UI Testing

mit Xcode 7

Übersicht

- Was ist UI-Testing
- Tools
- Testing best practices

UI testing

- spezielle Form von Akzeptanztests
- User features
- „Vogelperspektive“ auf die Anwendung abseits technischer Details
- Test über die UI-Schicht (der selbe Weg, den auch der Endbenutzer geht)

Xcode 7 UI Testing

- direkte Integration in Xcode und XCTest-Infrastuktur
- Swift und Objective-C
- aufbauend auf der Accessibility-API
- iOS 9 und OS X 10.11

Demo

Testrunner

- eigenes Xcode Target
- Xcode-Templates für UI-Testing
- Zugriff auf UI über Accessibility-API des Host-OS

API

- XCUIApplication
- XCUIElement
- XCUIElementQuery

XCUIApplication

- Proxy für die getestete App
- separater Prozess
- jeder Test startet die App erneut und beendet vorherige Instanz
- kein State aus vorherigen Tests

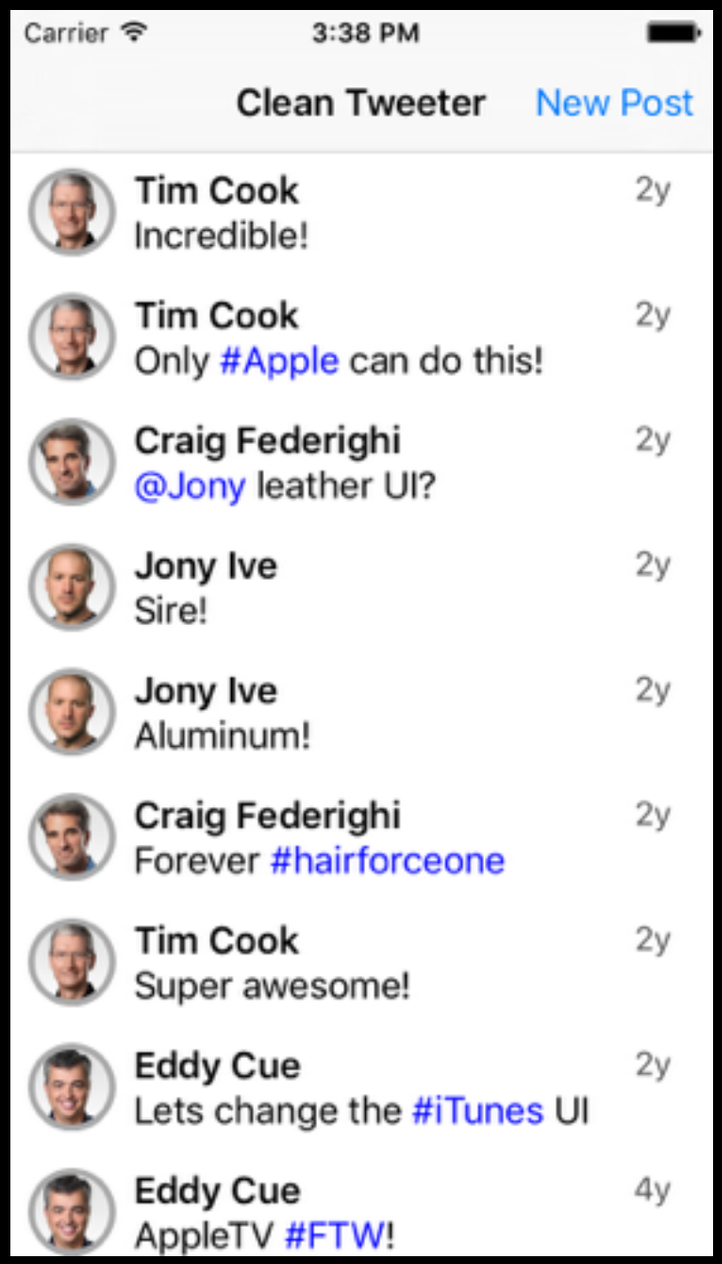
➡ einfacher Test-Setup

XCUIElement

- Proxy für UI-Elemente der App
- Typ
  - Button, Window, Cell etc.
- Identifier
  - von Accessibility bereitgestellte String-Metadaten des UI (accessibilityLabel, accessibilityTitle, accessibilityValue etc.)
- Elemente werden aus einer Kombination von Typ und Identifier gefunden (z.B. Button mit Titel „Abbrechen“)

UI ist eine hierarchische Baumstruktur

- Application
  - Navbar
    - „Clean Tweeter“
    - „New Post“
  - View
    - Table
      - Cell
        - „Tim Cook“
      - Cell
        - „Jony Ive“
      - Cell
        - „Craig Federighi“

