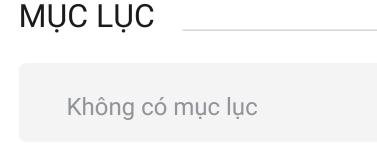


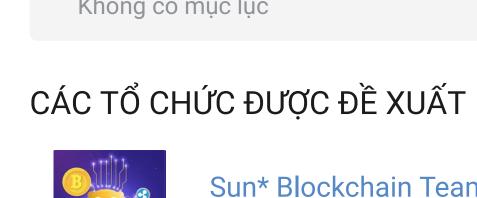
Hiển thị tiến độ download bằng **URLSession**

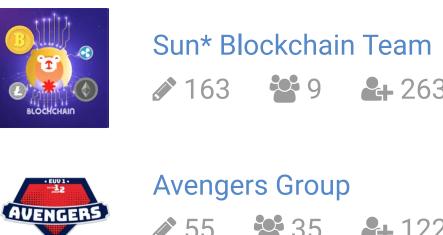
Hà Tuấn Thịnh @ha.tuan.thinh Theo dõi

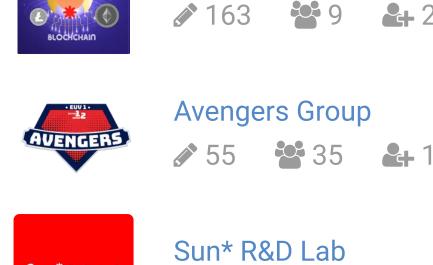
★ 999 **♣** 42 **♦** 14













000 Cái này tưởng đơn giản mà lại làm mình mất khối thời gian, chỉ vì mình không biết đến việc cái này override cái kia, dẫn đến một số cái không chạy như ý. Do đó mình viết bài này để bạn nào chưa biết cách làm thì sẽ biết cách làm, bạn nào chưa hiểu tại sao mình

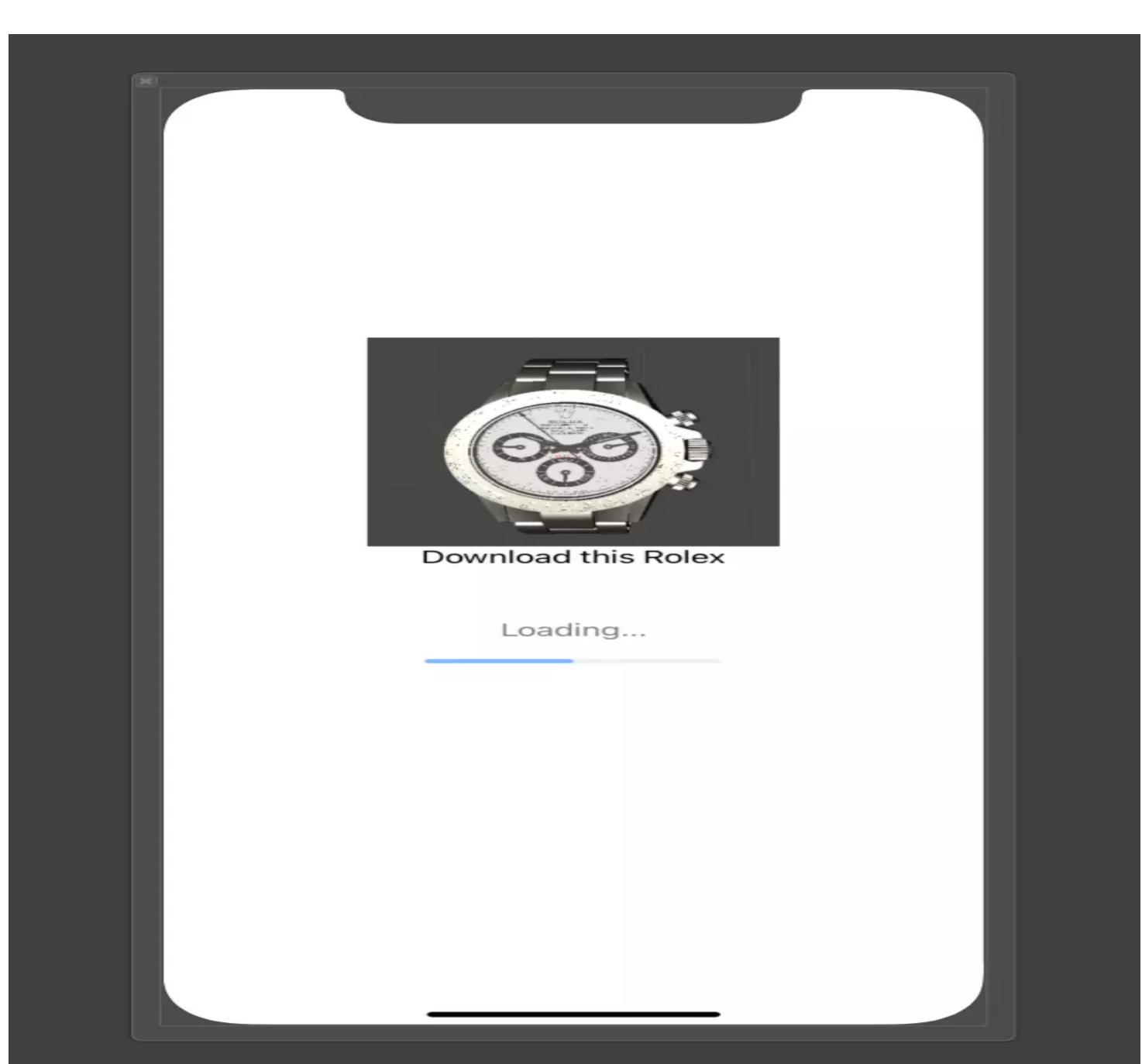
viết trông chuẩn lắm rồi mà nó vẫn không chạy thì hiểu được nguyên nhân nhé. Hiển tiến độ download là một chức năng khá cơ bản trong các thể loại ứng dụng. Trong Swift việc này không khó. Về nguyên lý thì như sau: Khi bạn download một file thì hệ

