GitHub Username: ivanisidrowu

EzPrompter

Description

Simply import your script and start teleprompting it!
You can adjust text font, text color, and scrolling speed for your needs.
What's more, you can use EzPrompter to create, edit, or delete scripts!

Intended User

The intended users are filmmakers, reporters, and speakers.

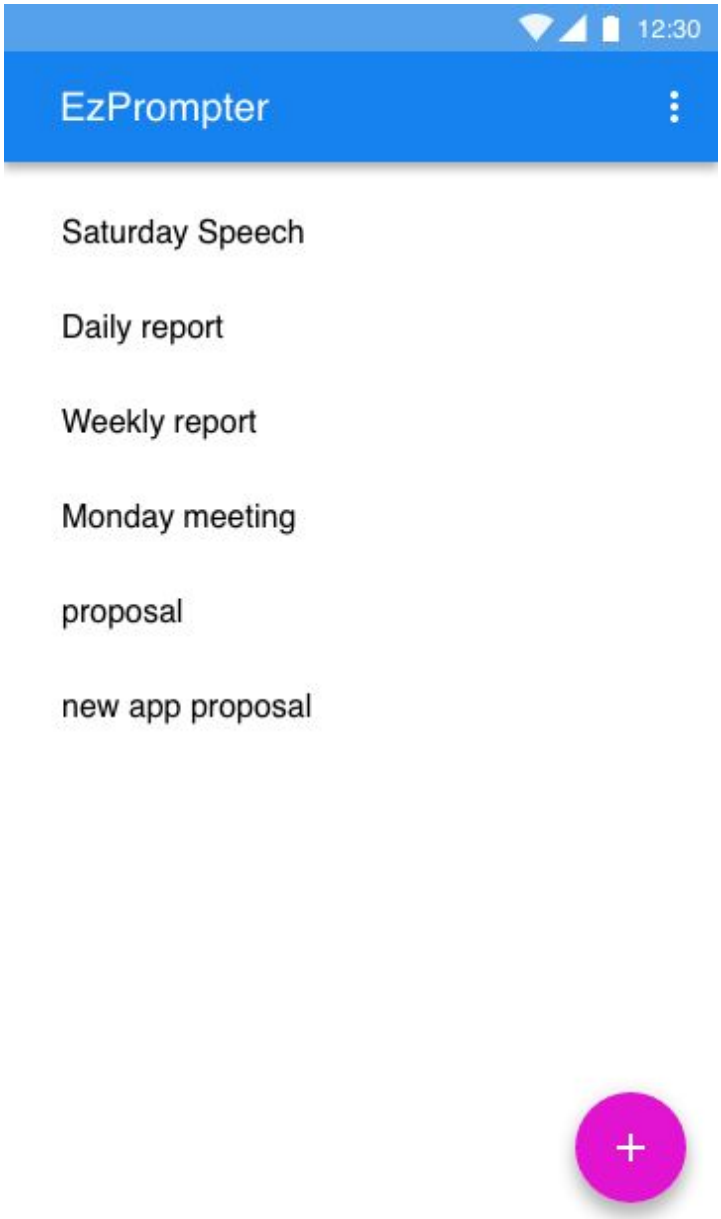
Features

- Import or create txt scripts
- Customize text size, color, font and scrolling speed
- Save, edit, and delete txt scripts
- Auto-start teleprompter

