UI SNAPSHOT TESTS

Christian Menschel @cmenschel

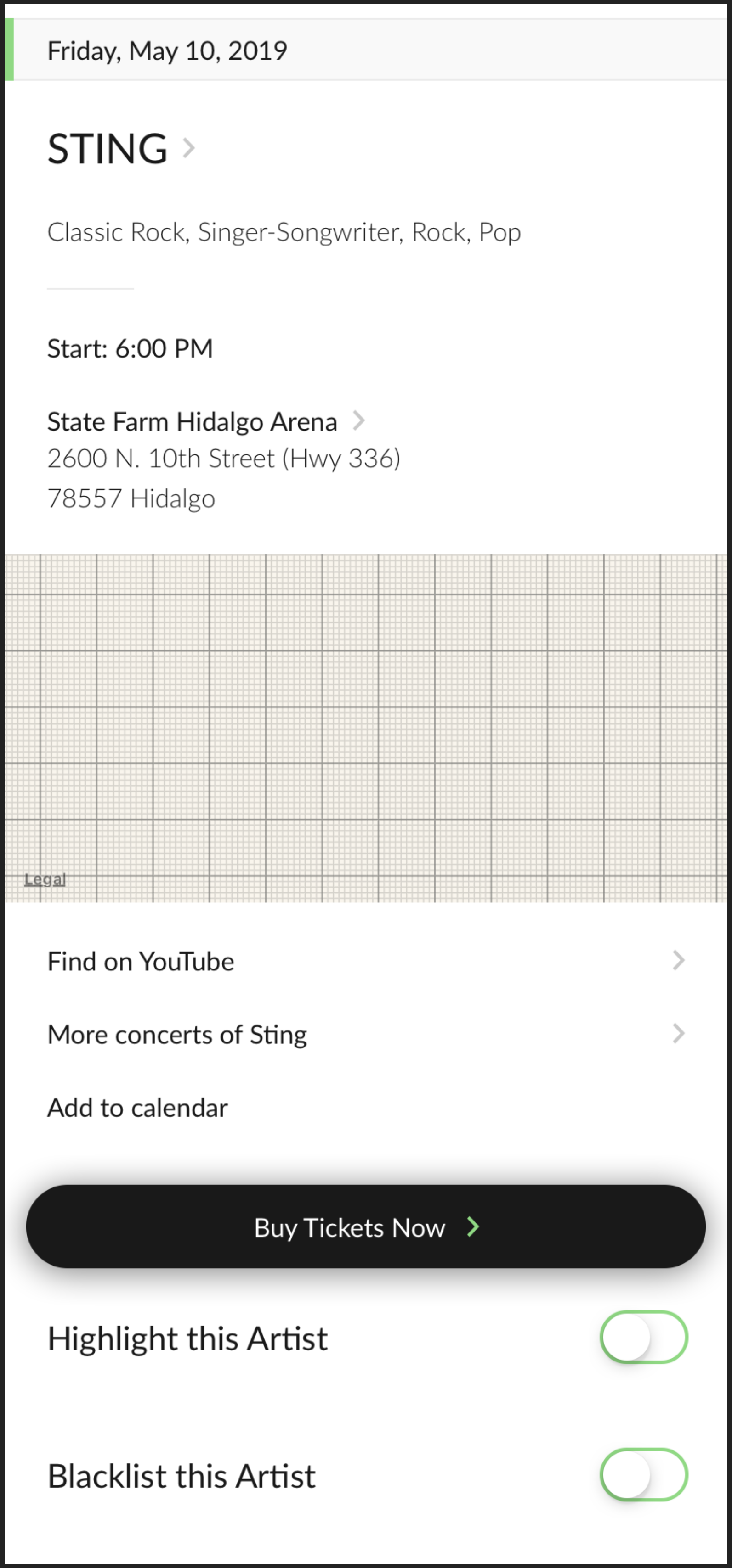
25.04.2019

UI SNAPSHOT TESTS

- iOSSnapshotTestCase (previously named FBSnapshotTestCase)
- Introduced by Facebook - now maintained by Uber
- https://github.com/uber/ios-snapshot-test-case

UI SNAPSHOT TESTS

- Works as XCTest
- Tests your UI (pixel based)
- Does not replace logic test
  - It’s an addition