thống sẽ liên tục cập nhật cho bạn các thông tin về quá trình download. Các thông tin này gồm tổng lượng byte cần phải download, lượng byte vừa được download (trong lần báo cáo này), và tổng lượng byte đã được download (tính tổng các lần báo cáo). Mỗi lần báo cáo cách nhau bao nhiêu lâu thì mình cũng không rõ lắm, nhưng thấy khá là liên tục.

Vậy để tính xem đã download được bao nhiêu % thì chúng ta lấy tổng lượng byte đã download chia cho tổng lượng byte phải download rồi nhân 100 là ra. Mà mình không thích dùng % nên trong bài này mình sẽ dùng ProgressView.

Phần setup project chắc bạn nào cũng biết rồi nên mình bỏ qua.

Về giao diện thì mình vẽ nhanh như sau:



dưới là thanh *ProgressView* để hiển thị tiến độ download. Mình để cái ảnh đồng hồ để giả vờ đang download cái đồng hồ cho nó giống thật tí.

Ở màn hình này, khi user bấm vào nút download thì mình sẽ hiện chữ loading kèm bên

ViewController minh đặt tên là *InitialViewController*:

Label "Loading..." với thanh ProgressView ban đầu mình để *hidden*.

```
class InitialViewController: UIViewController {
    @IBOutlet weak var loadingLabel: UILabel!
    @IBOutlet weak var loadingProgressView: UIProgressView!
```

URLSession. Khi user bấm vào cái đồng hồ mình sẽ gọi hàm này:

Giờ vào phần code. Để download file mình sẽ dùng hàm downloadTask của

func loadAssetFromServer() {

```
showLoading()
        let path = "http://192.168.1.4:8080/"
        let url = URL(string: path + fileName)!
        let session = URLSession(configuration: .default, delegate: self, d
elegateQueue: .main)
        session.downloadTask(with: url).resume()
```

hiện label "Loading..." với cái ProgressView.

Mình tự host 1 cái server local để test cho tiện. Hàm showLoading() là mình dùng để

// MARK:- URLSessionDownloadDelegate

Đối với tiến độ download chúng ta sẽ chú ý đến hàm

Phần quan trọng sẽ là đoạn này:

```
let session = URLSession(configuration: .default, delegate: self, delegateQ
ueue: nil)
session.downloadTask(with: url).resume()
```

URLSessionDownloadDelegate. Do đang test nhanh nên mình để delegate là self, và dùng cái *InitialViewController* hiện

