## Android Fundamentals Project Self-Evaluation

**Instructions:** Once you’ve completed your Final Project, please evaluate it against the components of the rubric below. For each criteria that you met, put an “X” in either the “Does Not Meet Specifications” or the “Meets Specifications” box. For some criteria, we ask you to provide an explanation of where and how it was implemented in your app. This is a chance for you to briefly explain to the grader your thought-process during development. Once you are done, include this with the source code and accompanying files you are submitting. Then, give yourself a pat on the back for making a great app!

### 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** |
| Standard Design |  |  |
| App does not redefine the expected function of a system icon (such as the Back button). |  | X |
| App does not replace a system icon with a completely different icon if it triggers the standard UI behavior. |  | X |
| App does not redefine or misuse Android UI patterns, such that icons or behaviors could be misleading or confusing to users. |  | X |
| App includes a tablet layout which takes advantage of the additional space (if possible). |  | X |
| App includes at least two distinct views and uses intents properly to move between these views. |  | X |
| **Navigation** |  |  |
| App supports standard system Back button navigation and does not make use of any custom, on-screen "Back button" prompts. |  | X |
| All dialogs are dismissible using the Back button. |  | X |
| Pressing the Home button at any point navigates to the Home screen of the device. |  | X |
| **Permissions** |  |  |
| App requests only the absolute minimum permissions that it needs to support core functionality. |  | X |
| App does not request permissions to access sensitive data or services that can cost the user money, unless related to a core capability of the app. |  | X |
| **Please elaborate on why you chose these permissions:**  Internet Permission: I chose to use this permission because my app needs to be able to perform http requests against the Utah state government’s public notice api.  Read\_sync\_settings, Write\_sync\_settings, Authenticate\_accounts: I chose to use these permissions so that I could implement a sync adapter.  Read\_calendar, Write\_calendar: These permissions were chosen so that I could interact with the user’s calendar allowing them create create a calendar event related to a public notice they may have interest in.  Set\_Alarm: This permission was chosen so that I could set notifications to occur in the future. A user can choose to have my app notify them 1 hour before a meeting begins. |  |  |
| **Performance and Stability** |  |  |
| App does not crash, force close, freeze, or otherwise function abnormally on any targeted device. |  | X |
| **ContentProvider** |  |  |
| App implements a ContentProvider to access locally stored data. |  | X |
| 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.  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. |  | X |
| App uses a Loader to move its data to its views. |  | X |
| 1. **What's the content provider called, and how is it backed?**   I named my content provider the PublicNoticeProvider. It is backed by an sqlite database   1. **What backend does it talk to? What is the SyncAdapter called? What mechanism is used to actually talk over the network?**   It talks to the Utah government’s public notice api (<http://www.utah.gov/developer/>)  . My sync adapter is called the PublicNoticeSyncAdapter.  The component that actually talk’s to the network is the UtahOpenGovApiProvider and it simply wraps up a URL connection and returns nicely formatted pieces of data.   1. **What loaders/adaptors are used?**   I use a couple of cursor adapters and cursor loaders in a fashion very similar to sunshine.  Specifically there is a NoticeAdapter class that extends Cursor adapter and then my fragments make use of the LoaderManager.LoaderCallbacks interface. |  |  |
| **User/App State** |  |  |
| App correctly preserves and restores user or app state. |  | X |
| When the app is resumed after the device wakes from sleep (locked) state, the app returns the user to the exact state in which it was last used. |  | X |
| When the app is relaunched from Home or All Apps, the app restores the app state as closely as possible to the previous state. |  | X |
| **Please elaborate on how/where your app correctly preserves and restores user or app state:**  My app will store the user’s home town as a preference and use that preference to ensure they get up to date information about public notices.  I also took a cue from sunshine and made sure that my listview fragment remembers the user’s selection between screen rotations so that the user experience is not interrupted after a rotation completes. |  |  |

### 

### Optional Components

To receive “exceeds specifications”, your app must fully implement all of the criteria listed under at least two of the four categories below (e.g. Notifications, ShareActionProvider, Broadcast Events, and Custom Views).

|  |  |  |
| --- | --- | --- |
| **Criteria** | **Does Not Exceed Specifications** | **Exceeds Specifications** |
| Notifications |  |  |
| Notifications do not contain advertising or content unrelated to the core function of the app. |  | X |
| Notifications are persistent only if related to ongoing events (such as music playback or a phone call). |  | X |
| Multiple notifications are stacked into a single notification object, where possible. |  | X |
| App uses notifications only to indicate a context change relating to the user personally (such as an incoming message). |  | X |
| App uses notifications only to expose information/controls relating to an ongoing event (such as music playback or a phone call). |  | X |
| **Please elaborate on how/where you implemented Notifications in your app:**  I chose to implement notifications via an overflow menu option in my NoticeDetailFragment class.  The user while viewing a full public notice can choose to have the app give them a notification 1 hour before a meeting begins with that overflow menu option. |  |  |
| ShareActionProvider |  |  |
| Uses ShareActionProvider to share content with an outside application. |  | X |
| Makes use of Intent Extras to send rich content (i.e. a paragraph of content-specific text, a link and description, an image, etc). |  | X |
| **Please elaborate on how/where you implemented ShareActionProvider:**  I chose to implement the share action provider in my NoticeDetailFragment.  I make use of the intent Extra\_Subject and Extra\_Text to make a richer shared content experience. The app will share a given public notice’s title, time, date, and location.  I really enjoyed implementing this feature since before I took this course I had not heard of or used the ShareActionProvider. |  |  |
| Broadcast Events |  |  |
| App intercepts broadcast events. |  | X |
| App responds to Broadcast events in a meaningful way. |  | X |
| **Please elaborate on how/where you implemented Broadcast Events:**  I implemented a broadcast receiver that listens for an alarm manager intent and the app responds to this broadcast by sending out the relevant notification based on the information passed in the broadcast. |  |  |
| **Custom Views** |  |  |
| App creates and uses a custom View. | X |  |
| App uses a novel View that couldn’t sufficiently be satisfied by the core Views in Android. | X |  |
| **Please elaborate on how/where you implemented Custom Views:** |  |  |