Friday, May 10, 2019

STING

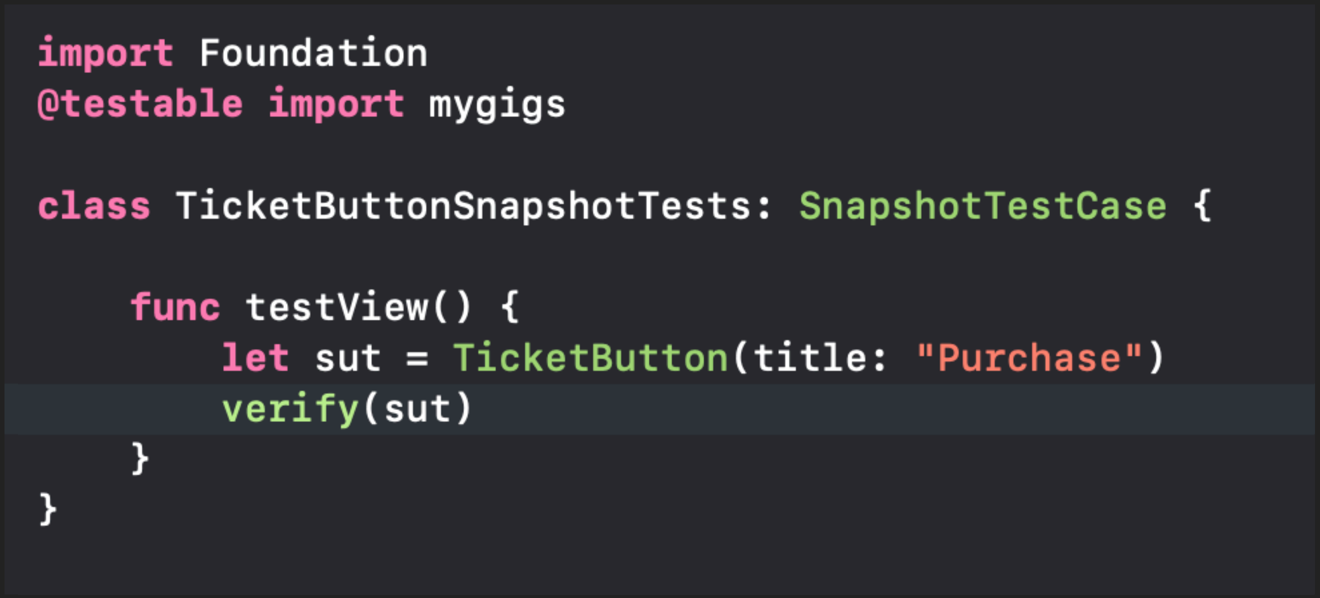
Classic Rock, Singer-Songwriter, Rock, Pop

Start: 6:00 PM

State Farm Hidalgo Arena
2600 N. 10th Street (Hwy 336)
78557 Hidalgo

UI SNAPSHOT TESTS

- Create snapshot once as your expected

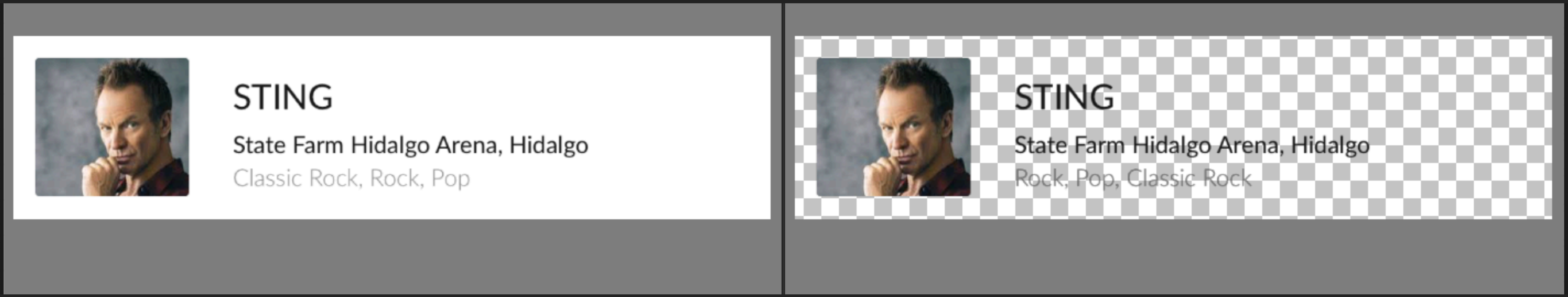


```swift
import Foundation
@testable import mygigs

class TicketButtonSnapshotTests: SnapshotTestCase {

    func testView() {
        let sut = TicketButton(title: "Purchase")
        verify(sut)
    }
}
```