Để nhận được tiến độ download, chúng ta sẽ cần implement protocol

tại để implement luôn. Trong thực tế có thể bạn sẽ muốn tạo class riêng để implement protocol này. extension InitialViewController: URLSessionDownloadDelegate {

```
func urlSession(_ session: URLSession, downloadTask: URLSessionDownloadTask, didFinishDownloadingTo
        location: URL) {
    func urlSession(_ session: URLSession, downloadTask: URLSessionDownloadTask, didWriteData bytesWritten:
        Int64, totalBytesWritten: Int64, totalBytesExpectedToWrite: Int64) {
Hàm
```

func urlSession(_ session: URLSession, downloadTask: URLSessionDownloadTask

, didFinishDownloadingTo location: URL) là nơi bạn xử lý tiếp sau khi download xong.

```
func urlSession(_ session: URLSession, downloadTask: URLSessionDownloadTask
, didWriteData bytesWritten: Int64, totalBytesWritten: Int64, totalBytesExp
ectedToWrite: Int64)
```

totalBytesExpectedToWrite là tổng lượng byte cần phải download, và bytesWritten là tổng lượng byte đã download.

Để thay đổi *ProgressView* cho ứng với tiến độ download chúng ta làm như sau:

```
func urlSession(_ session: URLSession, downloadTask: URLSessionDownloadTask
, didWriteData bytesWritten: Int64, totalBytesWritten: Int64, totalBytesExp
ectedToWrite: Int64) {
        let progress = Float(totalBytesWritten) / Float(totalBytesExpectedT
oWrite)
        loadingProgressView.progress = progress
```

Giá trị của biến *progress* chính là tiến độ download, bạn không muốn dùng ProgressView mà dùng % thì cũng cứ thế mà dùng.

Vậy là xong.

Rất đơn giản đúng không? Vậy mình mất nhiều thời gian ở chỗ nào?

À là bởi vì khi mình dùng *URLSession*, mình truyền luôn *completionHandler* vào hàm downloadTask, kiểu như này:

func loadAssetFromServer() {

showLoading()

```
let session = URLSession(configuration: .default, delegate: self, d
 elegateQueue: .main)
          let path = "http://192.168.1.4:8080/"
          let url = URL(string: path + fileName)!
          let task = session.downloadTask(with: url) { url, response, error i
 n
             // Download complete, do something
         task_resume()
Tuy nhiên, nếu xử lý completion kiểu này thì các hàm trong delegate của bạn sẽ
KHÔNG CHẠY.
```

Vì Swift thấy có tận 2 chỗ đang xử lý completion, nên sẽ chỉ chạy một cái thôi. Và do Swift chọn completionHandler phía trên nên toàn bộ hàm trong delegate sẽ không

được gọi => **ProgressView** của mình không được update.

được. Hi vọng bài viết này của mình sẽ có ích cho các bạn đang muốn theo dõi tiến độ

Vậy mình bỏ completionHandler đi và handle completion trong hàm của delegate là

download file bằng *URLSession*. Swift URLSession Download Progress

```
All rights reserved
```

Các cách tốt nhất để tắt

Keyboard trong 1 View...

● 995 **■** 6 **●** 2 **♦** 5

Minh Nguyen

4 phút đọc

How use closure in your project? Hà Văn Đức 5 phút đọc

Bài viết liên quan

Bài viết khác từ Hà Tuấn Thịnh Throwing properties trong Swift

Sử dụng enum thay cho boolean trong Swift Hà Tuấn Thịnh

Làm quen với AR trong iOS: Mang mặt trăng vào phòng...

Kiến thức phỏng vấn iOS _

Phần 2: Grand Central...

Lê Văn Tuấn

4 phút đọc

5

Bạn không hiểu SSH?

Kiến thức phỏng vấn iOS _

Phần 2: Grand Central...

● 1147 ■ 3 ● 0 ◆ 3

Lê Văn Tuấn

Hà Tuấn Thịnh

20 phút đọc

6 phút đọc

Hà Tuấn Thịnh 2 phút đọc **◎**9 **■**0 **≥**0 **⇒**0 Bình luận

4 phút đọc

Hà Tuấn Thịnh 8 phút đọc

Đăng kí

Download on the

App Store

f 0 5 6

LIÊN KẾT

Bài viết

Câu hỏi

Videos

Thảo luận

Trạng thái hệ thống

Công cụ

ỨNG DỤNG DI ĐỘNG TÀI NGUYÊN DİCH AÑ Tổ chức 国 从经 国 Viblo Code GET IT ON Google Play Tags **CV** Viblo CV

⊋ Đăng nhập để bình luận

Tác giả

Đề xuất hệ thống

Machine Learning

Đăng nhập

Viblo CTF

Viblo Learning