satyadevchauhan / **FlickrSearch**

Flickr search application written in Swift

#flickr-api #flickr-collections #swift #imagedownload #operationqueue #networkmanager #collectionview #uber #uber-assignment

5 commits **1** branch **0** packages **0** releases **1** contributor

Branch: **master** | New pull request | Create new file | Upload files | Find file | Clone or download

satyadevchauhan Removed unused variable. Latest commit 11326a5 on Oct 1, 2018

FlickrSearch.xcodeproj	Added unit testing and updated readme.	2 years ago
FlickrSearch	Removed unused variable.	2 years ago
FlickrSearchTests	Added unit testing and updated readme.	2 years ago
FlickrSearchUITests	Initial Commit	2 years ago
.gitignore	Initial commit	2 years ago
README.md	Updated readme file.	2 years ago
Screenshot1.png	Added unit testing and updated readme.	2 years ago
Uber Mobile Coding Challenge - iOS.pdf	Initial Commit	2 years ago

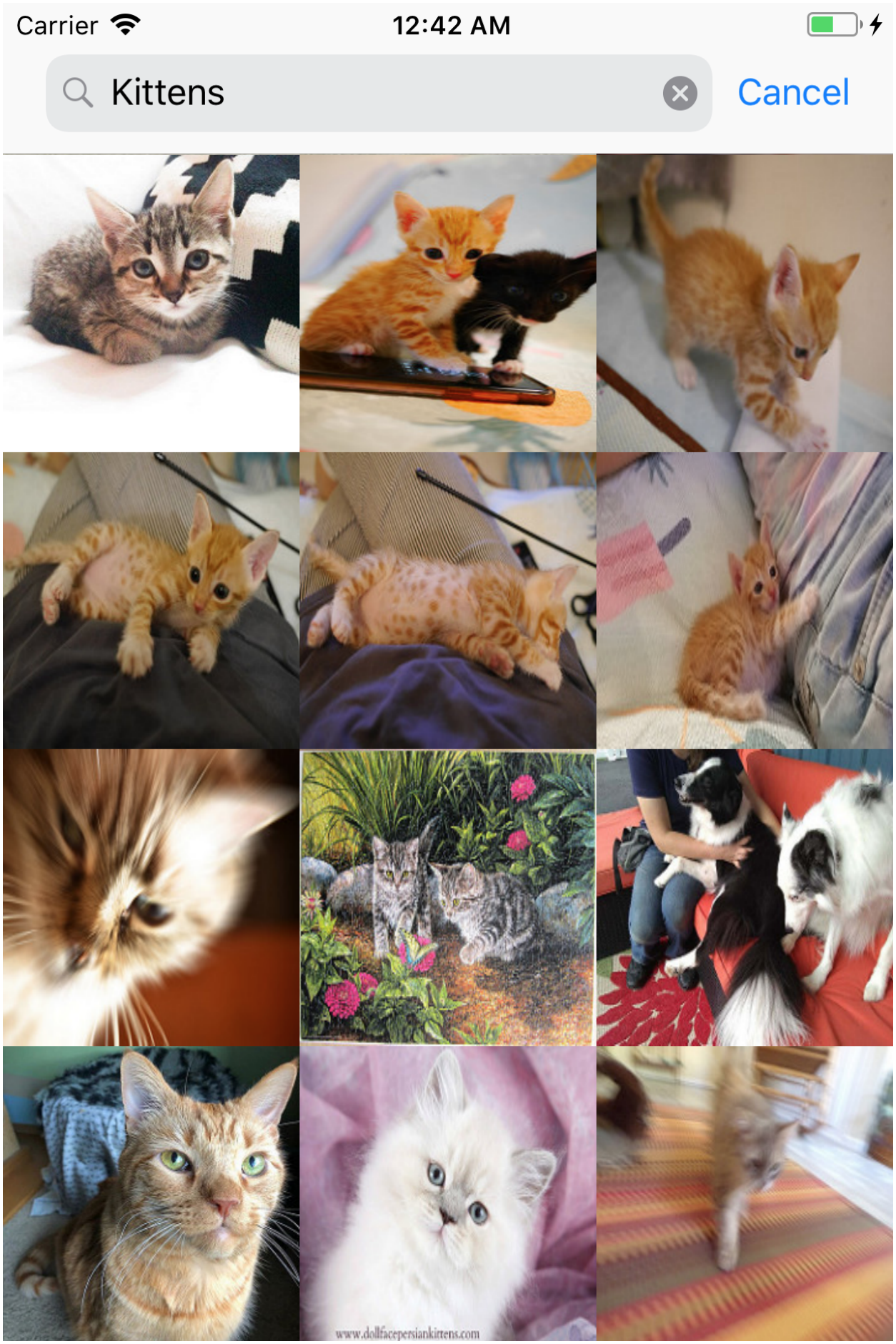
README.md

FlickrSearch

Flickr search application written in Swift. This project searches for images using Flickr API and displays in 3 column grid and provides endless scrolling. The data is paginated and service takes care of fetching data from Flickr service. The application uses MVVM architecture.