- Each test run renders your current view to compare against the snapshot image
- Support for multiple devices (iPhone 8 / X / XS Max)

UI SNAPSHOT TESTS

- Fails if someone breaks the UI by …
  - Changing colors, fonts
  - Updating content
  - Changing layout (Autolayout constraints)
  - …


STING
State Farm Hidalgo Arena, Hidalgo
Classic Rock, Rock, Pop

STING
State Farm Hidalgo Arena, Hidalgo
Rock, Pop, Classic Rock

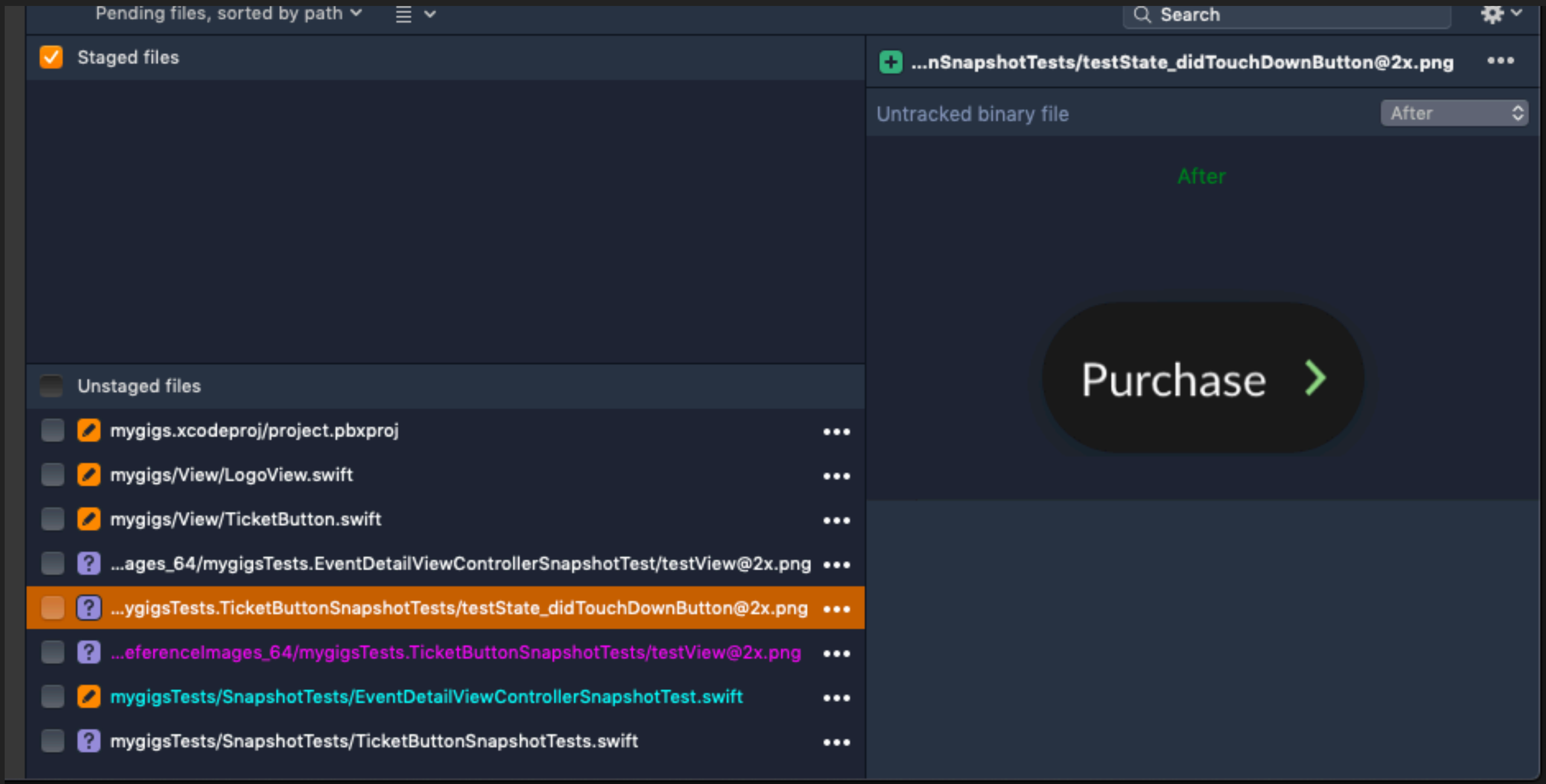
TEST DRIVEN UI DEVELOPMENT (TDUID👩🏼‍💻)

