Not A Basement Studio
#4.13, Thien Son Plaza, 800 Nguyen Van Linh Str.
Tan Phu Ward, District 7, Ho Chi Minh City, Vietnam

Image Downloader

Assignment Background

As a member of the iOS team, you are tasked with the R&D project to improve the downloading feature of one of our popular app “Manga Rock”.

This R&D project aims to equip our dev team with understanding of the latest iOS backend APIs such as: URLSession and NSProgress, before we decide to use them in our production app.

SPECIFICATIONS

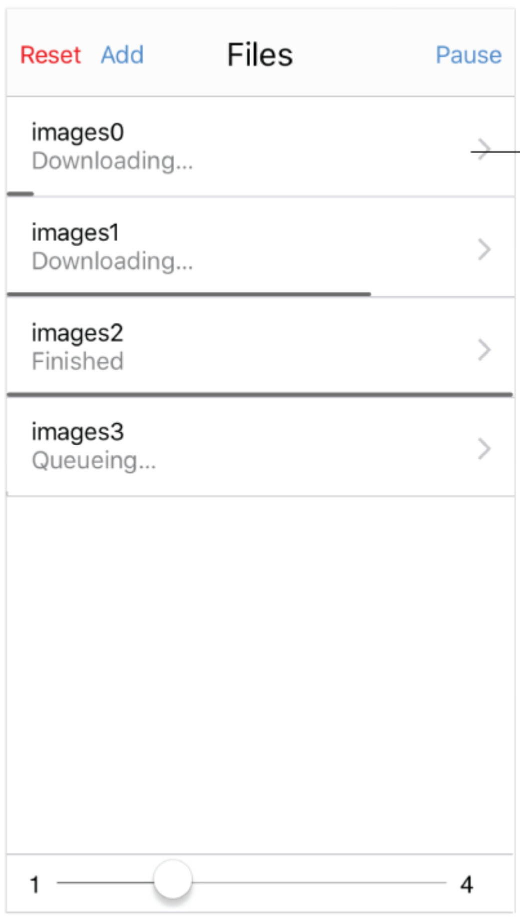
Overall requirement

Our R&D project is a simple download manager app allows user to download the content from this url:
https://storage.googleapis.com/nabstudio/Developer/iOS/Interview/Image%20Downloader/JSON%20files%20updated.zip

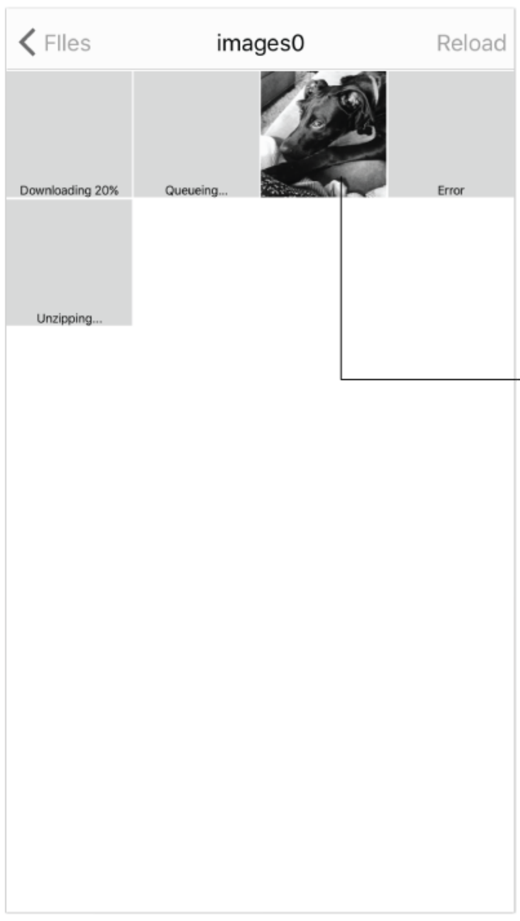
The zip file in the above url contains multiple JSON files. Each JSON file contains an array of URLs pointing to different resources. Each resource can be one of the following types: JPG, PNG, PDF, or ZIP.

After extracting the zip file, the app also downloads all the resources, in those JSON, and display them as images. It automatically converts PDF and ZIP resources to image before displaying to the UI.

