Attention! THIS APP IS ALREADY DEVELOPED AND AVAILABLE IN THE APP STORE!

The easiest way to learn about the project/app is to download it from the app store and try it out.
https://play.google.com/store/apps/details?id=com.playposse.thomas.lindenmayer

You can look at the code in GitHub:
https://github.com/thomasfischersm/LSystemAndroid

Note: To prepare for the Udacity submission, I've made some big changes to the app. They have not been released. Therefore, some of the screenshots below will look a bit differently than what you see in the published app.

Attention! THIS APP IS ALREADY DEVELOPED AND AVAILABLE IN THE APP STORE!

Description

Intended User

Features

User Interface Mocks
- Onboarding Step 1
- Onboarding Step 2
- Onboarding Step 3
- Navigation Drawer
- Turtle Trainer
- Sample L-System Rule Sets
- Activity To Enter L-System Rule Sets
- Activity That Renders The L-System
- Help activity

Key Considerations
- How will your app handle data persistence?
- Describe any edge or corner cases in the UX.
- Describe any libraries you'll be are using and share your reasoning for including them.
- Describe how you will have implement Google Play Services or other external services.

Next Steps: Required Tasks
- Task 1: Project Setup
- Task 2: Implement UI for Each Activity and Fragment
- Task 3: Your Next Task

GitHub Username: thomas28

Your App Name Here

Description

The app renders Lindenmayer Systems, a tiny sub specialty in fractal mathematics. The app includes a on-boarding flow and YouTube video to explain what they are. There is also a “turtle trainer” in the app that helps people learn Lindenmayer Systems. (Again this app is in the app store. You can install it to get the user education: https://github.com/thomasfischersm/LSystemAndroid)

You can also look at the Wikipedia entry:
https://en.wikipedia.org/wiki/L-system

In regards to store descriptions, I have tried many variations. The following store description has the best performance for retained installs (retained installs = installs that aren’t immediately followed by an uninstall).

Short description:
Play with the most accessible types of fractals during your lunch break.

Long description:
Lindenmayer Systems are the most accessible fractals to the layperson. Their underlying logic is as easy as Sudoku. This app takes a cool visual from the complicated math world and makes it fun to play around with. While other L-System Apps require complicated screens, this one lets you type and instantly preview the output. Create your own fractal, stamp your name on it, and share it with your friends. There are still plenty of L-Systems waiting to be discovered. You are standing at the edge of a frontier!

Intended User

There are three user groups:

- **L-System enthusiasts (primary user)**: Like myself, there are people who are enthusiastic about L-Systems. They tend to have an interest in mathematics. Someone who is into L-Systems usually requires software to render them. Among the Android

apps in the app store, mine seems to be the best based on rendering speed, feature scope, quality of technology, and most importantly user ratings. L-System researchers will likely develop their own software, yet a hobbyist doesn't have a year to spend on developing his/her own software to play with them. They benefit from my app.
- **Visual artists (secondary user)**: Aside from the math interest, the rendered visuals are very pretty and can be used by visual artists in their art work.
- **Entertainment user (secondary user)**: L-Systems are also fun for non-mathematicians to play around. Say someone at a bus stop might have fun browsing the samples and toying with them.

There are currently 178 installs. This may appear like a low number in comparison to Instagram. However, L-Systems are a real niche. A lot of people seem to download the app, spend a few minutes playing with it and move on. L-Systems for most people is more like a movie that's watched once rather than a toothbrush that's used every day. According to Google Analytics, the historic average session length moves between 1.5-25 minutes.

Features

Main feature:
- Enter rules for an L-System and have it rendered.

Secondary features about L-System rendering:
- Supports the most extensive and feature rich L-System grammar. (I assume the reviewer is not familiar with the details and doesn't want to be bored with it.)
- Fastest L-System rendering. (I don't have extensive benchmarks. This is based on ad hoc reviews. I don't think any of the competitive apps were created by a real software engineer let alone had the algorithm profiled.)
- Live preview: As the user enters rules, there is a little live preview. (No competitor has this.)
- Stochastic L-Systems: L-Systems were invented by Lindenmayer to model plants. Part of plants is variability. Plants aren't exactly symmetrically. The app has implemented the necessary grammar for stochastic L-Systems. A down swipe on the rendering activity will re-render the L-System with a different random seed. E.g., if a user defined the grammar for a tree, each down swipe will render the tree looking slightly differently.

