Project 1

RSS Reader - Due 1/24 @ 11:59 pm

For Easy Reading, Go to View -> Disable Print Layout

Background

News has been on a steady rise with regard to the content, stories, and data shared and broadcasted to the world. With the various ups and downs from Covid 19 to stock market and other occurrences throughout 2020 and now in 2021. This app is designed to help identify and track data found through the United Nations Feed via the following RSS Feed link: https://news.un.org/en/rss-feeds. We will be exclusively using the “Feeds by topic” section for this project.

- **Helpful Tip:** When you click through the various topics under “Feeds by topic” take note of what changes in the url and what doesn’t change. This will be useful when addressing the toggling between various topics in Phase 3.

Technical Description

For this project we will be using fundamental Kotlin programming and Layouts in order to develop an application that can show valuable information summarized in a list. We will also use Intents (Explicit and Implicit) to change between different screens. This project will also cover implementing the more widely accepted and industry approved version of lists.

Note

As android is an evolving language, at times you may get alerts from AndroidStudio saying certain portions are deprecated, however please don’t try to update such areas since the project may not work as intended if you do.

Key Concepts

- Kotlin and Xml
- RecyclerView

- Layouts
- Activities and Intents

Requirements

Screen 1: List of Topics

- This is the “Home Page” and has a list of topics in individual cards
    - One topic title per card
    - No overlap between cards

Screen 2: List of Articles

- Display List of Articles
    - Article titles
    - Article description
    - Publication date
- Clicking an Article Card sends the user to Screen 3

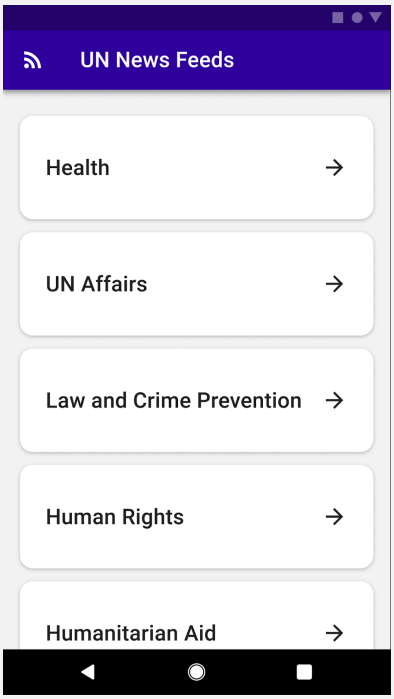
Screen 3: View Data of One Article

- Article title
- Article description
- Button that sends user to article web page
    - Launch site via browser when clicked
- Display UN logo image (display the same image for every article)

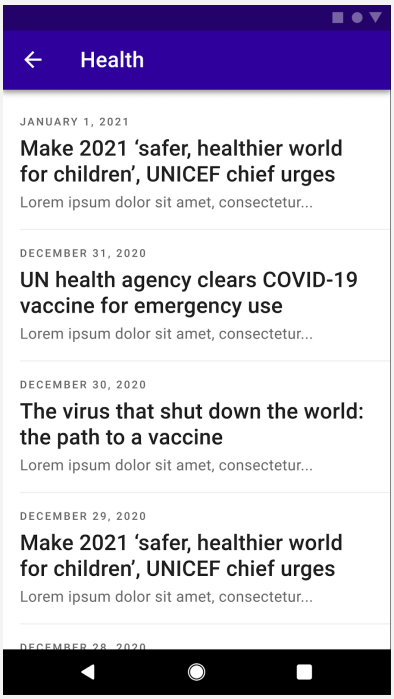
Article in Web Browser

- IMPORTANT: Don’t create this page from scratch
- This “page” is reached from Screen 3’s button

Screens



MainActivity
(Screen 1)



FeedActivity
(Screen 2)

NewsArticleActivity
(Screen 3)

Web Page

Architecture Breakdown

Given Core Pieces

- Libraries
- APIs
- General graphic assets
- Code Outlines on where to place code

Students need to develop

- Recycler View
- Layout Mechanics
- Activities
- Generating Intents

Given/Expected Files

The project itself is on GitHub

Project File Structure

- **Bold Black files -** are ones you need to complete the codebase with logic
- **Bold Blue files** - are ones you need to create and implement yourselves
- Any non-formatted files should be left alone

