NICOLÁS GARCÍA (@NICOONGUITAR)

UI TESTING : DEALING WITH PUSH NOTIFICATIONS

INDEX

- Multi-app Testing
- Sending a push notification through a UITest
- Handling user interactions on a notification in a UITest

INDEX

- Multi-app Testing
- Sending a push notification through a UITest
- Handling user interactions on a notification in a UITest

INDEX

- Multi-app Testing
- Sending a push notification through a UITest
- Handling user interactions on a notification in a UITest

INDEX

- Multi-app Testing
- Sending a push notification through a UITest
- Handling user interactions on a notification in a UITest

XCODE 9: UI TESTS CAN TARGET MULTIPLE APPLICATIONS

- Apps can interact with other apps (ex: Springboard)

```
let springboard = XCUIApplication(bundleIdentifier: “com.apple.springboard”)
```

- Accessibility allows us to interact with a received notification

```
springboard.otherElements["APPSNAME, now, Hello World”].tap()
```

```
springboard.otherElements["NotificationShortLookView"].tap()
```

- WWDC 2017 - Session 409 “What's New in Testing”
- System apps bundle identifiers list

DEMO

- Interacting with the Control Center by using a UITest

NWPUSHER: THE HERO

- Author: Noodlewerk
- GitHub NWPusher
- Sends Push Notifications (Sandbox and Production environments).
- Allows us to define the notification's payload.
- Original idea by Jörn Schoppe

SETTING THINGS UP

- Generate a .p12 file from the Keychain APNs app’s certificate
- Add the .p12 file on app’s UITests target
- Install NWPusher on app’s UITests target
- Write those UITests!

INSTALLING NWPUSHER

- Install in app’s UITests target
- Podfile example (also available for Carthage…)

```
target 'TestingPushNotifications' do
   use_frameworks!

   target 'TestingPushNotificationsUITests' do
       inherit! :search_paths
       pod 'NWPusher', '~> 0.7.0'
    end
end
```

WRITING THE UITEST

- Getting the Device token
- Sending the Push Notification through NWPusher
- Using waitForExistence API
- Interacting with the received notification

GETTING THE PUSH NOTIFICATION'S DEVICE TOKEN

- Allow remote notifications permission in app
- Dealing with Location Services permission system alert
- How to make the Device Token visible to the UITest? 🤔

GETTING THE PUSH NOTIFICATION'S DEVICE TOKEN

The current solution:

- Use of XCUIApplication's launchArguments
- Add a UILabel on screen with the device token as its text
- Device token is now visible to Accessibility framework

SENDING A PUSH NOTIFICATION FROM THE UITEST

- Requires the .p12 file and its password
- Requires a customizable payload

```
    func triggerPushNotification(payload: String, deviceToken: String) {
        let uiTestBundle = Bundle(for: TestingPushNotificationsUITests.self)
        guard let url = uiTestBundle.url(forResource: "apns_dev.p12", withExtension: nil) else { return }
        
        do {
            let data = try Data(contentsOf: url)
            let pusher = try NWPusher.connect(withPKCS12Data: data, password: "cocoaheadsnantes", environment: .sandbox)
            try pusher.pushPayload(payload, token: deviceToken, identifier: UInt(arc4random_uniform(UInt32(999))))
        } catch {
            print(error)
        }
    }
```

DEMO

- Sending a push notification from a UITest
- Handling the received push notification inside the UITest

UNNOTIFICATIONACTION

- Notification with actions
- Using .swipeDown() on the notification’s XCUIElement
- Testing user interactions on the received notification
- UNTextInputNotificationAction: answering the message

DEMO

- User interactions on a received notification with actions in a UITest

SUMMARY

- UI Tests can target multiple apps 🚀
- Test notifications with actions 🚀
- Test analytics code for notification's user interactions 🚀
- Only works with a real device 👨🏻‍💻
- UITests stability 🤕
- Dealing with APNs 🤕

THANK YOU!

@NicoonGuitar on Twitter
iOS @VelcoBike 🚵

Thanks to…

* Jörn Schoppe (for the cool idea!).
* CocoaHeads Nantes.