Secondary app features:
- Turtle trainer: Turtle graphics is an integral concept to L-Systems. Novices can use the turtle trainer to learn about turtle graphics. Advanced users can use the turtle trainer to sketch turtle instructions. (See Wikipedia for an explanation what turtle graphics are: https://en.wikipedia.org/wiki/Turtle_graphics)
- On-boarding flow: Explains how L-Systems work briefly.
- Share function: Allows users to share their rendered L-Systems.

- Samples: The app comes preloaded with L-System sample rule sets.
- Saving rule sets: Users can save rule sets that they created.
- Survey tickler: Asks the user to fill out a survey about the app.

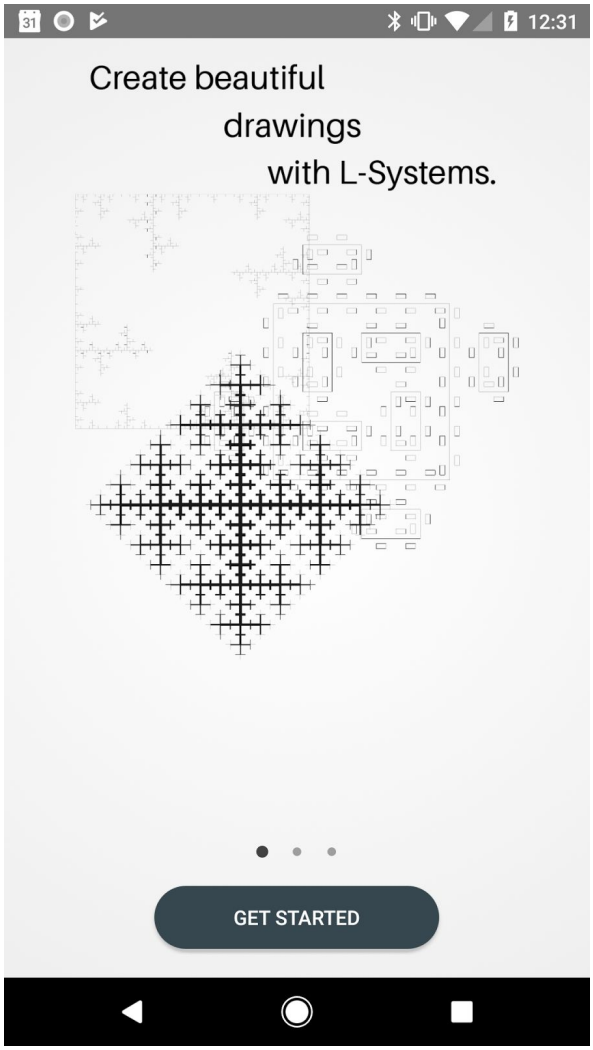
Future enhancement:

- I'd like to add the ability for users to publish their rule sets to a public library that every user can see. That feature will include a sub feature to allow users to rate rule sets.
- Export to SVG.
- Generate Android XML path Drawables. Use animations to transition between these drawables.

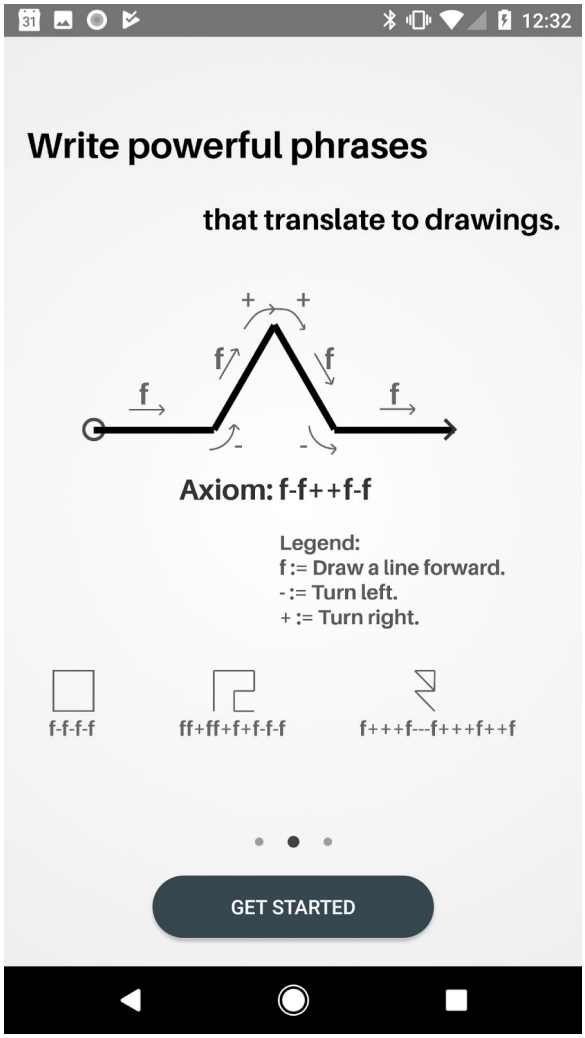
User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