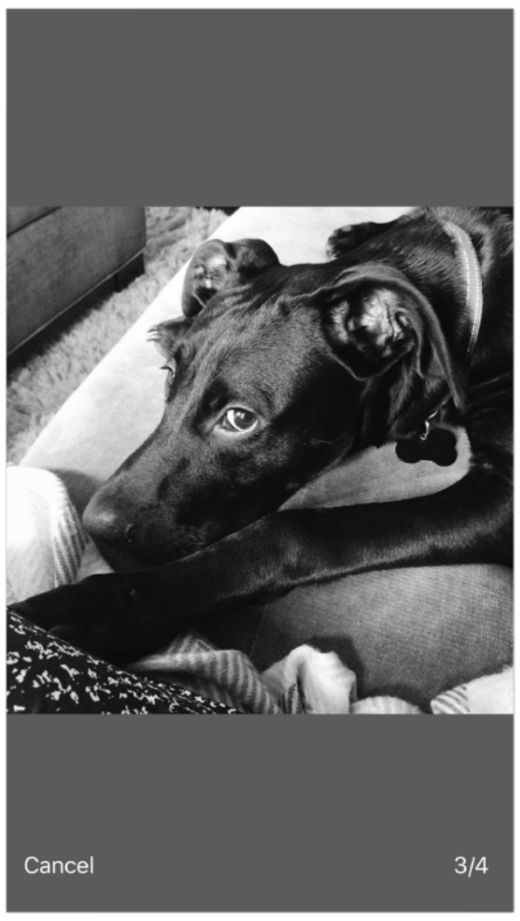
This app contains 3 screens as listed below:



(figure 1)



(figure 2)



(figure 3)

Screen 1

This screen has a Table View containing a list of JSON files.

Each cell contains:

- Title: JSON file name
- Subtitle: Downloading status. This status text can be one of the followings:
    - “Downloading…” : when some of resources listed in the JSON file are still downloading.
    - “Queuing…” : when all resources listed in the JSON file are still waiting to be downloaded.
    - “Finished” : when all resources listed in the JSON file are downloaded.
- Progress bar: show the current progress of number of downloaded resources listed in JSON file.

The screen also has a slider at bottom, which controls the maximum concurrent number of resources can be downloaded at a time.

Tap on “Add”:

- The app downloads the zip file from url: download link then populate its data into the app, and display as seen in figure 1.

Tap on “Reset”:

- Stop all download processes and delete everything.
- The list should be empty like on first launch.

Tap on “Pause” / “Resume”:

- Pause or resume all download processes.
- This button should be disabled when there is no activity.

Tap on each cell:

- Tap on it will open Screen 2

Screen 2

This screen has a Collection View displays all the resources listed in the JSON file.

Each cell contains:

- Image: thumbnail image of the resource downloaded. Gray color if the resource is downloading, queueing or failed to download
- Progress text: Downloading status. This status text can be one of the followings:
    - “Downloading xx%” : if the resource is downloading, xx is the downloaded percentage of the resource file.
    - “Queueing”: if the resource is waiting to be downloaded.
    - “Error”: if the resource failed to download or render the resource file.
    - “Unzipping”: if the resource requires unzipping.

Tap on “Reload”:

- Force redownload all the resources

Tap on each cell:

- Tap on it will open Screen 3

Screen 3

This screen is a simple Viewer to browse all the resources in each JSON file.

- If the resource is not yet available, please show a gray image.

Tap on “Cancel”:

- Tap on it will close the Viewer.

DELIVERABLES

Required

- Duration: **3 days** starting from the date you received this document.
- Programming language: **Objective-C**
- Code cleanliness and readability matter.
- **Only use Apple APIs** to support Networking, Concurrency, and Progress reporting; however, you are allowed to use minizip (or any third-party zipping library) to support unzipping.
- Use iOS **Background Transfer Service** to support downloading when app is suspended.
  Sample code: Simple Background Transfer
- The app must be able to continue the downloading process in case of crash or being terminated.
- Use **NSProgress** for progress reporting.
  Sample code: Photo Progress

Optional

- You are free to improve the look and feel of this app as you see fit.
- Support all device screens (with Split Screen and Slide Over) using iOS 10 adaptive layout features.