User Interface Mocks

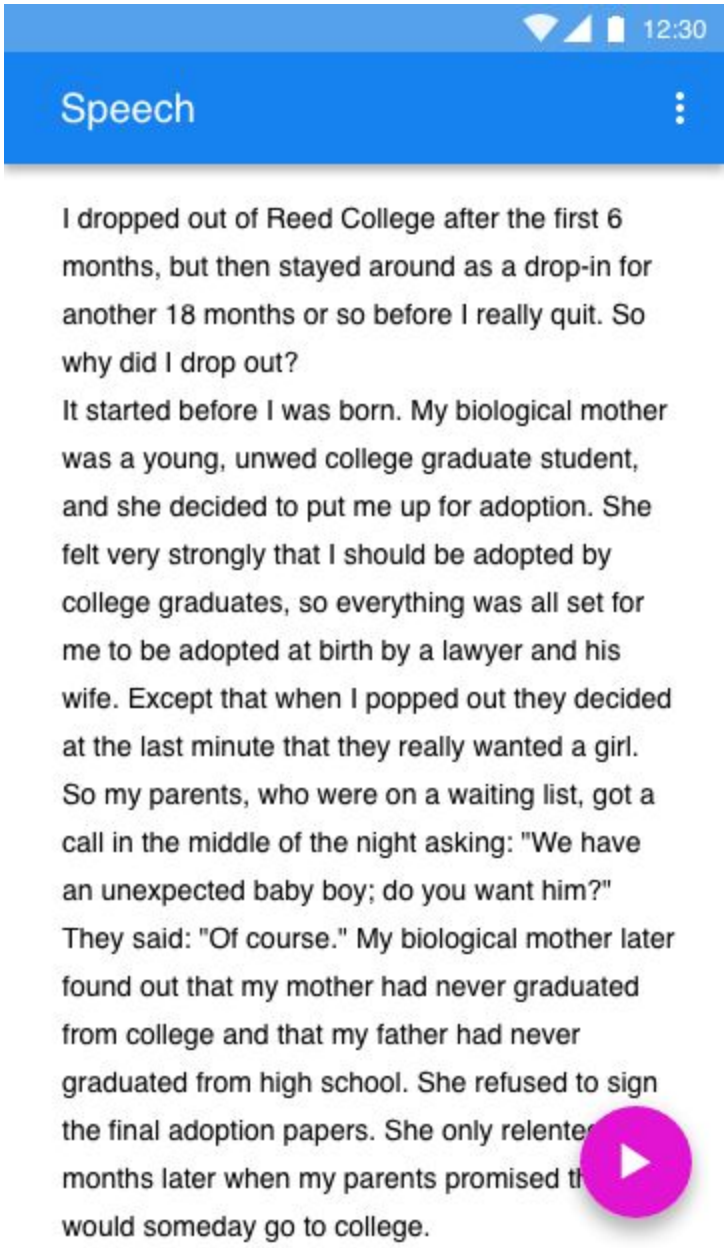
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Main Screen



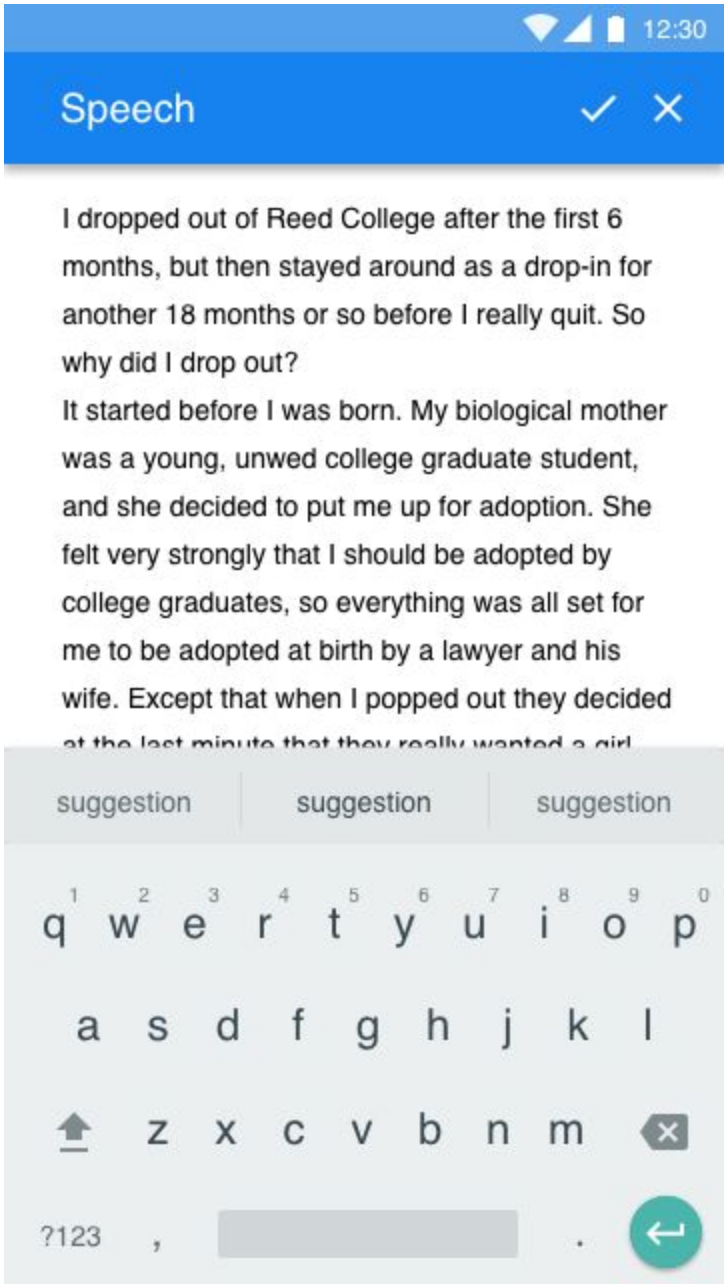
Users can choose to add new script by clicking the floating action button or open files from the list on main screen.