Getting Started

- Clone the repo and run FlickrSearch.xcodeproj
- Create a Flickr API key and replace placeholder with key in FlickrConstants.swift
- Run the project and search for any keyword like "kittens".
- The application doesn't uses any third party library.


Carrier
12:42 AM
Kittens
Cancel
www.dollfacepersiankittens.com

Flickr API

Images are retrieved by hitting the Flickr API.

- **Search Path**: https://api.flickr.com/services/rest/?method=flickr.photos.search&api_key={fickr_api_key}&format=json&nojsoncallback=1&safe_search=1&per_page={page_size}&text={search_text}&page={page_num}
- **Example**: https://api.flickr.com/services/rest/?method=flickr.photos.search&api_key=a4f28588b57387edc18282228da39744&format=json&nojsoncallback=1&safe_search=1&per_page=60&text=kittens&page=1
- Response includes an array of photo objects, each represented as:


```
{
"id": "43213681030",
"owner": "164058447@N08",
"secret": "a4bf8df905",
"server": "1937",
"farm": 2,
"title": "Puss under the boot",
"ispublic": 1,
"isfriend": 0,
"isfamily": 0
}
```


We use the farm, server, id, and secret to build the image path. Flickr Photo Source URLs

- **Image Path**: http://farm{farm}.static.flickr.com/{server}/{id}_{secret}.jpg
- **Example**: https://farm8.staticflickr.com/7564/15981410640_a0d5006167_m.jpg
- Response object is the image file.

Class Details

FlickrSearch

This module consists of all files related to Flickr search and presenting on UI. A single view that contains a UICollectionView to display the retrieved images in a three-column layout. It supports endless scrolling and automatically loads next pages when the user approaches the end of the current page. The search bar supports searching for multi-word phrases not only a single keyword. The view supports all orientations and uses that autolayout to arrange items inside the collection view.

Views

- **ViewControllers**: This module consists of primary class **FlickrCollectionViewController** which consists of collection view and search text field. It also encapsulates functionality like fetching, refreshing and searching. On fetching data it binds this data with viewmodel and hence render it.
- **ImageCollectionViewCell**: This is the view which is reused for displaying images fetched from service. It also renders image by downloading it asyncronously using **ImageDownloadManager** and also handles some important usecase like cancelling downloading while recycling.

ViewModels

- **FlickrViewModel**: This class represents data to be rendered in **FlickrCollectionViewController** via bindings.
- **ImageModel**: ImageModel represents data to be rendered in collection view cells, this is been generated while views are recycled.

Models

- **FlickrSearchResults**: This class represents the structure of response of request from flicker service for the searched text. This class uses codable protocal and used for parsing.
- **FlickrPhoto**: This class represents the structure of pagination details and array of images from **FlickrSearchResults**.

This class uses codable protocal and used for parsing.

- **Photos**: This model represents data structure for images details. It encapsultes all related info and hence generates url for fetching images. This class uses codable protocal and used for parsing.

Services

- **FlickrSearchService**: This service class is responsible for preparing the request, fetching and parsing response for consecutive pages. It internally uses **NetworkManager** to perform the request.

NetworkManager

This module consists of classes related to network used to fetch the stream of data, careate request and check reachability of internet.

- **NetworkManager**: This class is basically used to fetch data from server. It is used to fetch data using http protocol like GET, POST etc. Has 2 methods on which fetches data as Data type and Image. It takes a URL and retrieve a stream of data contained at this URL no matter what kind of data it is. The advantage of this abstraction is that many other modules in the app can use the same module to retrieve different kinds of data such as images, json ... etc
- **ImageDownloadManager**: This class is responsible for caching, queuing and fetching images from server. It is uses **ImageDownloadOperation** to fetch the data from server.
- **ImageDownloadOperation**: This class uses the **NetworkManager** class to download an image from Flickr API and returns the image object.

UnitTests

This module consists XCTTest classes for testing.

- **FlickrNetworkTests**: This class tests the network interactions for fetching image and data, and vallidating its consistency. UI testing is not done due to time constraints.