Về bộ công cụ FastLane-Tools

***FastLane là một công cụ thư viện của bên thứ 3 với chức năng tích hợp nhiều công cụ giúp tiết kiệm thời gian và công sức trong việc phát triển ứng dụng di động trên nền tảng iOS.***

**1. Tổng quan về bộ công cụ phát triển FastLane:**

Home page : https://fastlane.tools/

GitHub : https://github.com/fastlane/fastlane

Cụ thể bộ công cụ hỗ trợ phát triển mà fastlane cung cấp tới các developers gồm có :

1. Deliver : Tải ảnh chụp màn hình, dự liệu và ứng dụng của bạn lên appstore chỉ với một dòng lệnh.

2. Snapshot : Tự động hoá việc chụp màn hình ứng dụng của bạn trên mọi thiết bị.

3. Frameit : Đưa ảnh chụp màn hình của bạn vào các khung device tương ứng một cách nhanh chóng.

4. Pem : Tự động tạo, làm mới cấu hình push notification của ứng dụng của bạn.

5. Sigh : Tiết kiệm thời gian cho công việc với provisioning cho ứng dụng của bạn.

6. Produce : Tạo các ứng dụng mới trên Itunes Connect và Developer Portal bằng cách sử dụng các dòng lệnh trong command.

7. Cert : Tự động tạo và duy trì (bảo trì) các Code Signing Certificates cho ứng dụng của bạn.

8. Scan : Cách dễ dàng nhất để test ứng dụng của bạn.

9. Gym : Hỗ trợ tối đa việc xây dựng ứng dụng của bạn.

10. Match : Dễ dàng đồng bộ các chứng chỉ (Certificate) và hồ sơ (Profile) của nhóm bạn sử dụng Git.

Ngoài ra còn có các công cụ hỗ trợ việc test khác như sau :

iOS: - Pilot : Công cụ hỗ trợ quản lý những tính năng quan trọng của Test Flight sử dụng terminal.

- Boarding : Ngay lập tức tạo ra và thêm một trang đăng ký đơn giản vào phiên bản ứng dụng TestFlight của bạn.

Bên cạnh đó cũng có hai công cụ hộ trợ dành cho môi trường android là : supply and screengrab. (Có thể tham khảo thêm trong trang chủ của FastLane).

Để cấu hình để sử dụng FastLane trên một chiệc máy chạy MacOS của bạn được giới thiệu đầy đủ trên FastLane Github. Bạn chỉ cần thực hiện theo các bước với các điều kiện cần thiết là có thể cài đặt và sử dụng FastLane một cách nhanh chóng. Tương tự như vậy với từng tools khác nhau của FastLane. Hãy đi từ trang chủ tới GitHub để thấy các bước cài đặt và sử dụng chúng.

Hiện FastLane đang dần trở lên phổ biến hơn với hơn 6k stars trên Github và ước tính thời gian tiết kiệm mang lại cho các Dev là 813195 hours.

Để biết rõ hơn về cách cài đặt và sử dụng những chức năng cơ bản nhất của FastLane bạn có thể tham khảo bài hướng dẫn theo đường dẫn sau :

https://www.raywenderlich.com/116065/fastlane-tutorial-getting-started

Tuy nhiên hướng dẫn này được thực hiện trên FastLane phiên bản 1.32.1 nên cần chú ý cài đặt đúng phiên bản trước khi làm theo.

**2. Cài đặt và sử dụng các tools nổi bật :**

***a. Cài đặt FastLane cho project của bạn :***

Đầu tiên là cài đặt FastLane vào thiết bị OSX của bạn, các bước cài đặt được ghi chi tiết và đầy đủ trên GitHub, xin được nhắc lại như sau :

Mở terminal của bạn và nhập các dòng lệnh sau :

*sudo gem install fastlane --verbose*

*xcode-select --install*

*gem cleanup*

Hoàn thành 3 bước trên FastLane đã được cài đặt trên thiết bị OSX của bạn. Nó cũng có thể được cài đặt trên thiết bị Linux với Ruby 2.0.0

Tiếp theo chúng ta khởi tạo và tích hợp FastLane vào Project mà chúng ta mong muốn bằng cách thực hiện theo bước sau:

Trỏ tới thư mục Project của bạn sử dụng lên cd, sau đó dùng lệnh fastlane init. Sau khi câu lệnh init được thực hiện xong bạn sẽ thấy trong thư mục dự án của bạn xuất hiện thêm một thư mục con là fastlane trong đó chứa 2 file textedit document là Fastfile và Appfile cùng với một thư mục con là actions. Như vậy là bạn đã tích hợp thành công FastLane tools vào Project của bạn. Bước tiếp theo sẽ là cài đặt, khởi tạo và sử dụng những công cụ con trong bộ công cụ của FastLane.

