Android Nanodegree Capstone Project Description & Rubric

Table of Contents

Overview

For this project, you will make your own app. You can either come up with your own idea or select from one of the ideas listed under App Ideas. You are welcome to reuse blocks of code from [earlier courses/projects] to create your new app, but the project you design should be entirely your own.

Before you start working on the project:

Review the final project rubric carefully. Think about the following questions:

1. How will you incorporate each of the rubric criterions into your project?
2. Why are these aspects important?
3. What is your strategy to ensure that your project “meets specifications” in the given criteria?

Once you are convinced that you understand each part of the rubric, please start working on your project. Remember to refer to the rubric often to ensure that you are on the right track.

Two Phase Submission Process:

Before you begin building your app of choice, we’d like to review your design. This mimics a common situation you may find on the job where a project is planned, specified and then reviewed before actual development is given a greenlight to commence.

Phase 1: Submit a proposal for your app idea for review

This is what you will do:

Template here

1.Pick an app to build, and draft a one page summary of what the app does, and what problem it solves. Here is an example of a final project description for a Reddit App.

2.Create mocks for all user-facing screens for both mobile phones and tablets. Hand-drawn sketches are OK, but they must be saved in a digital format that you can share with

3.Draft a Project Requirement Document/specification document that outlines a technical plan on how you plan to implement your app idea. (similar to the implementation guides that you've seen through projects within this nanodegree)

- •Specifically call out any libraries you plan to use and for what purpose
- •Outline the data needs of your app and how you plan to manage them (sharedPreferences , Content Provider etc…)

4.Submit this document for review. Once you review approval, you have a Udacity thumbs up to commence building your app!

Phase 2: Submit your app src code for final grading

5.Build your app using your specification documents as a guide!

6.When you are ready to submit, please follow the Final Submission Instructions to submit your project.

Project Requirements

Your Project must:

- •Import and build on the latest version of Android Studio (Eclipse users take note)
- •Be entirely self-contained on an Android device (**No external devices/bluetooth peripherals**).
- •Include a problem description of the problem your app solves.
- •Include mocks for all user-facing screens.
- •Include a signed .apk of your app. Instructions on creating a signed .apk can be found here.
- •Include at least one alternate mock for tablet / large screens.
- •Implement all mockups, including your tablet layout.
- •Have at least two distinct screens (ex. a list view and a detail view).
- •Work properly with the app lifecycle (i.e. you must preserve any dynamic instance state on orientation change).
- •Use permissions responsibly.
- •Use Intents to move between activities inside your app or to an outside app.
- •Create and use your own ContentProvider

- Use Loaders to move your data to your views.
- If the application pulls or sends data to/from a web service or API, it handles this network activity properly (i.e on the correct thread, does not abuse network resources)
- Include only safe-for-work content in your app

App Ideas:

- Food ordering app
- Teleprompter app
- Local news app
- App for your company, university or local club
- Education app
- Public transit schedule
- Reddit Reading List
- Github OSS project tracker

Helpful Resources

If you're looking for action bar icons, you can download the pngs for different screen densities for holo light and holo dark themes from the Android Design Guide downloads page. The Adobe Illustrator source files are also included if you want to style them to match your app.
Action bar icon pack download: https://developer.android.com/design/downloads/index.html
Helpful tools to create assets: http://romannurik.github.io/AndroidAssetStudio/
Other resources:

- http://developer.android.com/distribute/essentials/quality/tablets.html
- http://androidniceties.tumblr.com/
- http://android-developers.blogspot.com/

Capstone Project Rubric

Overview

This rubric is here to help you understand the expectations for how your project will be evaluated. It is the same rubric that the person evaluating your project will use. You should look at the rubric **before you begin working** on this project **and before you submit it**.
The criteria included in this rubric are a subset of the Core App Quality Guidelines that your app would be evaluated against if you submitted it to the Google Play Store. You can find the full list of guidelines here: http://developer.android.com/distribute/essentials/quality/core.html

How Grading Works

1. Your project evaluator will be able to see all of your code submissions. They will use this rubric to evaluate your code as well as your written responses.
2. Your grade will simply be "meets specifications" or "doesn't meet specifications."
    1. You earn "meets specifications" if **all** criteria in the Required Components section "meet specifications".
    2. Your project "doesn't meet specifications" if **any** criteria in the Required Components section are graded as "does not meet specifications." In this case, you will have the opportunity to revise and resubmit your work.

The Rubric

Required Components

To "meet specifications", your app must fulfill all of the criteria listed in this section of the rubric.

Criteria	Does Not Meet Specifications	Meets Specifications
Common Project Requirements		
App conforms to common standards found in the Android Nanodegree General Project Guidelines		
Core Platform Development		
App integrates a third-party library		
App validates all input from servers and users. If data does not exist or is in the wrong format, the app logs this fact and does not crash.		
App includes support for accessibility. That includes content descriptions, navigation using a D-pad, and, if applicable, non-audio versions of audio cues.		
App keeps all strings in a strings.xml file and enables RTL layout switching on all layouts.		
App provides a widget to provide relevant information to the user on the home screen.		
Google Play Services		
App integrates two or more Google services		
Each service imported in the build.gradle is used in the app.		

If Location is used, the app customizes the user's experience by using the device's location.		
If Admob is used, the app displays test ads. If admob was not used, student meets specifications.		
If Analytics is used, the app creates only one analytics instance. If analytics was not used, student meets specifications.		
If Maps is used, the map provides relevant information to the user. If maps was not used, student meets specifications.		
If Identity is used, the user's identity influences some portion of the app. If identity was not used, student meets specifications.		
Material Design		
App theme extends AppCompat.		
App uses an app bar and associated toolbars.		
App uses standard and simple transitions between activities.		
Building		
App builds from a clean repository checkout with no additional configuration.		
App builds and deploys using the installRelease Gradle task.		
App is equipped with a signing configuration, and the keystore and passwords are included in the repository. Keystore is referred to by a relative path.		
All app dependencies are managed by Gradle.		
Data Persistence		
App implements a ContentProvider to access locally stored data.		
Must implement *at least* ***one*** of the three If it regularly pulls or sends data to/from a web service or API, app updates data in its cache at regular intervals using a SyncAdapter. **OR** If it needs to pull or send data to/from a web service or API only once, or on a per request basis (such as a search application), app uses an IntentService to do so.		

OR If it performs short duration, on-demand requests(such as search), app uses an AsyncTask.		
App uses a Loader to move its data to its views.		

Optional Components

Once you have a functioning project, consider adding more features to make your app stand out. Here are a few suggestions:

- •Make your app more delightful with material design patterns such as **shared element transitions** across activities and **parallax scrolling** where two or more items must scroll in the same activity.
- •Implement **notifications** in your app. Remember the following when implementing notifications:
    - •Notifications should not contain advertising or content unrelated to the core function of the app.
    - •Notifications should be persistent only if related to ongoing events (such as music playback or a phone call).
    - •Multiple notifications are stacked into a single notification object, where possible.
    - •Use notifications only to indicate a context change relating to the user personally (such as an incoming message).
    - •Use notifications only to expose information/controls relating to an ongoing event (such as music playback or a phone call).
- •Implement **sharing** functionality in your app, making use of intent extras to share rich content (i.e. a paragraph of content-specific text, a link and description, an image, etc).
- •Create and use a **custom view** in your app that could not be achieved with the standard widgets provided by the core views on Android.