finger, the dot path is accepted, and the dots in the path are removed.

- Sound should be played when connecting the dots and at any other time you think would be helpful.

- The user should be able to change at least 2 different settings in the SettingsActivity. Some ideas: color of background, size of game board, sound on/off. When the app is restarted, the same settings should be set.

- The app should allow device rotations and show an ideal screen for either portrait or landscape orientation. The game state should be retained across device configurations.

- The app should have a custom icon.

- The app should be constructed using MVC design. The model should not have Android-specific code; it should manage the dots, dot paths, score, etc. The view should have a game grid composed of ImageView widgets; no other option is allowed.

Bonus: 5 additional points for implementing animations of some sort. Example: You could animate dots appearing or disappearing. Or you could animate the dots falling into the place of the missing dots.

Teams

Everyone has been assigned to a two-person team:

1. Heather Anderson and Samuel Cobb
2. Nathan Burner and Bryan Cuneo
3. John David Griffin, Jhoel Zuniga, and Noah Kinslow
4. Caleb Krise and Benoit Lacoss
5. Bradely Marques and Jared Nesbit
6. Zack McKee and Jerred Shepherd
7. Caleb Spann and Dawna Stirrup
8. Nicholas Terrill and Joshua Werzt
9. Trevor Hale, Dylan Watson, and CJ Wilson

Both teammates should aim to contribute equally to the project. Teams may use pair programming if desired. Ideally, the project should be implemented in parallel with each teammate responsible for a part of the application.

Each team should create a private repo on github.com and add me (fmccown) as a contributor. Both teammates should consistently push their work to the repository so a history of the project's creation is accurately captured on GitHub.

The repo's README file should contain the following information:

1. Link to the repo on GitHub
2. Brief summary of what the program does
3. List of any known bugs that still remain (hopefully none)
4. List of any extra credit that was implemented
5. List what each teammate contributed (e.g., code to transform images, populate the list view, etc.)
6. The percentage of work performed by each teammate

Ideally the work performed by each teammate will be 50/50, but if one teammate does more than the other, the percentages should be adjusted accordingly. The percentages will influence each teammate's final grade on the project. Note that the repo history on GitHub should be a fairly accurate indicator of how much work each teammate did. Pair programming is one technique that helps to equalize the amount of effort each teammate puts in, but it does not allow for parallel implementation.

Grading

These are the things I will be looking for when evaluating your programs:

1. Does your program implement all the requirements and work properly?
2. Is your program well written? Did you use proper Java naming conventions for variables, methods, and classes? Did you use constants and methods when appropriate?
3. Did you clearly cite any code that you may have found online?
4. Did you use git adequately when developing your solution? There should be at **least 20 commits** in your repo, and each person must have committed at least twice. Your .gitignore should keep out any unnecessary files.

Your source code on GitHub should not be modified after you have submitted the README to Easel. I will clone your repository on my machine to run and grade it. One teammate should submit the team's README file to Easel before the deadline along with the app's APK file.

This is a challenging project to implement. Starting early and getting help when you need it will benefit you greatly! Like any assignment, you are not to share your code with anyone but your teammate! That's why you are using a private repo.

Implementation

The model and demo APK file is example from this repo: https://github.com/fmccown/AndroidDots

The repo also contains Test.java, showing how the model class DotsGame can be used. Note that the model has no Android-specific code. You may modify the model in your project, especially to incorporate saving stats during device rotations.

The game board should be composed of ImageViews in a GridLayout. Spend time examining the tic-tac-toe app that uses ImageViews to see how to create and use ImageViews. Create 5 different images (PNG or XML drawables) to use the ImageViews. Use a View.OnTouchListener to determine which ImageViews (dots) the user is selecting. Display the selected ImageViews with a color filter to make the dots brighter or darker, or swap out the images in the ImageViews with something else that makes the dots appear selected.

Use a SoundPool to implement the sound effects. You may use the sound files in my APK file if you'd like or use your own sound files.