Image Cache (feat. Storage)

목차

- Image Cache
- Storage

Image Cache?

- 캐싱은 모든 소프트웨어 개발(iOS 개발뿐만 아니라)에서 사용되는 매우 일반적인 기술
- 인터넷에서 다운로드한 이미지를 **임시로 저장**하여 재사용
- 따라서 이미지가 화면에 다시 나타날 경우, 다운로드하지 않아도 됨.
- 종류 (2가지)
  - memory cache : 기기를 끄면 사라짐
  - disk cache : 기기안에 저장 되어있고 껐다 켜도 남아 있음

Image Cache 구현 방법

- 1st party (SwiftUI에서 기본 제공)
  - NSCache : 현재 세션에 대해서만 저장되는 임시 개체를 위한 것
  - FileManager : 장치에 영원히 저장되는 보다 영구적인 개체를 위한 것

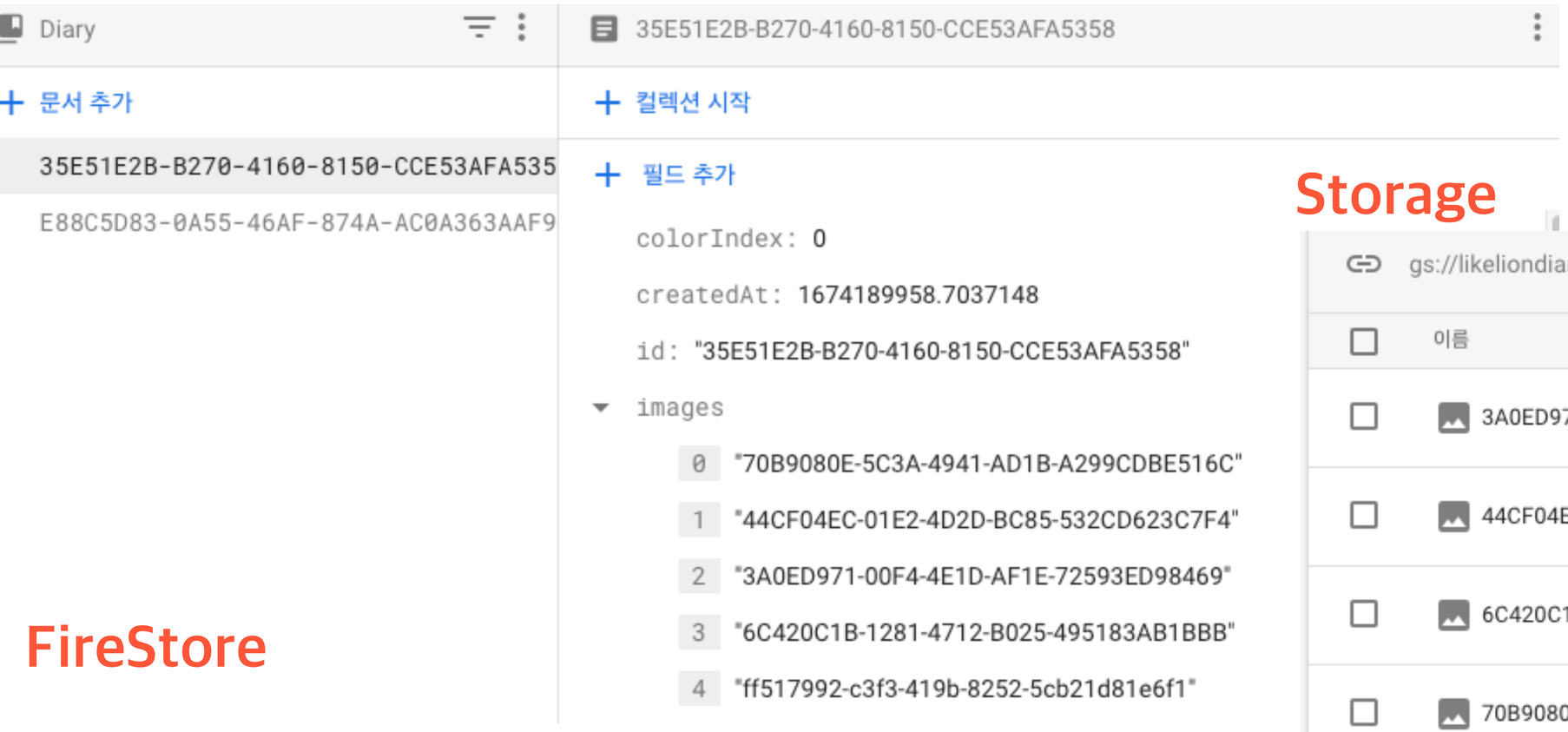
- 3rd party libraries
  - Kingfisher , Nuke , SDWebImageSwiftUI 등등

