Capstone Project App Proposal

GitHub Username: https://github.com/jackblas/

App Name: JB Wallpaper

Description

Browse images library provided by NASA APDO (Astronomy Picture of the Day) service and select any of the images from the library as your wallpaper.

The app includes a widget displaying latest picture added to the APOD service.

Intended Users

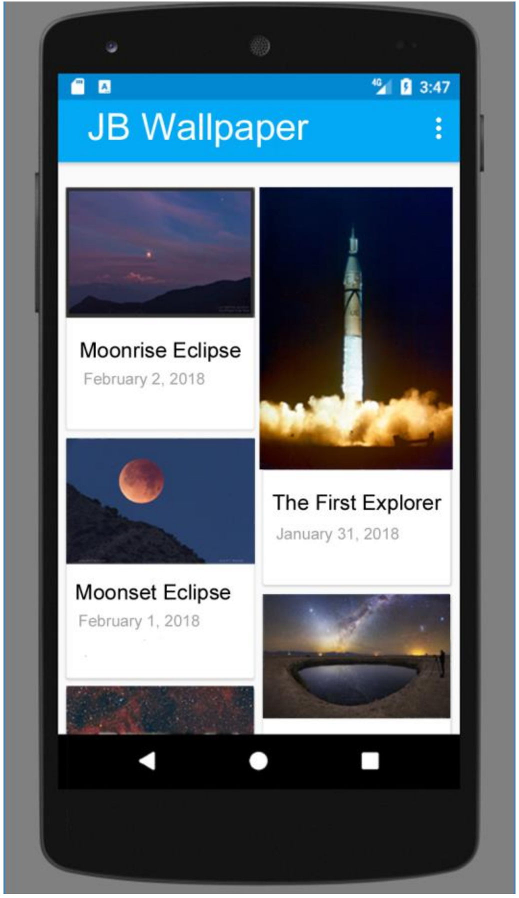
Everyone

Features

- Browse pictures from the APOD service
- Set selected image as the wallpaper
- Automatically set the latest image as your wallpaper
- Notifications for the picture of the day
- Includes a widget displaying latest picture added to the APOD

