fastlane

Introduction to fastlane by Fabric

Prepared for CocoaHeads

March, 2017

Fastlane

01 Fastlane Introduction

02 Pros, Cons, and Comparisons

03 Fastlane Tool

04 Snafus and Good to Knows

1.0

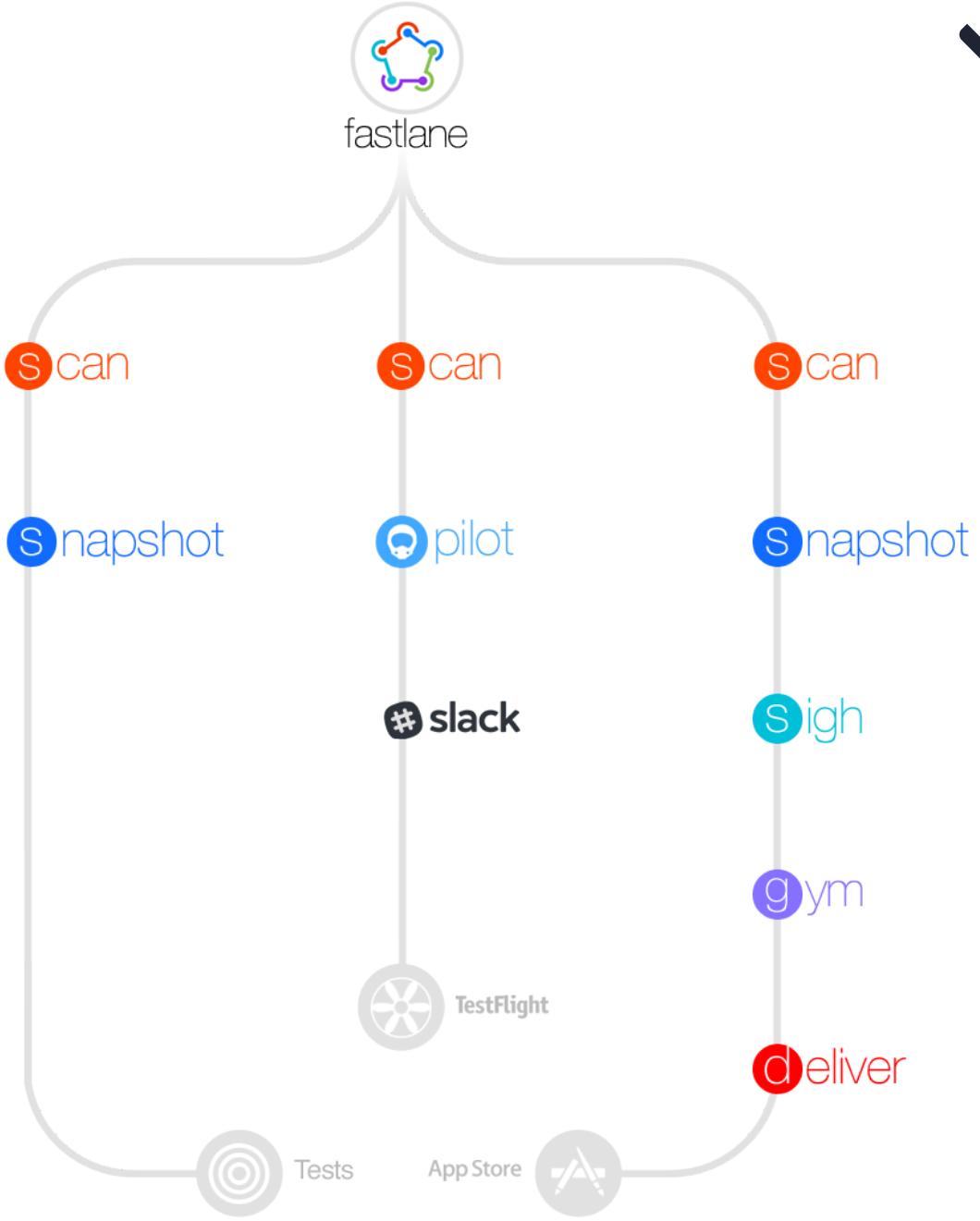
Fastlane Introduction

Fastlane By Fabric

- Fastlane is a open-source Ruby tool used to automate many parts of mobile development and app submission.
- Fastlane is a tool produced by Fabric, which is now a part of Google. (~ Jan 18th 2017)
- Fabric has ~20 sub-tools, and integrations into many apps and processes, like Slack and Crashlytics

Fastlane Can Automate All The Things!