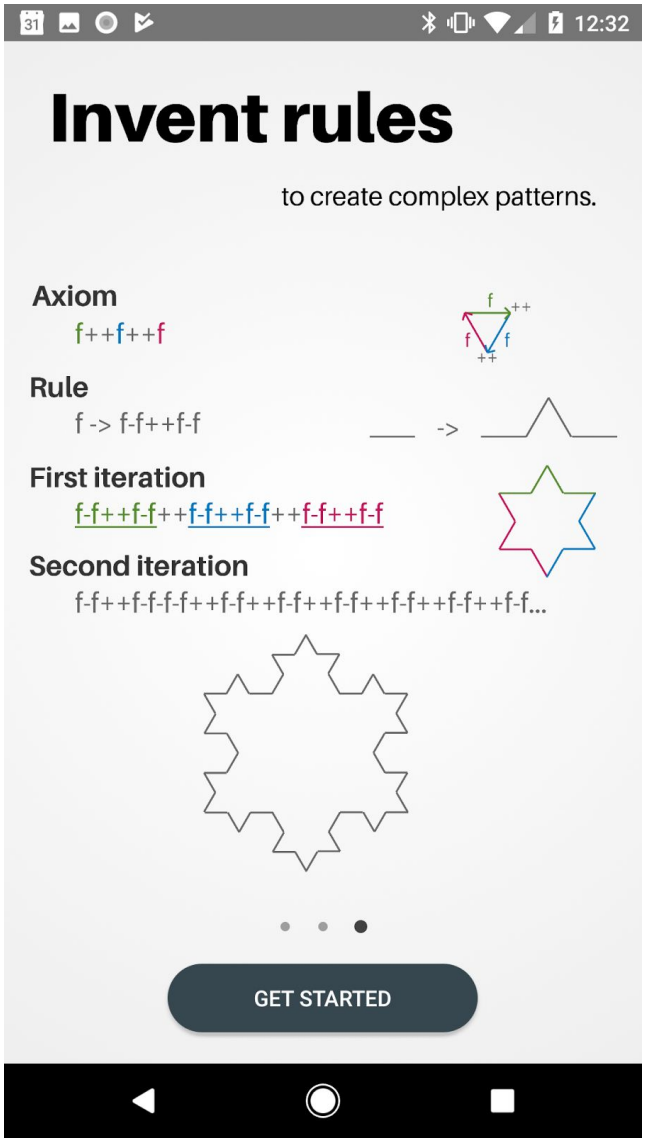
Onboarding Step 1



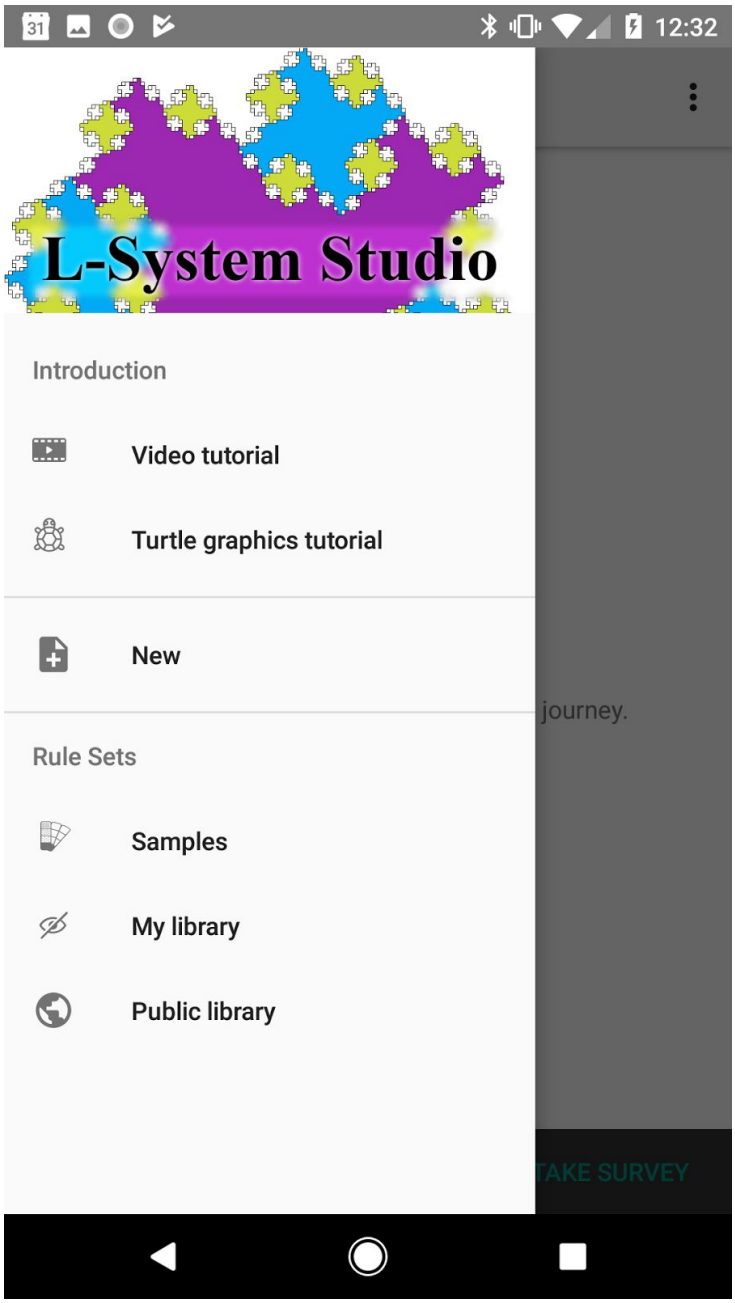
Onboarding Step 2


12:32
Write powerful phrases
that translate to drawings.
f
f
f
f
+
+
-
-
Axiom: f-f++f-f
Legend:
f := Draw a line forward.
- := Turn left.
+ := Turn right.
f-f-f-f
ff+ff+f+f-f-f
f+++f---f+++f++f
GET STARTED

