How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: SnehPandya18 (https://github.com/SnehPandya18)

Teleprompter

Description

Teleprompter app will allow users to easily read, edit, update and save text script file. It will be easy-to-read scrolling script for presentations, in front of camera or a group of people. User can type and save the script file from within the app. User can import script file from the gallery and save the script on the device. Once saved, it will show as a list on the main screen of the app.

Intended User

This app is for anyone who is presenting, or rehearsing in front of people or camera. A public speaker, a teacher, a developer, anyone can use as it suits them!

Features

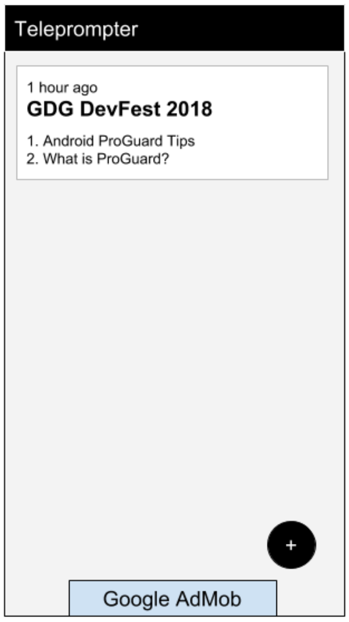
The main features of your app are:

- Saves information in form of text script
- Lists all the saved scripts on main screen
- Can import saved scripts from the device
- Can display scripts via widget on Home screen, if script(s) is available
- Pulls data from Google Slides API via IntentService
- Support for accessibility and content descriptions
- App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts.
- Java language will be used for development

User Interface Mocks

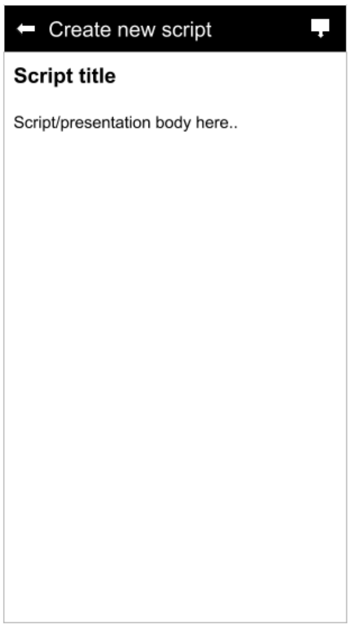
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



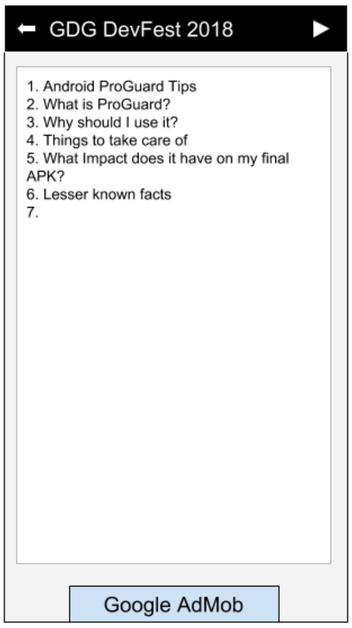
Main Screen: This screen will have list of saved scripts for rehearsing or reading

Screen 2



Add Script screen: This screen will have the script input in form of text

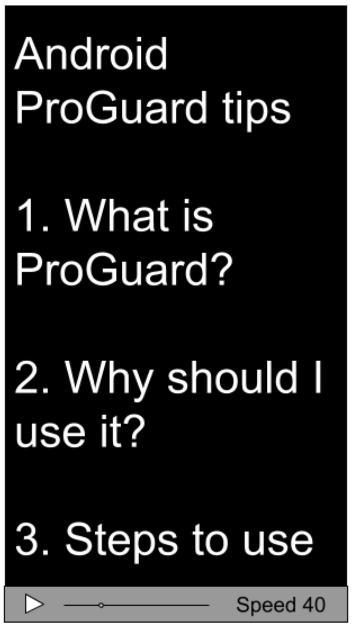
Screen 3



1. Android ProGuard Tips
2. What is ProGuard?
3. Why should I use it?
4. Things to take care of
5. What Impact does it have on my final APK?
6. Lesser known facts
7.

