Internship Technical Review by Jayna Menard

Purpose: In-depth review on first individual project assigned at the company FloreoLabs.

**Internship (FloreoLabs)**

**Personal notes**

My company FloreoLabs is a start-up lab for those who need a place of support to help them with starting their first project or creating a business. To do so, mentors are used to help guide these individuals along. Being an addition to such a company, I have learned many things most computer science majors do not learn in school. I was taught how to debug code properly, how to create bug reports, expanding my technical vocabulary, using resources like Trello to create boards based on a project’s completed/non-completed areas and more. The hardest part for me was learning a new language. Kotlin is a modern form of Java that gives us the same properties of any Java code but in a compacted form. Having experience in Java, I was able to distinguish the similarities between Kotlin and Java. Learning a new language is always difficult, but the experience I received from the TTP bootcamp and the support given to me by a mentor in my internship, I was able to create my first android app.

**Task: New York Times API in Kotlin**

For our first task, we created a restful front-end android application to mimic the New York Times application Top Stories API.

As a User, I:

* [ ] Load application on any Android
* [ ] See Top Stories from the “Home” Page of the original New York Times application
* [ ] Scroll through the home page and see an image, title, and abstract describing the current event

The requirements were to use RetroFit2, RecyclerView, Picasso, and LifeCycle. Retrofit is a class through which your API interfaces are turned into callable objects. RecyclerView is used for construction of any lists with XML layouts as an item which can be vastly customized. Picasso is an image library that caters to image loading and processing. Lifecycle performs actions in response to a change in the lifecycle status of another component, such as activities. To implement these dependencies, **build.gradle(Module: app)** must be modified. Using many online resources, I was able to find the dependencies and implement them as so:

A screenshot of a computer

Description automatically generated

To begin, Three XML layout were created in the **res** folder to imitate the New York Times news application. These are **activity\_main.xml**, **article\_card.xml**, and **content\_main.xml**.

**content\_main.xml** contains an ImageView for the NYT logo, a toolbar to display the name of page, a progressBar to imitate a “Loading Screen”, and RecyclerView for XML layout and scrolling. **Article\_card.xml** uses an ImageView for article images and two TextViews for the title and abstract. **Activity\_main.xml** is used to display them.

Now, a unique API key is required. The one used for this project is New York Times Developer Networks “Top Stories”. To view the JSON data, we must select one of the topic URLs (EndPoint) on the website and add on our own API key.

A screenshot of a social media post

Description automatically generated

The target data, **results**, is nested within another array. To enter the array of objects, three models were needed to receive the data - **SectionM.kt**, **ArticleM.kt**, and **MultimediaM.kt**.

The SectionM.kt file handles the first array in the JSON data – **results**. Referencing our image above, **results is a list of Arrays**. Then, we must write down the what we want the Article to look like. The title, abstract, and multimedia is what we wanted to see. So, **ArticleM.kt** hasvariables **title**,**abstract**, and **multimedia**. Title and abstract being strings and **multimedia is a list of multimedia**. To define the information we want from the multimedia array, a string variable **url** is created.

A screenshot of a computer screen

Description automatically generated

**Interface.kt** is then created to receive the necessary info from the correct source using Retrofit. From the URL above, the endpoint will be used for the get request, the path will be the section selected (in this case, “US”), and the query for the API key generated calling **Section.kt** so we can enter display the target data.

Next, an object class (**NYTService.kt**) is created to use Retrofit’s built in Builder and GsonConverterFactory class to convert JSON to GSON, making it readable to browsers. Retrofit.Builder() class uses the Builder API to allow defining the URL end point for the HTTP operations and finally build a new Retrofit instance. GsonConverterFactory is called done via create()with the method addConverterFactory() and it will automatically take care of mapping the JSON data to your Kotlin objects.

**getArticle.kt** uses MutableLiveData from the dependencies Lifecycle is an observable data holder. By passing the reference of the Activity (**<Section>**), it notifies the lifecycle owner with updates and then the UI redraws itself with updates. Then, the API is implemented and the **{section}** part of the endpoint wanted is then initialized. (ViewModel is responsible for preparing and managing the data of the Activity).

A screenshot of a cell phone

Description automatically generated

To use this class, **FetchJson.kt** is referenced. Using Retrofit, **FetchJson.kt** is used to fetch and parse the Json. To execute the request asynchronously with the enqueue method, Retrofit’s Callback class must be used, which expects the implementation of two methods (onResponse and onFailure).

**MainActivity.kt,** implements the applications basic startup logic. To handle the different stages of application, the Activity class provides six callbacks: onCreate(), onStart(), onResume(), onPause(), onStop(), and onDestroy(). We only implement two (onResume and onPause) but the rest have their own built in code that is automatically run by Android Studios. Finally, to run the app, permissions are needed to access the internet. To do so, the internet permission is added in **AndroidManifext.xml**.

A screenshot of a cell phone

Description automatically generated

To see the full application, visit <https://github.com/JaynaMMenard/NYTimesAPI.git>.

In the past few months, I have learned a lot about what it means to be computer scientist. I must be dedicated and patient with any task, no matter how big or small. What I hope to learn is to be confident in my skills. I want to easily dive into any subject and retain said information so I may use what/how I learned a language with other various languages. What I aim to accomplish is to teach others from the mistakes I made and along the way learn from them so we can all be better computer scientists.