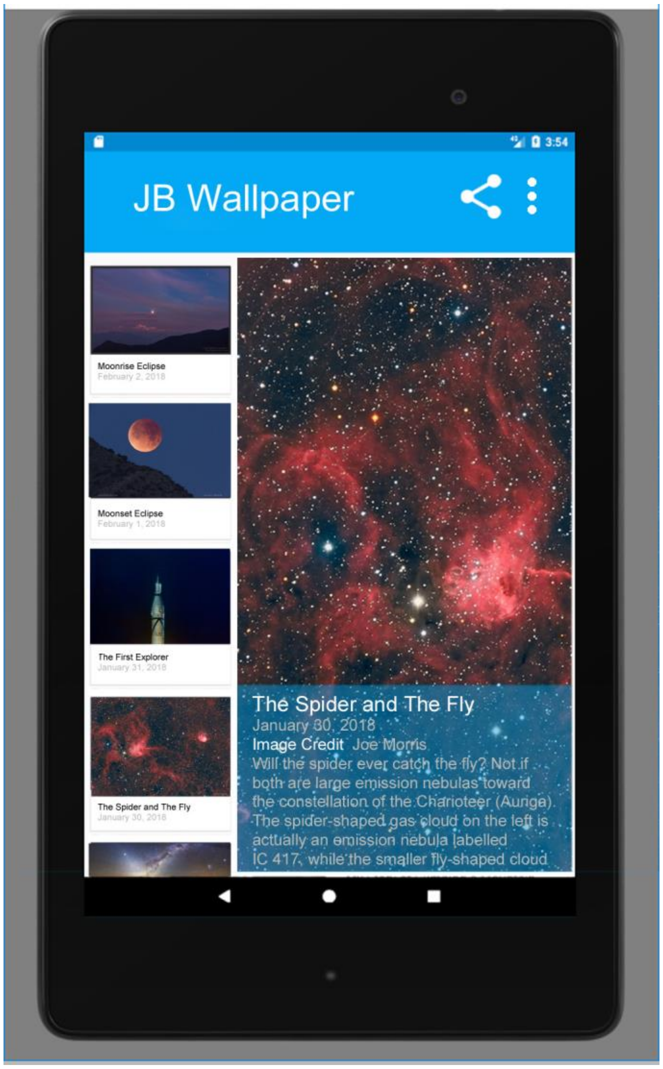
User Interface Mocks

Main Activity Screen - Mobile



Main Activity – Displays all images

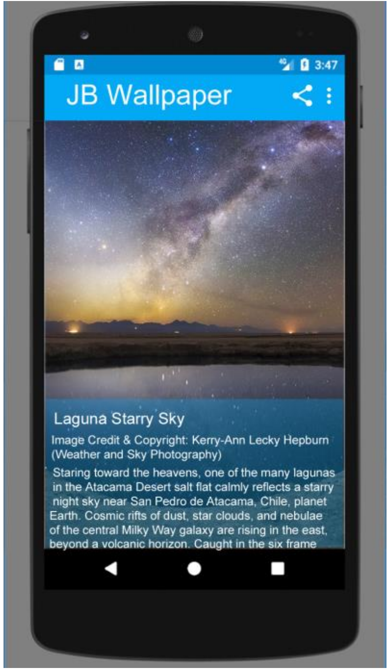
Multi-pane Main Activity Screen - Tablet



The Spider and The Fly
January 30, 2018
Image Credit Joe Morris
Will the spider ever catch the fly? Not if both are large emission nebulas toward the constellation of the Charioteer (Auriga). The spider-shaped gas cloud on the left is actually an emission nebula labelled IC 417, while the smaller fly-shaped cloud

On a tablet-sized screen, the Main Activity layout contains both images list fragment and selected image detail fragment.

Details Activity Screen



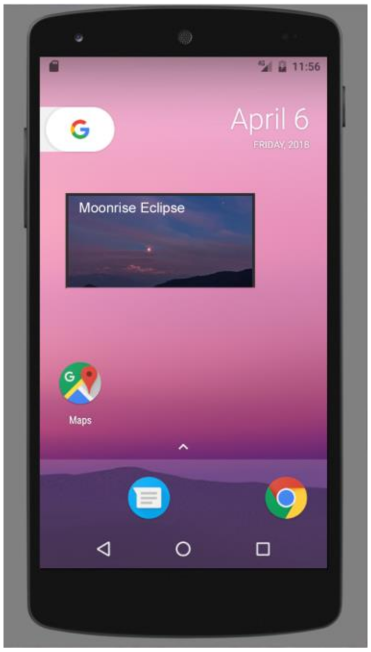
Laguna Starry Sky

Image Credit & Copyright: Kerry-Ann Lecky Hepburn (Weather and Sky Photography)

Staring toward the heavens, one of the many lagunas in the Atacama Desert salt flat calmly reflects a starry night sky near San Pedro de Atacama, Chile, planet Earth. Cosmic rifts of dust, star clouds, and nebulae of the central Milky Way galaxy are rising in the east, beyond a volcanic horizon. Caught in the six frame

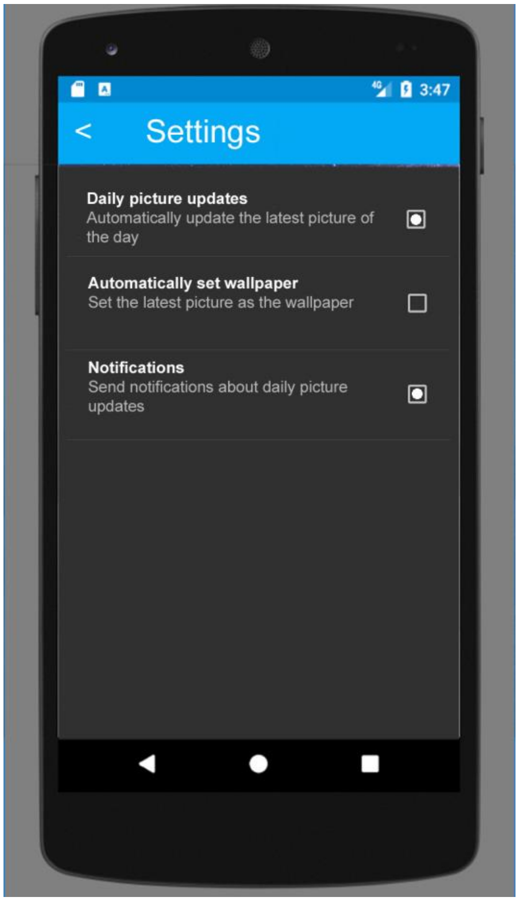
Details Activity – Displays selected image with description.

App Widget



The app includes a 2x1 app widget. The widget displays the most recent picture with a title.

Settings Activity Screen



3:47

< Settings

Daily picture updates
Automatically update the latest picture of the day ☑

Automatically set wallpaper
Set the latest picture as the wallpaper □

Notifications
Send notifications about daily picture updates ☑

Settings Activity - allows users to modify app features.