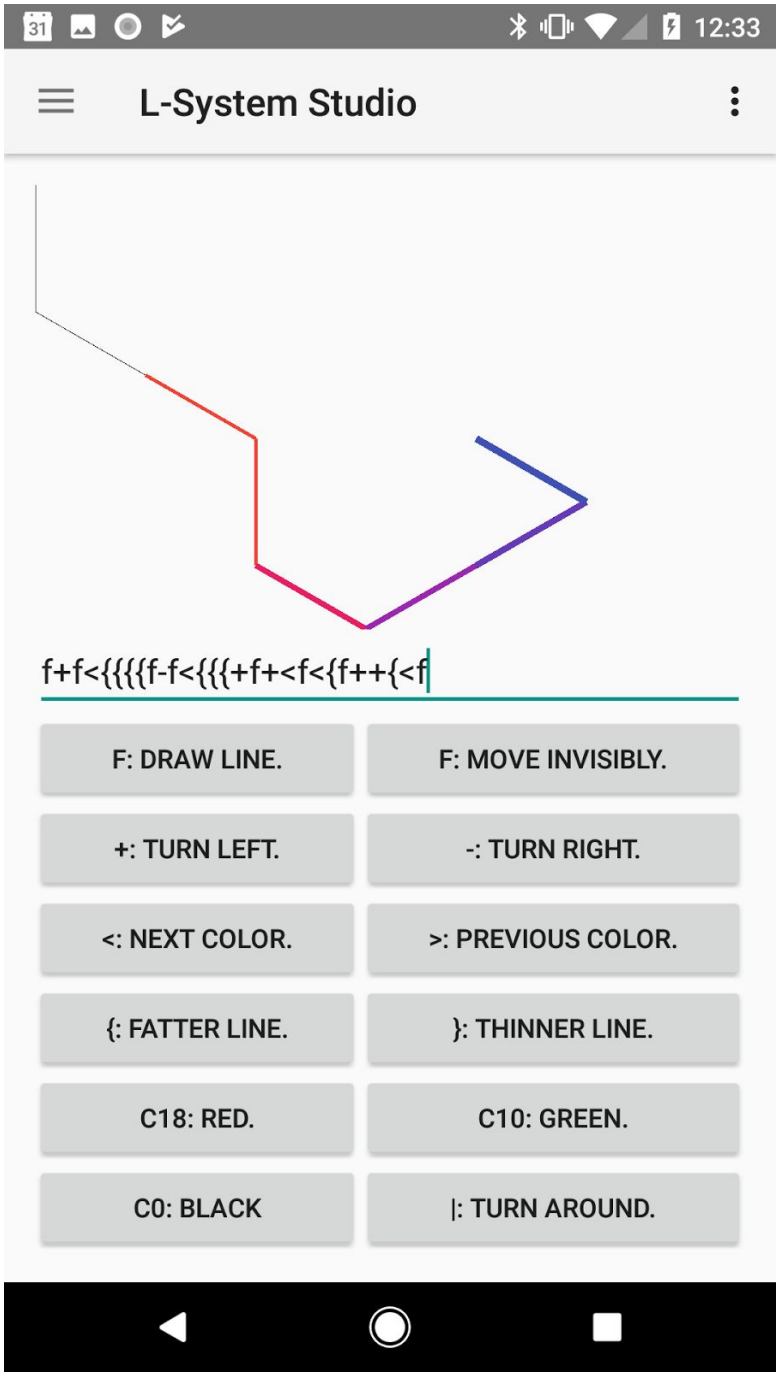
Onboarding Step 3



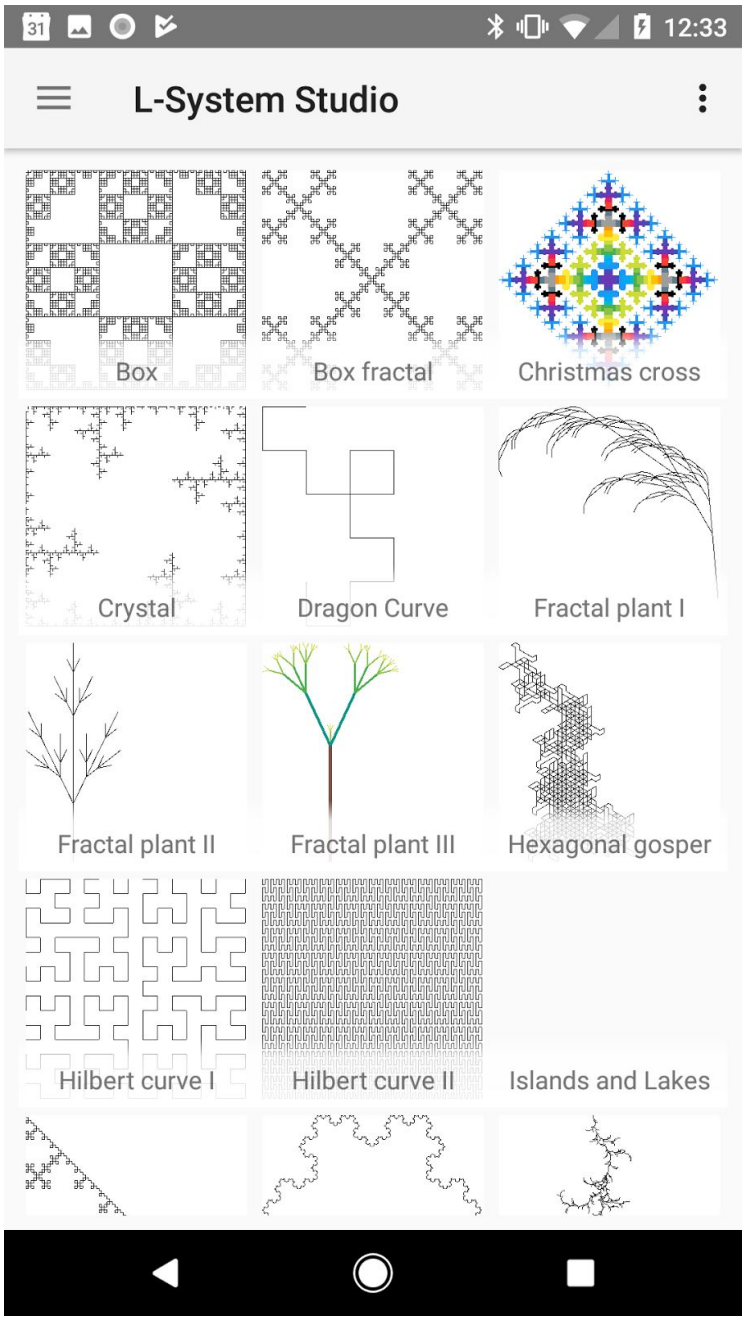
Navigation Drawer



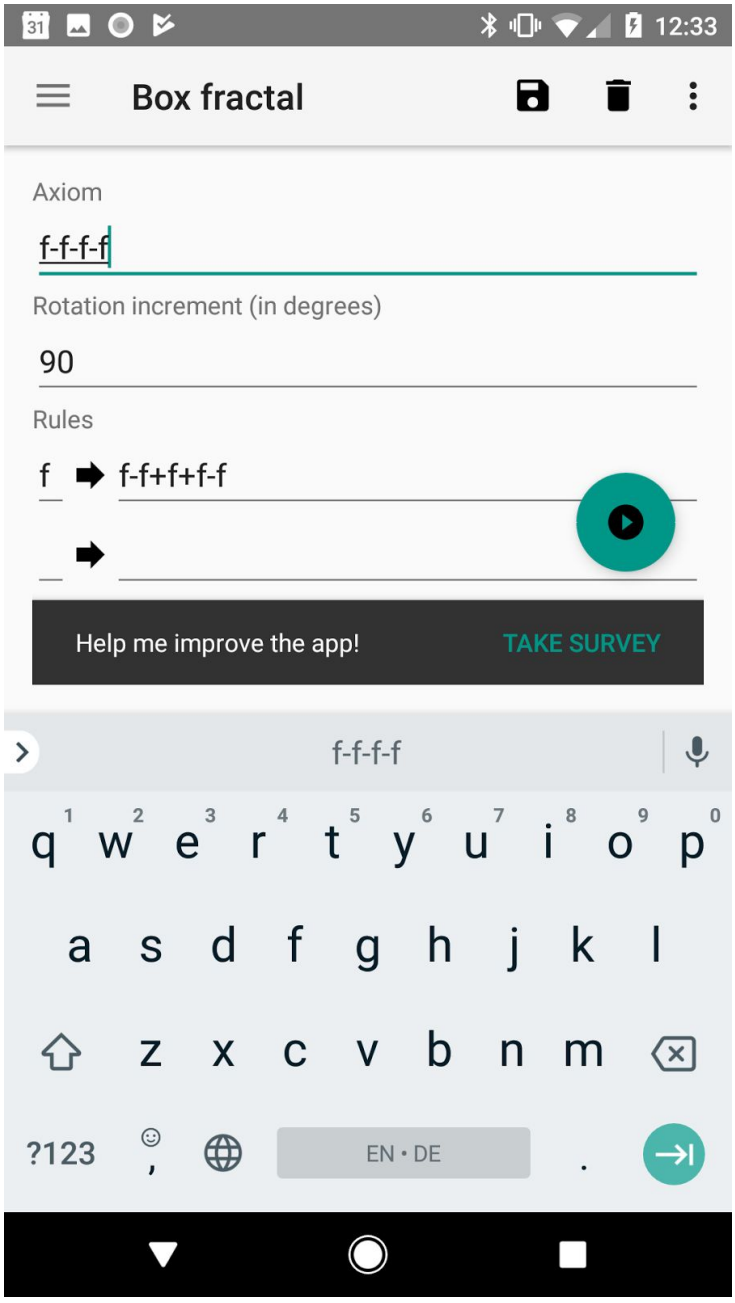
Turtle Trainer



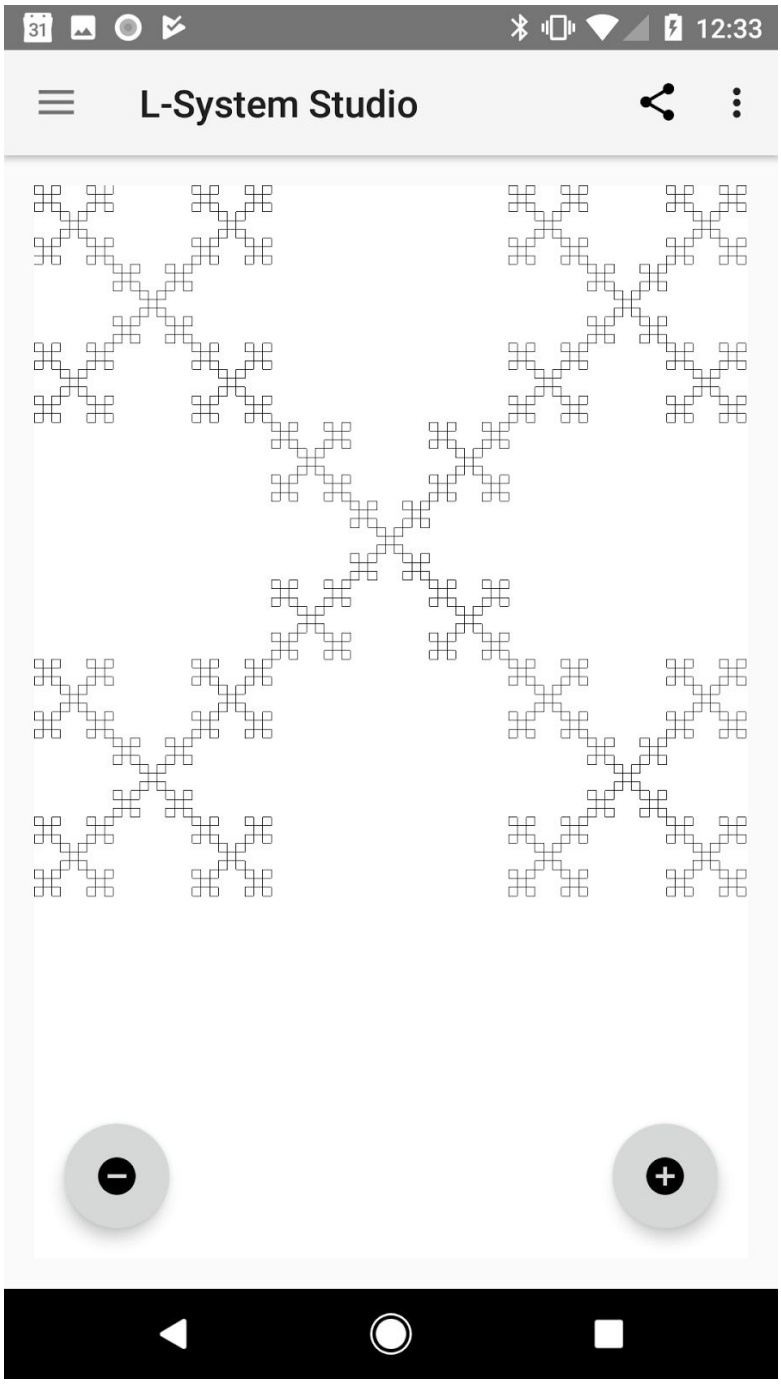
Sample L-System Rule Sets



Activity To Enter L-System Rule Sets

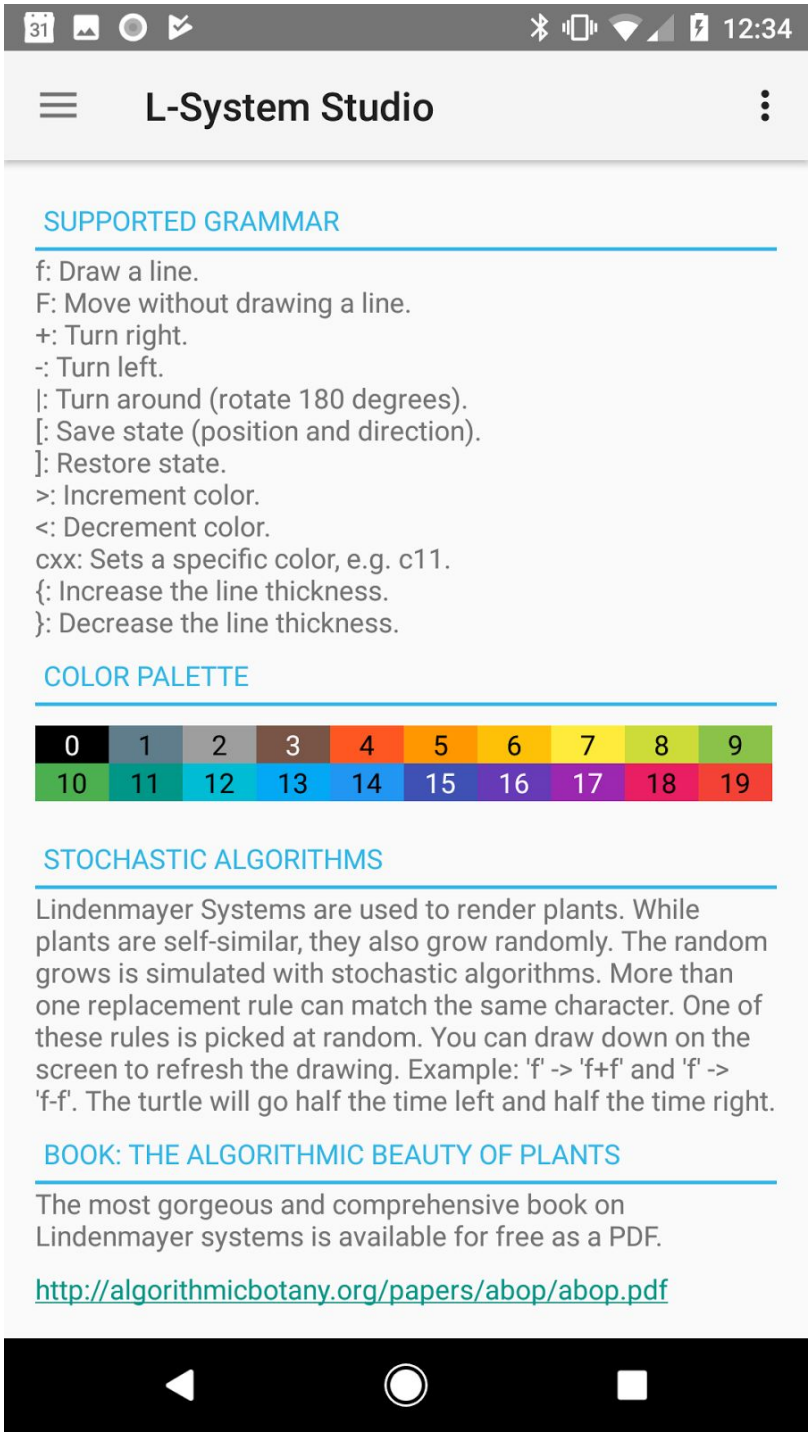


Activity That Renders The L-System



Add as many screens as you need to portray your app's UI flow.

Help activity



Key Considerations

How will your app handle data persistence?

The app used to persist the data as JSON files. JSON files are faster and more lightweight than the SQLLite database. For the Udacity submission, I