TDUID

- Use recordMode while developing

```
// Then
verify(sut, createNewSnapshot: true)
```

- Visualized code changes (after running the test)



Pending files, sorted by path

Staged files

Unstaged files

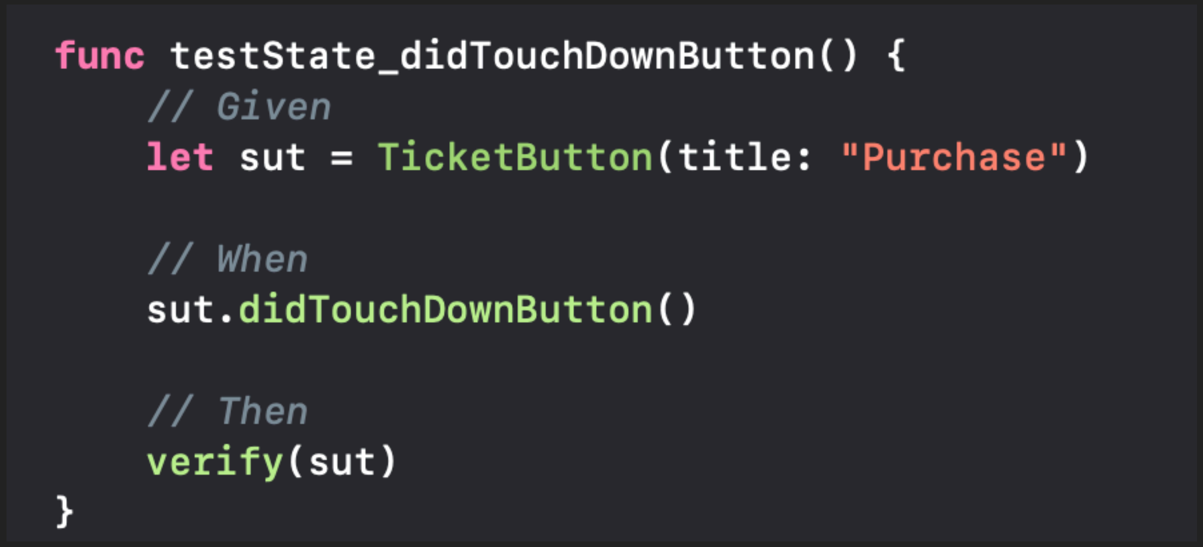
- mygigs.xcodeproj/project.pbxproj
- mygigs/View/LogoView.swift
- mygigs/View/TicketButton.swift
- ...ages_64/mygigsTests.EventDetailViewControllerSnapshotTest/testView@2x.png
- ...ygigsTests.TicketButtonSnapshotTests/testState_didTouchDownButton@2x.png
- ...eferenceImages_64/mygigsTests.TicketButtonSnapshotTests/testView@2x.png
- mygigsTests/SnapshotTests/EventDetailViewControllerSnapshotTest.swift
- mygigsTests/SnapshotTests/TicketButtonSnapshotTests.swift

...nSnapshotTests/testState_didTouchDownButton@2x.png

Untracked binary file

TDUID

- Saves you to …
    - … Launch the iOS Sim all the time
    - … Going through your navigation stack



```swift
func testState_didTouchDownButton() {
    // Given
    let sut = TicketButton(title: "Purchase")

    // When
    sut.didTouchDownButton()

    // Then
    verify(sut)
}
```