Available preferences:

- Daily Picture Updates
  - Automatically update the latest picture of the day: Yes/No (Default-Yes)
- Automatically set wallpaper
  - Set the latest picture as the wallpaper: Yes/No (Default-No)
- Notifications
  - Send notifications about daily picture updates: Yes/No(Default-Yes)

Options Menu

Main Activity menu items:

- Settings

Details Activity menu items:

- Settings
- Set As Wallpaper
- Share

Key Considerations

How will your app handle data persistence?

The app will not store data locally; however, the app will use Content Provider object to access data from APOD API. Content Provider is used here mainly to facilitate loading data into the UI views using a Cursor Loader. Also, if future versions of the app require data storage in a local database, Content Provider module can be updated without affecting the rest of the code.

Daily Picture Update

One a day, the app will use Intent Service to pull latest image data from the server. This service will be scheduled by Alarm Manager (on devices running below API level 21) or by Job Scheduler for API level 21 and higher.

Describe any edge or corner cases in the UX.

Return to the app from the widget:

1. When the user hits image in the latest picture widget, the Details Activity opens.

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

Libraries used in project:

1. Picasso - Image loading library

Advantages:

- Automatic memory and disk caching provides fast and optimized image download and efficient use of memory to transform images to better fit into UI layouts.
- Supports download and error placeholder images.
- Downloading images in a separate tread

2. OkHttp – Efficient Http Client

Advantages:

- faster loading
- saves bandwidth
- recovery from common connection problems

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

Google Play Services implemented:

1. AdMob

The app will display test banner ads. A banner AdView object will be added to the Main Activity layout. Detail implementation steps are listed in the Required Tasks section.

2. Firebase Analytics

The app will be integrated with Firebase SDK. Firebase Analytics will be used to collect basic app usage data such as first open and ad click events. Detail implementation steps are listed in the Required Tasks section.

Next Steps: Required Tasks

Task 1: Project Setup

Subtasks:

- Configure libraries
- Configure/Setup Google Play Services
- Obtain API key from api.nasa.gov

Task 2: Implement UI for Each Activity and Fragment

Subtasks:

- Build UI for Main Activity
- Build UI for Details Activity

- Build UI for tablet multi-pane Main/Detail Activity

Task 3: Implement Scheduling of Daily Picture Download Service

Subtasks:

- Implement IntentService – to download daily images from the server; sends NEW_DAILY_PICTURE broadcast intent.
- Register the service in manifest
- Implement AlarmManager to execute the download service (for API below level 21)
- Implement BroadcastReceiver to re-start AlarmManager after device reboot
- Register the receiver for the android.intent.action.BOOT_COMPLETED broadcast
- Implement JobService class to use Job Scheduler to execute the download service on devices running API 21 and above.
- Register job service in manifest

Task 4: Implement Content Provider

Subtasks:

- Implement UriQuery class. This utility call will expose one public method:

  ```
  public Cursor queryDataFromServer()
  ```

  The method will get a JSON string from the server and return data as a Cursor. A MatrixCursor class will be used to convert JSON string into a cursor. This method will be called within the query() method of the Content Provider.

- Implement Contract class
- Implement Content Provider class
- Declare Content Provider in manifest file

Task 5: Implement Main Activity

Subtasks:

- Create layout for list fragment
- Implement Adapter for List Fragment data
- Implement List Fragment class
- Implement Main Activity class

Task 6: Implement Details Activity

Subtasks:

- Create layout for details fragment
- Implement Details Fragment
- Implement Details Activity

Task 7: Implement Settings Activity

Subtasks:

- Define Settings preferences in res/xml/preferences.xml
- Define values and labels for List Preferences in res/values/arrays.xml
- Implement Preference Fragment class

Task 8: Implement Notifications

Subtasks:

- Create and register a standalone Broadcast Receiver to handle NEW_DAILY_PICTURE broadcast intents send by the daily download service (Task 3)
- In the onReceive() method create Notification object

Task 9: Implement APOD Widget

Subtasks:

- Create widget's layout file
- Define widget's properties in a AppWidgetProviderInfo xml file
- Implement AppWidgetProvider class
- Add widget configuration to the manifest file
- Implement service to update data in the widget

http://www.vogella.com/tutorials/AndroidWidgets/article.html

Task 10: Implement AdMob

Subtasks:

- Import the Mobile Ads SDK
- Add a Gradle dependency
- Initialize MobileAds in MainActivity onCreate() method
- Add AdView to the MainActivity layout
- Load an ad

Task 11: Implement Firebase Analytics

Subtasks:

- Create Project in Firebase console
- Add Firebase to your Android app
- Copy config file google-services.json into app module root dir
- Add gradle dependencies for Firebase Analytics

Task 12: Upload App to Google Play

Subtasks:

- Create signed APK
- Set up App store Listing
- Upload and publish the APK

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**”