- App Unit Tests
- App Screenshots
- App Building
- App Signing
- App/Metadata Delivery
- App Release



2.0

Pros, Cons, and Comparisons

Things to look for as a good candidate to use with Fastlane.

Pros

- Easy to use once set-up.
- Good for build servers/large teams to share credentials and workflow.
- Works with a large variety of situations.

Cons

- Larger initial set-up.
- The more complex the project, the more hiccups you may encounter.
- Documentation is slightly lacking.

Note: As with any tool, mileage will vary greatly project to project.

Comparing Fastlane with the manual way

Fastlane Way

- Run "Fastlane Match"*
- Provide match password once
- Run "Fastlane LaneGymAndPilot"

Old Way

- Export .p12 files from creator's Mac.
- Import to recipient Mac.
- Download provisions from iTunes.
- Build Project in Xcode.
- Submit to Apple From Organizer.

*Note: This is after set-up of the match repo

3.0

Fastlane Tools

There are a LOT of them

Fastlane – Snap Shot (iOS) + Screen Grab (Android)

- “Automatically” takes screenshots of your app!
- Requires UI Tests (iOS) to be Setup…
- But luckily they’re easier than ever!

Fastlane – Simple UITest Integration Examples

```
    func testExample() {

        let app = XCUIApplication()
        setupSnapshot(app)

        snapshot("initial")
        app.buttons["Navigate To Screen 1"].tap()
        snapshot("scene 2")
        app.buttons["Navigate To Screen 2"].tap()
        snapshot("scene 3")
        app.buttons["Navigate To Screen 3"].tap()
        snapshot("scene 4")
        app.buttons["Navigate To Screen 4"].tap()
        snapshot("scene 5")
        app.buttons["Navigate To Screen 5"].tap()


        // Use recording to get started writing UI tests.
        // Use XCTAssert and related functions to verify your tests produce the correct results.
    }

}
```

Fastlane – Match

- Easier Distribution/Creation of Provisions and Certs
- Secure and Access-Limiting; One person creates it, everyone else can use them.
- Never have to pass around p12 Files again.

Fastlane – Gym (iOS) + Gradle (Android)

- One time set-up.
- Not generally used Solo
- Pass the build on to the next tool to automate delivery of your app.

Fastlane – Pilot/Deliver + Crashlytics

- After building with Gym, you can push to your “favorite” distribution systems.
- Beta (Crashlytics)
- Deliver -> Apple TestFlight
- (Supply is like Deliver, but for Android)

Fastlane - Lane Examples

```
lane :devBuild do
    snapshot(
        launch_arguments: [“-snapEmail tester123@gmail.com -snapPass password123”],
        #reinstall_app: true,
        #erase_simulator: true,
        devices: ["iPhone 5s”, "iPhone 6s"],
        languages: ["en-US”],
        app_identifier: “com.company.appname”,
        scheme: “scheme1”
    )
    match
    gym(
        scheme: 'SuperMegaApp9001',
        configuration: "Debug" # or "Release"
    )
    crashlytics(
            api_token: 'token...',
            build_secret: 'buildSecretFromCrashlytics',
            emails: nil,
            groups: ['group1Name'], #(group names from Crashlytics)
            notes: 'Distributed with fastlane',
            notifications: true
    )
end

lane :uatBuild do
    gym(
        scheme: 'SuperMegaApp9001',
    )
    pilot(
        username: 'username@emailForYOURItunes.com',
        skip_waiting_for_build_processing: true,
        skip_submission: true  # not auto submit to apple
    )
end
```

Other tools/actions usable with Fastlane

Increment version number (options for patch vs minor vs major)

Increment build number

Register devices

CocoaPods

Reset simulator contents

Update fastlane (self updating… hmmmmm)

Slack Notifications

and many MANY more listed at https://docs.fastlane.tools/actions/

Demo Time

DEMO TIME

4.0

Snafus and Good to Knows

Snafus

- Fastlane uses Ruby to communicate to other apps and API's, so keeping Ruby up to date is necessary when submitting to the App store.
- Similar to the "Fix it" button in Xcode, you can break quite a bit if you're not careful. (ALWAYS back up your project before running certain Fastlane options)

Some Misc Uses

- In your fastlane file you can specify a certain Xcode version like so : (there are other ways, but they generally require Sudo)
- **ENV['DEVELOPER_DIR'] = '/Applications/Xcode-Beta6.3.app/Contents/Developer'**
- Jenkin Integration is fairly painless!
- Private Lanes

Questions and Thank You