***b. Cài đặt và sử dụng một số công cụ con nổi bật :***

Đi từ trang chủ tới đường dẫn của các công cụ con của FastLane trên Github chúng ta có thể thấy được các bước cài đặt cũng như sử dụng chúng. Về cơ bản bước cài đặt và khởi tạo là tương tự nhau. Tuy nhiên cách sử dụng thì khác nhau tuỳ theo từng công cụ. Ví dụ :

***1. Cert:***

Đầu tiên hãy xem cách cài đặt và sử dụng công cụ con Cert(Certificate) như thế nào :

Mở terminal và cài đặt công cụ Cert cho thiết bị OSX của bạn bằng câu lệnh : *sudo gem install cert*

Như vậy là công cụ certificate đã được cài đặt. Giờ hãy dùng nó để tạo ra các certificate cần thiết cho Project của bạn. cd tới Project và dùng câu lệnh : *cert -u yourAppleID.*

Sau khi câu lệnh này được thực hiện bạn sẽ thấy các certificate được tạo ra và lưu lại trong thư mục dự án của bạn gồm có file .cer, .certSigningRequest và .p12. Những công việc được thực hiện là :

- Tạo ra một private key mới.

- Tạo ra một signing request mới.

- Tạo ra, tải về và cài đặt certificate vào Project của bạn.

- Thêm tất cả những files được khởi tạo cần thiết vào keychain của bạn.

Như vậy có thể thấy việc cài đặt và sử dụng công cụ con Cert là khá đơn giản và dễ dàng.

***2. Snapshot:***

Tiếp theo chúng ta hãy cùng xem cách cài đặt và sử dụng công cụ Snapshot(Công cụ tự động chụp màn hình trên nhiều thiết bị và upload nó lên trang thông tin ứng dụng trên appstore).

Để cài đặt snapshot chúng ta sử dụng lệnh : *sudo gem install snapshot*

Khởi tạo snapshot với câu lệnh : *snapshot init* tại thư mục dự án.

Sau khi khởi tạo thành công chúng ta sẽ thấy trong thư mục dự án xuất hiện thêm files *Snapfile*(textEditor) và *SnapshotHelper.swift*. Trong đó snapfile sẽ dùng để thiết lập chụp màn hình trên các thiết bị nào, ngôn ngữ nào và một vài thông số khác. SnapshotHelper.swift thì dùng để tích hợp và và sử dụng trong project của bạn, tuỳ biến chụp ở màn hình nào...

Giờ thì cần phải tích hợp và tuỳ biến việc chụp ảnh của màn hình nào vào Project của bạn. Các bước thực hiện như sau :

- Thêm file SnapshotHelper.swift và UI Test target của bạn (bước này mình chưa rõ lắm. Nếu chỉ đơn thuần là add nó vào Project thì hiện đang xảy ra lỗi.)

- Nếu Project của bạn sử dụng Objective C thì cần thêm vào bridging hearder phần import để có thể sử dụng được file.swift như sau : *#import "MYUITests-Swift.h"* . bridring header được đặt tên sau test target của bạn với phần hậu tố -Swift.h.

- Trong UI Test class của bạn, nhấn vào button record phía dưới bên trái và ghi lại tương tác của bạn.

- Để chụp một snapshot, gọi theo cách sau :

+ swift : *snapshot("01LoginScreen")*

+ Ob-C *:[snapshot snapshot:@"01LoginScreen" waitForLoadingIndicator:YES] ;*

- Thêm đoạn code sau vào phương thức setup() của bạn :

+ swift : *let app = XCUIApplication()*

*setupSnapshot(app)*

*app.launch()*

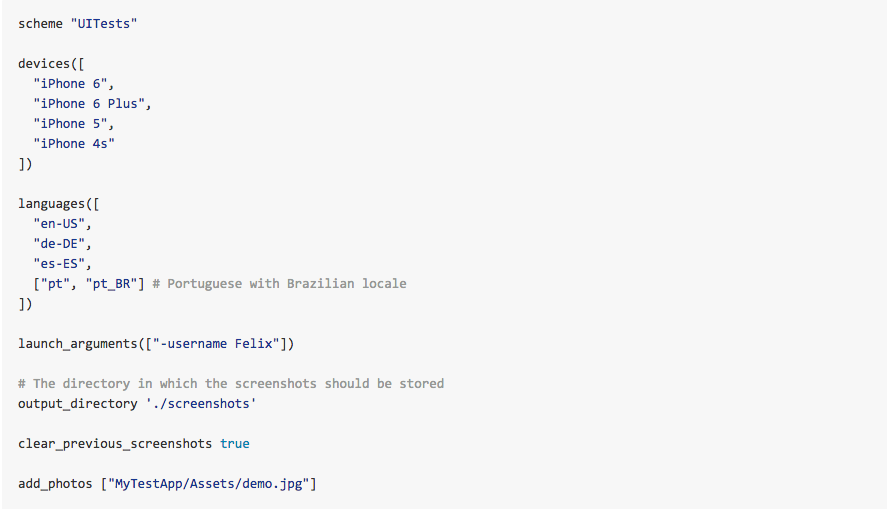
+ Ob-C : *XCUIApplication \*app =[[XCUIApplication alloc] init] ;*