*The files below are listed in no particular order

- ❏ App
    - ❏ Java
        - ❏ Com.android.example.rssreader
            - ❏ Model
                - ❏ Channel.kt
                - ❏ Item.kt
                - ❏ RSSWrapper.kt
            - ❏ FeedActivity.kt

- ❏ FeedApi.kt
- ❏ FeedTopic.kt
- ❏ **MainActivity.kt**
- ❏ **FeedActivity.kt**
- ❏ **NewsArticleActivity.kt**
- ❏ **RSSFeedAdapter.kt**
- ❏ **TopicAdapter.kt**

❏ Res

❏ Layout

- ❏ **Activity_feed.xml**
- ❏ **Activity_main.xml**
- ❏ **Activity_news_article.xml**
- ❏ ***Feed_topic.xml***
- ❏ ***rss_feed_item.xml***

Gradle Scripts

- Do not touch anything in here

Resources

- What is an RSS Reader?
- UN RSS Feed

Phases

Phase 0: Understanding the Codebase

There is code already provided to you, make sure that you can understand and coordinate with the given code. This is done to make your jobs easier so that you can make apps with more functionality. The code base is sprinkled with all the phases.

Here are some key files to look through, but look through as many as you see fit. MainActivity is like the main() of most Android Projects

- MainActivity.kt (Activity_main.xml)
- FeedActivity.kt (Activity_feed.xml)
- NewArticle.kt (activity_new_article.xml)

- Item.kt
- FeedTopic.kt

Phase 1: RSS Feed (Screen 2)

This is the screen where users can scroll through the list of articles before choosing one to read.

In this Phase you will need to take a look at **`FeedActivity.kt`** and **`activity_feed.xml`**, you will also need to create the necessary components for displaying a List of Articles.

By default the “heath” feed from UN’s RSS Feed is being fetched. The data is being stored into a list of ***Item*** objects. In this phase you need to create a RecyclerView and populate it with the News Article Items from this list. Once you create the RecyclerView you will need to send the data to Screen 3 to view more information when the user clicks on an article in the list.

Phase 2: Article View (Screen 3)

Before users spend time reading an entire article, they may want to know a little bit about the contents. On this screen we want to display the information for one entry in the rss feed (article title, description, publication, date). If the user decides that they want to read the full article, we should present the user with an option to do so and send them to a browser to view the article.

This view should contain:

- Title of the article
- Date
- Summary/Description
- Button to visit article site

This can be achieved by working with the **`activity_new_article.xml`** layout file that corresponds to the **`NewArticle.kt`** kotlin file. This means being able to set up the basic layout components via the layout editor or XML code via respective views for the article’s corresponding title, date and summary. Make sure to add a button for visiting the article.

Also make sure that the button for visiting the article site carries an implicit intent so that the article will open on the default browser.

Phase 3: Select RSS Feed (Screen 1)

This is the screen where users can first go to examine the various types of topics that they may be interested in learning more about when scrolling through this linear layout.

- Create RecyclerView using a list of FeedTopic items
    - Define the adapter, viewholder, and layout
- There is a section of code that you need to play-test and then remove:

```
// ========== PHASE 3 : Remove this section if needed from here ==================
// TODO: Remove this section
/* Send User to FeedActivity to view UN News Article by Topic */
val topic = FeedTopic("Health", "health")
// TODO: try switching above line with below and see what happens on FeedActivit...
// val topic = FeedTopic("Human Rights", "human-rights")

val intent = Intent(this, FeedActivity::class.java)
intent.putExtra("feed", topic)
this.startActivity(intent)
// ========== PHASE 3 : to here ===================================================
```

- For each item in the recyclerview, send an intent to go to FeedActivity and send the FeedTopic that was selected
- There are several options that should be provided for the user to select. Note in the above example we have **FeedTopic(“Health”, “health”)**, “Health” is the topic of the feed and “health” is a portion of the rss feed url that is used on Screen 2. Below are the other feeds that should be supported
    - **TOPIC,** **URL-PIECE**
    - "Health", "health"
    - "UN Affairs", "un-affairs"
    - "Law and Crime Prevention", "law-and-crime-prevention"
    - "Human Rights", "human-rights"
    - "Humanitarian Aid", "humanitarian-aid"
    - "Climate Change", "climate-change"
    - "Culture and Education", "culture-and-education"
    - "Economic Development", "economic-development"
    - "Women", "women"
    - "Peace and Security", "peace-and-security"
    - "Migrants and Refugees", "migrants-and-refugees"