converted the data persistence to a ContentProvider/SQLLite combination. The app still has some support for the JSON files to: (1) convert user data from old app versions to SQLLite and (2) provide an easy mechanism for adding new sample rule sets to the app in a GitHub version controlled way.

For the future, I want to allow users to publish their rule sets for all other users to see. To implement that I plan on using Firestore, Firebase authentication, and Firebase cloud functions.

Describe any edge or corner cases in the UX.

There are many edge cases in the app. Listing them all here would take a lot of time and be tedious to read for the reviewer as well. As the app is already implemented, please refer to the source code: https://github.com/thomasfischersm/LSystemAndroid

Describe any libraries you~~'ll be~~ are using and share your reasoning for including them.

- **Glide** to render and cache L-Systems asynchronously. A custom resource type, DataFetcher, key, and so on is implemented to connect Glide to my rendering engine (instead of downloading an image from the Internet).
- **Butterknife** for UI binding.
- **Guava** for useful Java methods.
- **GSON** for encoding/decoding Lindenmayer rule sets to/from json.

Describe how you ~~will~~ have implement Google Play Services or other external services.

- **Google Analytics** to analyze user behavior.
- **Fabric** to analyze user behavior.
- **Crashlytics** to capture crash reports near real-time.

To support users to publish their rule sets, I plan on using:

- **Firebase authentication** as a prerequisite for Firestore (A user's published rule set has to be protected from another user being modified.)
- **Firestore** to persist to publish the rule sets to other users.
- **Facebook SDK** as a prerequisite for Firebase authentication.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

Already done. As stated up front, this app is already deployed to the app store with real users. See the project setup in GitHub:
https://github.com/thomasfischersm/LSystemAndroid

Task 2: Implement UI for Each Activity and Fragment

Already done. As stated up front, this app is already deployed to the app store with real users. See the project setup in GitHub:
https://github.com/thomasfischersm/LSystemAndroid

Task 3: Your Next Task

Already done. As stated up front, this app is already deployed to the app store with real users. See the project setup in GitHub:
https://github.com/thomasfischersm/LSystemAndroid

Add as many tasks as you need to complete your app.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"

- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"