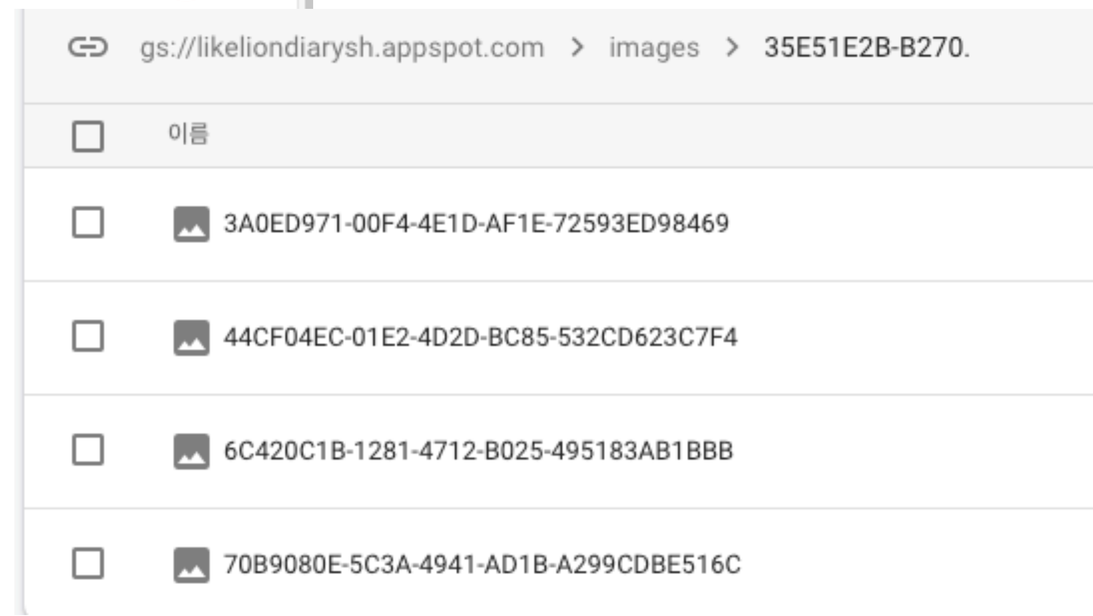
NSCache Basic ver.

```swift
class CacheManager {

    static let instance = CacheManager() // Singleton
    private init() { }

    var imageCache: NSCache<NSString, UIImage> = {
        let cache = NSCache<NSString, UIImage>()
        cache.countLimit = 100
        cache.totalCostLimit = 1024 * 1024 * 100 // 100mb
        return cache
    }()

    func add(image: UIImage, name: String) {
        imageCache.setObject(image, forKey: name as NSString)
        print("Added to cache!")
    }

    func remove(name: String) {
        imageCache.removeObject(forKey: name as NSString)
        print("Removed from cache!")
    }

    func get(name: String) -> UIImage? {
        return imageCache.object(forKey: name as NSString)
    }
```

Firebase Storage?

- Firestore에는 큰 용량을 저장할 수 없어 사진이나 동영상은 Storage에 업로드하여 사용





Storage 코드(1)

```
import Firebase
import FirebaseFirestore
import FirebaseStorage    // for saving image to Storage
import UIKit

class DiaryStore: ObservableObject {
    @Published var diaries: [Diary] = []
    @Published var imageDict: [String: UIImage] = [:]

    let database = Firestore.firestore()
    // 참조 (클라우드의 파일에 대한 포인터)
    let storage = Storage.storage()
```

Storage 코드(2)

```
// storage에서 fetch하는 함수
// imageName: diary.id + "/" + imgName
func fetchImage(diaryId: String, imageName: String)    {
    print("imageName: \(imageName)")
    let ref = storage.reference().child("images/\(diaryId)/\(imageName)")

    // Download in memory with a maximum allowed size of 1MB (1 * 1024 * 1024 bytes)
        ref.getData(maxSize: 1 * 1024 * 1024) { data, error in
            if let error = error {
                print("error while downloading image\n\(error.localizedDescription)")

            } else {
                let image = UIImage(data: data!)
                self.imageDict[imageName] = image
                print("imageDict: \(self.imageDict)")
            }
        }
}
```

Storage 코드(3)

```
// uploadImage 함수의 name parameter에 full reference path를 넘겨주게 했다
// name: diary.id + "/" + imgName
func uploadImage(image: UIImage, name: String) {
    let storageRef = storage.reference().child("images/\(name)")
    let data = image.jpegData(compressionQuality: 0.1)
    let metadata = StorageMetadata()
    metadata.contentType = "image/jpg"

    // uploda data
    if let data = data {
        storageRef.putData(data, metadata: metadata) { (metadata, err) in
            if let err = err {
                print("err when uploading jpg\n\(err)")
            }

            if let metadata = metadata {
                print("metadata: \(metadata)")
            }
        }
    }

}
```

참고 자료

- [Image Cache] https://www.avanderlee.com/swiftui/downloading-caching-images/
- [Image Cache] https://nsios.tistory.com/58
- [NSCache 에 대한 기본적인 설명] https://www.youtube.com/watch?v=yXSC6jTkLP4
- [NSCache 에 대한 기본적인 설명2] https://www.youtube.com/watch?v=fmVuOu8XOvQ
- [Storage] https://devyan.tistory.com/43#recentEntries