Script Screen



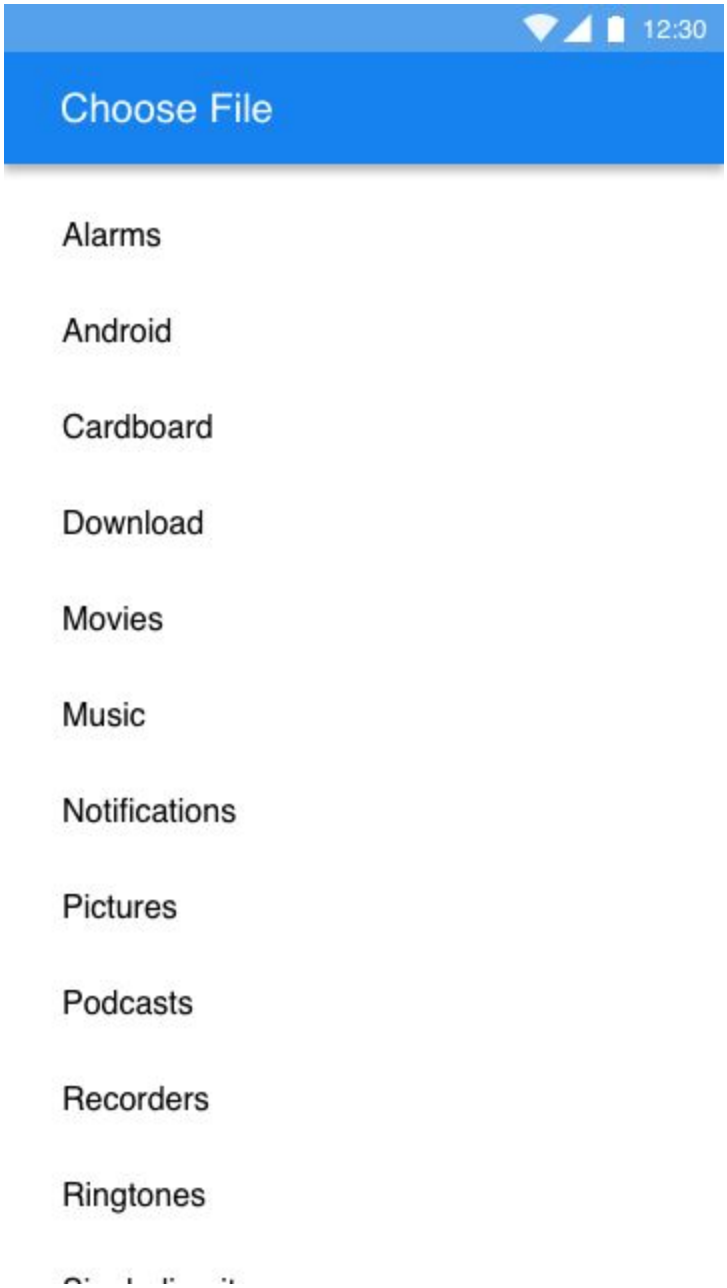
In this screen, users can play the script or click text to edit it.