- "SDGs", "sdgs"

Phase 4: UI design

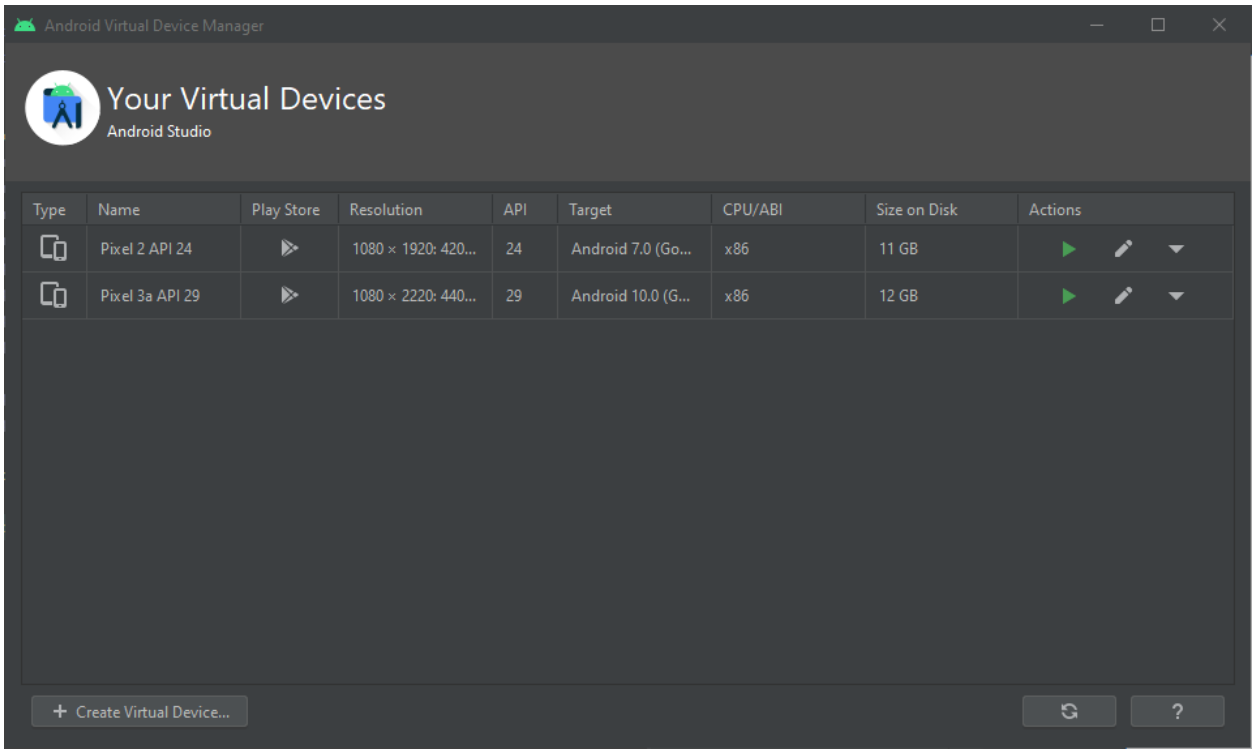
After completing all the above phases, take this time to clean all the User Interface. Do your best to make it look nice and presentable. We have provided mockups (under ‘Screens’ section) created by a designer.

Submission

Projects will be submitted through gradescope. Please only have one partner submit the project and add the other partner to the gradescope submission so they can see it. (More details about the submission coming soon)

Additional Notes

- Make sure your emulator supports the google play store. You can verify this by checking if there is an icon under the play store columbine the Android Virtual Device Manager..



- **Mac Users with M1 Chips ONLY**
    - Mac M1 laptops currently do not have full support for the Android Studio emulator, but Android has since put out a preview emulator
        - https://github.com/741g/android-emulator-m1-preview/releases/tag/0.1
        - Download the android-emulator-m1-preview.dmg file
    - The Android Emulator will be its piece of software which must be opened manually before clicking run in Android Studio
    - Launching the Web Page may not fully function as expected. You may only see a white screen pop up, but you should still see the correct URL being displayed in the web search bar