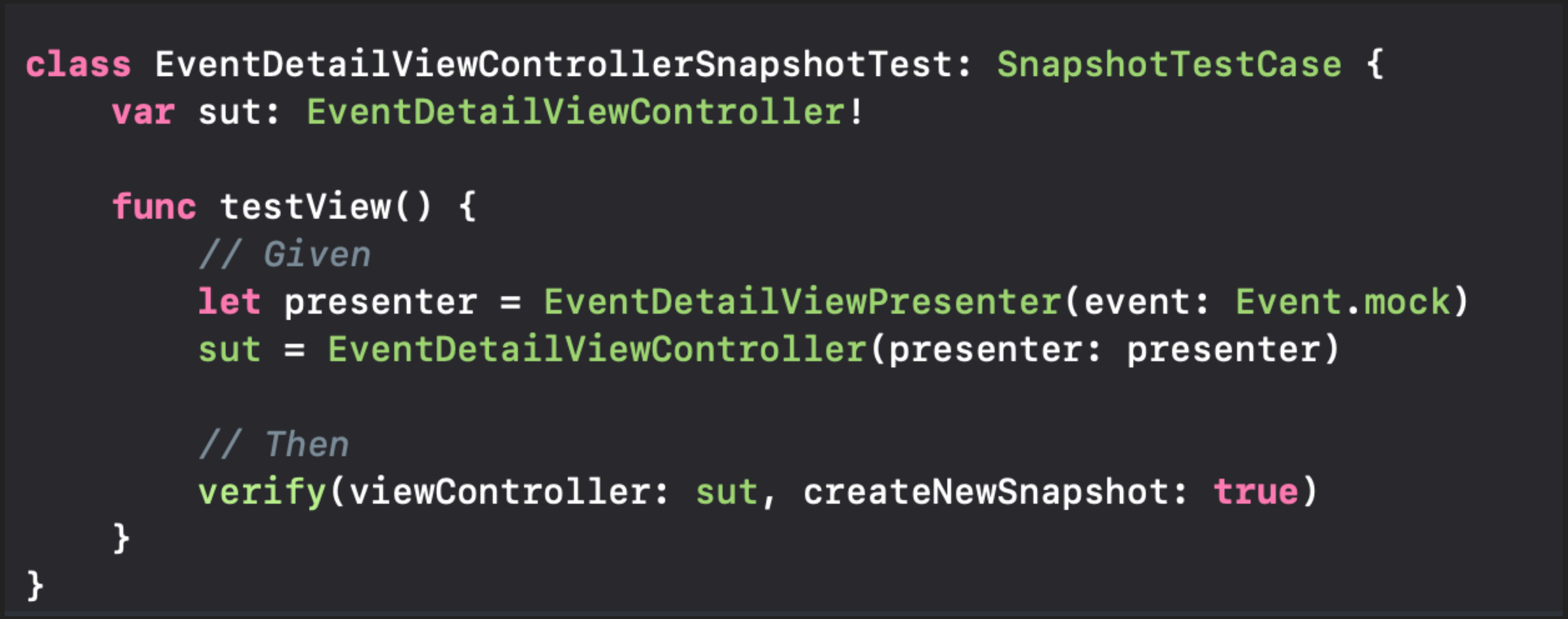
TIPS & TRICKS

TIPS & TRICKS

- Use always the same mock data (careful with dates)
- Test single views (UITableViewCell) instead the whole UITableView
- Like with every Unit Test: Keep the test simple
- Don’t mix logic and UI tests
- Use Kaleidoscope or other visual diff tools

TIPS & TRICKS

- Avoid testing UIViewController
  - If needed use a test UIWindow and set the rootViewController
  - Test UIViewController for integration test



```swift
class EventDetailViewControllerSnapshotTest: SnapshotTestCase {
    var sut: EventDetailViewController!

    func testView() {
        // Given
        let presenter = EventDetailViewPresenter(event: Event.mock)
        sut = EventDetailViewController(presenter: presenter)

        // Then
        verify(viewController: sut, createNewSnapshot: true)
    }
}
```

PRO & CONS

PROS

- See immediately if the expected layout has changed
- Easy and fast to write
- Test multiple UI states (cover all edge cases)
- Allows Test Driven Development

PROS

- App’s UI state with all edge cases
- “Agreement” between developers, PO & designer
- Pull Request:
  - See visually what the PR does
  - Include your designers into the review process

CONS

- Asynchronous testing is hard & flaky (like with all Unit tests)
- Changes in UIKit (i.e. font rendering) requires recording all the snapshots again
- Hard to see the diff if only few pixels changed
- Animations are hard to test (try to disable them)

LINKS

- https://github.com/uber/ios-snapshot-test-case
- Helper: https://github.com/tapwork/FBSnapshotTestCase-Subclass/