Display Script screen: This screen will have the script displayed in form of text

Screen 4



Android ProGuard tips

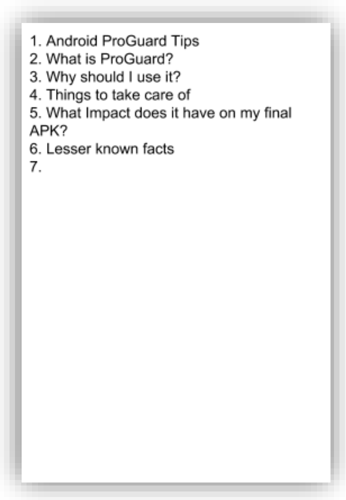
1. What is ProGuard?

2. Why should I use it?

3. Steps to use

Play Script screen: This screen will have the script scrolled in presentation with a dark background for focusing, rehearsals and presentation.

Screen 5



1. Android ProGuard Tips
2. What is ProGuard?
3. Why should I use it?
4. Things to take care of
5. What Impact does it have on my final APK?
6. Lesser known facts
7.

Widget screen: This screen will have the script widget in home screen if a script is available.

Key Considerations

How will your app handle data persistence?

Using Room Architecture Component for data persistence.

Describe any edge or corner cases in the UX.

Handling almost every possible corner case, for example, using Lifecycle Architecture Component to solve Android lifecycle problems. Cases while loading data from the database to prevent ANR, crashes etc. Also improved DAO implementation and null checks to prevent crashes at CRUD operations with data. ViewModel will handle data while screen rotation. Also, network availability check and show snackbar if network is not available, and other corner cases for better UX and navigation.

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

Using ButterKnife library to bind views and write less boilerplate code. Google Drive API to upload notes and scripts from the app to user's Google Drive (Google account required) via API service and appropriate logic, either IntentService or AsyncTask.

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

Using Crashlytics & Firebase core services for tracking events like logs, crashes, etc. Also, using Google Play Mobile Ads services to display ads.

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

Pre-requisites

- Java programming language
- IDE Android Studio Version 3.2
- Android SDK Version 28
- Android Build Tools Version 28.0.3
- Android Support Repository
  - appcompat-v7:28.0.0
  - cardview-v7:28.0.0
  - recyclerview-v7:28.0.0
  - design:28.0.0
  - espresso-core:3.0.2
- Android Architecture Components Room Version 1.1.1
- Android Architecture Components Lifecycle Version 1.1.1
- Firebase Core Version 16.0.5
- Firebase Crashlytics Version 16.2.1
- Fabric Crashlytics Version 2.9.6
- Google Play Services Auth Version 16.0.1
- Google OAuth Version 1.23.0
- Google API Client Version 1.23.0
- Google Drive API Version 16.0.0
- Android MultiDex Version 1.0.3

Tasks:
- Start a “New Project” from Android Studio
- Add required libraries in app level “build.gradle” file
- Add required dependencies and configure gradle setup
- Login to your Google account and setup Firebase services
- Add Firebase dependencies to app level “build.gradle” file and download “google-services.json”
- Save “google-services.json” file under “app” directory
- Setup Google Play Services in app level “build.gradle” file
- Sync everything and build the project and solve the errors if any

Task 2: Implement UI for Each Activity and Fragment

Subtasks:
- Build UI for MainActivity & : To display ScriptListFragment of scripts with title and ads
- Build UI for ScriptListFragment: To display list of scripts
- Build UI for AddScriptFragment: To add text script and save it
- Build UI for PlayScriptFragment: To view already saved script and play it
- Build UI for PlayFragment: To present script text in a huge black presentation view and play it
- Build UI for Widget

Task 3: Develop Room Database and Repository

Third task would be to implement develop ScriptRoomDatabase and repository logic.
Next subtasks:
- Develop DAO for scripts to be saved into database
- Develop entities for database

Task 4: Develop ViewModel and write logic

Next tasks:
- Develop ScriptViewModel which will decide and handle when and how to provide data to UI when data is available

Task 5: Build logic for Widget, check corner cases, loopholes and fix bugs if any

Next tasks:
- Develop ScriptWidget and required connecting logic including WidgetService to update widget
- Connect remaining logic or UI parts with each other for navigation and other functionalities
- Update the whole project by writing corner cases
- Run and check corner cases

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**"