*[Snapshot setupSnapshot:app];*

*[app launch];*

Để sử dụng Snapshot chúng ta dùng từ khoá snapshot, có rất nhiều câu lệnh với chức năng khác nhau chúng ta có thể tham khảo trên Github hoặc dùng snapshot --help để liệt kê chúng.

Có thể tham khảo một Snapfile sau :



***3. Gym:***

Tiếp theo chúng ta sẽ đến với công cụ con quan trọng khác đó là gym. Công cụ tối ưu hoá việc build app.

Dưới đây là bảng thông tin ưu điểm của công cụ con này.

|  |  |
| --- | --- |
| rocket: | gym builds 30% faster than other build tools like [shenzhen](https://github.com/nomad/shenzhen) |
| checkered_flag: | Beautiful inline build output |
| book: | Helps you resolve common build errors like code signing issues |
| mountain_cableway: | Sensible defaults: Automatically detect the project, its schemes and more |
| link: | Works perfectly with [fastlane](https://fastlane.tools/) and other tools |
| package: | Automatically generates an ipa and a compressed dSYM file |
| bullettrain_side: | Don't remember any complicated build commands, just gym |
| wrench: | Easy and dynamic configuration using parameters and environment variables |
| floppy_disk: | Store common build settings in a Gymfile |
| outbox_tray: | All archives are stored and accessible in the Xcode Organizer |
| computer: | Supports both iOS and Mac applications |

Cũng giống như những công cụ con khác chúng ta cài đặt gym bằng câu lệnh : *sudo gem install gym* .

Sau khi gym được cài đặt xong bạn đi tới thư mục dự án dùng lệnh : *gym init* để khởi tạo gym và gymfile. gymfile được tạo ra trong thư mục fastlane và bạn cần sửa nó lại như sau :

*scheme " SimiCartPluginFW "* // trong trường hợp này mình đang demo với bản cloud version 3.0.

*sdk "iphoneos"* // trong hướng dẫn của nhà cung cấp là iphoneos9.0 nhưng khi build xảy ra lỗi(chưa rõ nguyên nhân).

*output\_directory "./build"* // lưu file ipa được tạo ra vào thư mục này.

*output\_name "MyAppName"* // tên cho file ipa được tạo ra.

Tiếp theo bạn chạy câu lệnh : *gym*

Trong trường hợp của mình thì xảy ra lỗi và gym đưa ra giải pháp là sử dụng use\_legacy\_build\_ api. Dùng lệnh : *gym --use\_legacy\_build\_api* build lại và success. File ipa xuất hiện.

Bạn có thể tham khảo các lệnh với các chức năng khác nhau với gym trong phần usage được viết trên Github của nhà cung cấp.

***4. Produce, Deliver, Pem :***

Ngoài ra các công cụ khá đơn giản là produce(Tự động tạo app trên itunesConnect) ,deliver (Tải ảnh chụp màn hình, dự liệu và ứng dụng của bạn lên appstore chỉ với một dòng lệnh) và pem(Hỗ trợ tạo các file pem phục vụ cho PushNotification) mình đã làm theo hướng dẫn và thành công.

Đó là toàn bộ hướng dẫn cài đặt và sử dụng công cụ con snapshot của FastLane. Tuy nhiên còn một vài điểm chưa rõ nên mình vẫn chưa demo được, hai trọng những công cụ con quan trọng là snapshot và gym thêm thời gian tìm hiểu, chạy thử... để hoàn thành demo hoặc cần sự trợ giúp.

Trên đây là trình bày về hai trong số 12 công cụ con của FastLane. Một công cụ đơn giản nhất và một công cụ có thể là phức tạp nhất sau khi tham khảo từ tài liệu của nhà cung cấp. Các công cụ khác đều cài đặt và sử dụng tương tự như vậy. Hoặc nếu có khác thì cũng được trình bày rõ ràng từng bước trong hướng dẫn từ nhà cung cấp. Do thời gian có hạn cũng như hai khó khăn gặp phải về phiên bản cũng như các file nguồn viết bằng Ruby nên việc tìm hiểu còn hạn chế. Mình sẽ tiếp tục dành thời gian tìm hiểu và quan trọng nhất là demo thành công trong tương lai.