Editor Screen



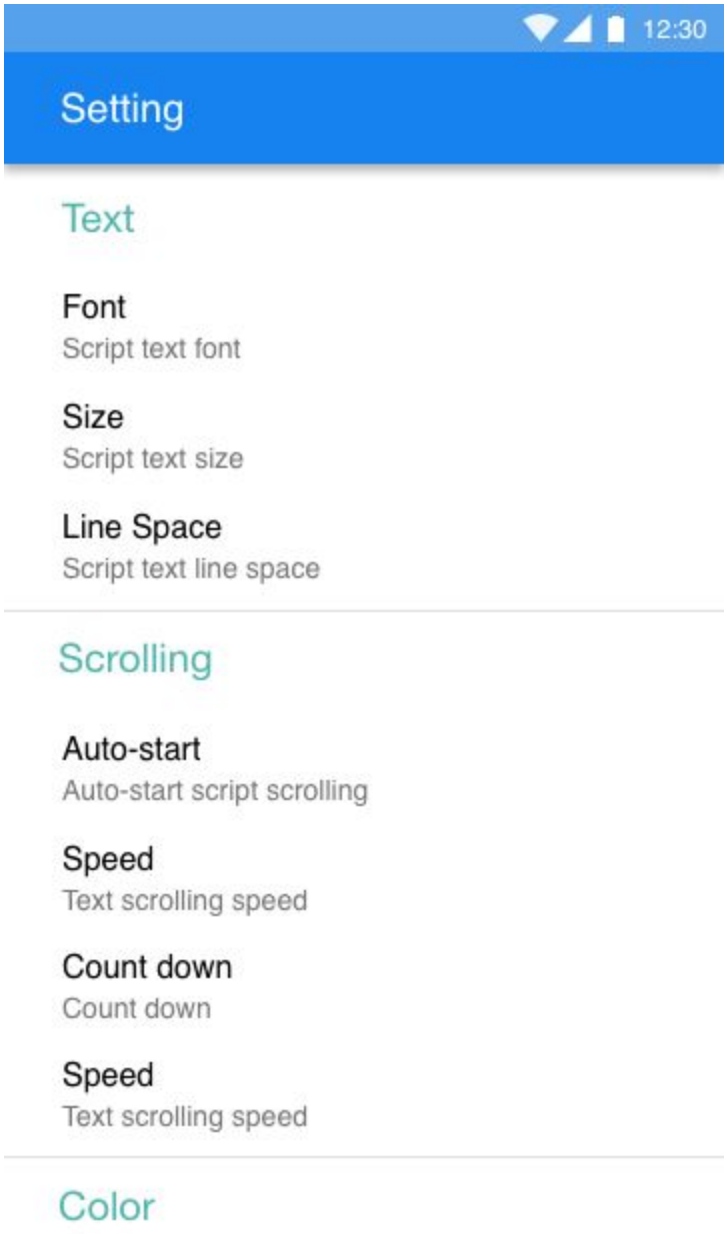
Edit script

File Chooser



Choose a file to import into EzPrompter.

Setting



Adjust script text, scrolling attributes, or colors.

Key Considerations

How will your app handle data persistence?

The application provides functions that users can import or create TXT documents as scripts and show them on the device screen as teleprompter. In addition, users can edit, save, or delete scripts in the application. The application saves Setting data in the SharedPreference, so the setting will be saved on devices.

Describe any corner cases in the UX.

1. When a script is playing, the device orientation should be locked. User rotates the device while playing script will cause the script scrolls to wrong line.
2. Before leaving the script editing mode, EzPrompter will popup a dialog to confirm that user want to leave the editing mode. In this way, we can ensure that users will not forget to save the script before they leave the editing mode.
3. In Setting screen, I'm planning to separate all settings into groups, so users will find wanted settings easily. In order to prevent the complicated setting screen, I will implement the each setting in dialogs. When users click setting item, it will popup a setting dialog of the item.

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

1. ButterKnife - To inject views and resources into Activities or Fragments.
2. Dagger 2 - To inject dependencies in order to make the application loosely coupled and maintainable.
3. RxJava - A Java reactive extension to compose asynchronous processing chain.
4. RxAndroid - A reactive extension for Android. This extension can provide Scheduler that schedules on main thread or loopers.

Next Steps: Required Tasks

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

Task 1: Project Setup

- Open a new project
- Configure libraries
- Configure plugins such as Fabric and Retrolambda
- Configure Dagger2
- Implement EzPrompterApplication for the future use

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for ScriptActivity
- Build UI for EditorActivity
- Build UI for FileChooserActivity
- Build UI for SettingActivity

Task 3: Implement Model

- Implement Script model
- Implement Script builder

Task 4: Implement View

- Define MainView interface
- Define ScriptView interface
- Define EditorView interface
- Define FileChooserView interface
- Define SettingView interface

Task 5: Implement Presenter

- Define MainPresenter interface
- Define ScriptPresenter interface
- Define EditorPresenter interface
- Define FileChooserPresenter interface
- Define SettingPresenter interface
- Implement MainPresenterImpl
- Implement ScriptPresenterImpl
- Implement EditorPresenterImpl
- Implement FileChooserPresenterImpl
- Implement SettingPresenterImpl

Task 6: Implement Activity

- Implement MainActivity
- Implement ScriptActivity
- Implement EditorActivity
- Implement FileChooserActivity

- Implement SettingActivity

Task 7: Testing

- Write unit tests
- Write UI tests
- Final test

Task 8: Publish APP

- Fill in store information on Google Play
- Create keystore
- Export signed APK
- Upload the APK to Google Play