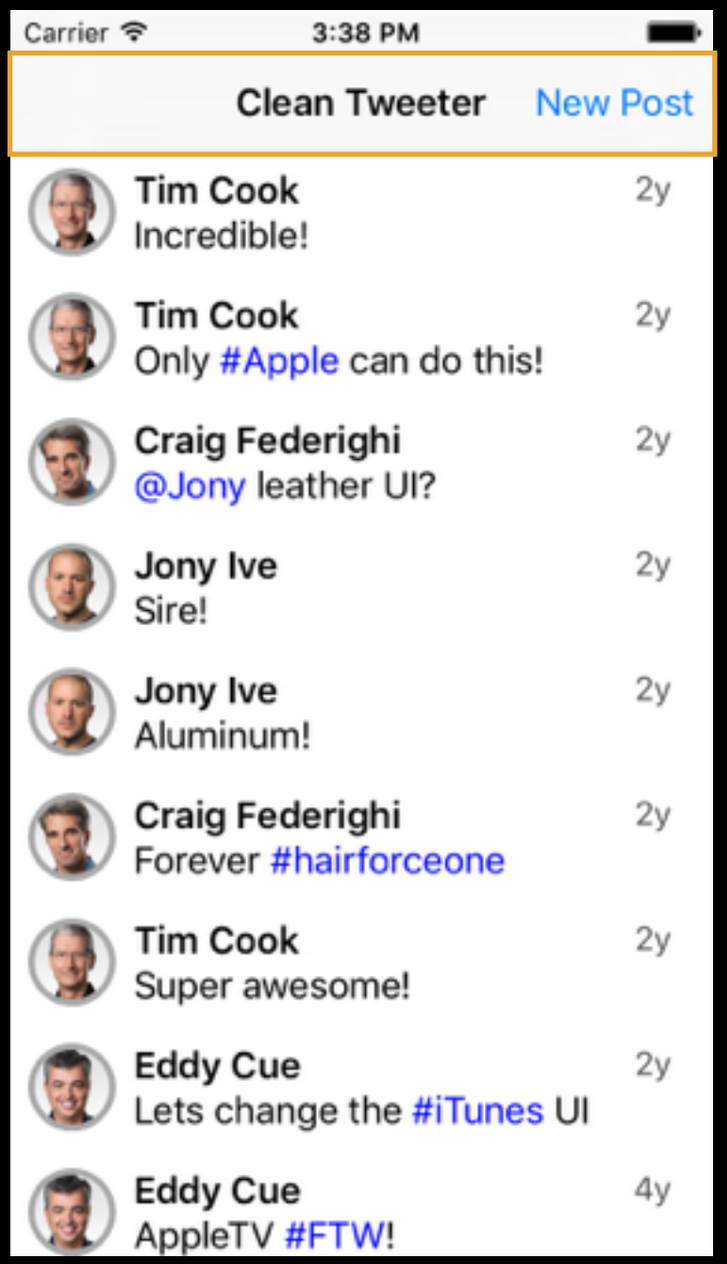


UI ist eine hierarchische Baumstruktur

- Application
  - Navbar
    - „Clean Tweeter“
    - „New Post“
  - View
    - Table
      - Cell
        - „Tim Cook“
      - Cell
        - „Jony Ive“
      - Cell
        - „Craig Federighi“

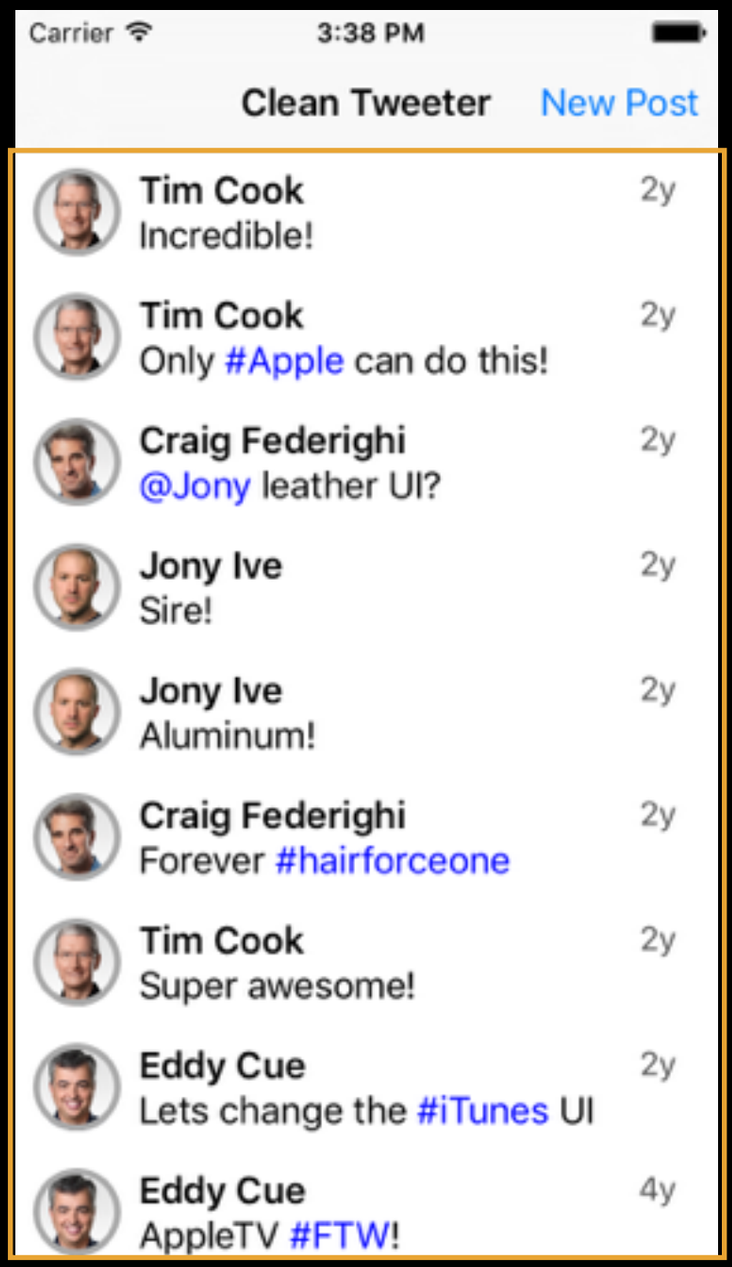
UI ist eine hierarchische Baumstruktur

Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



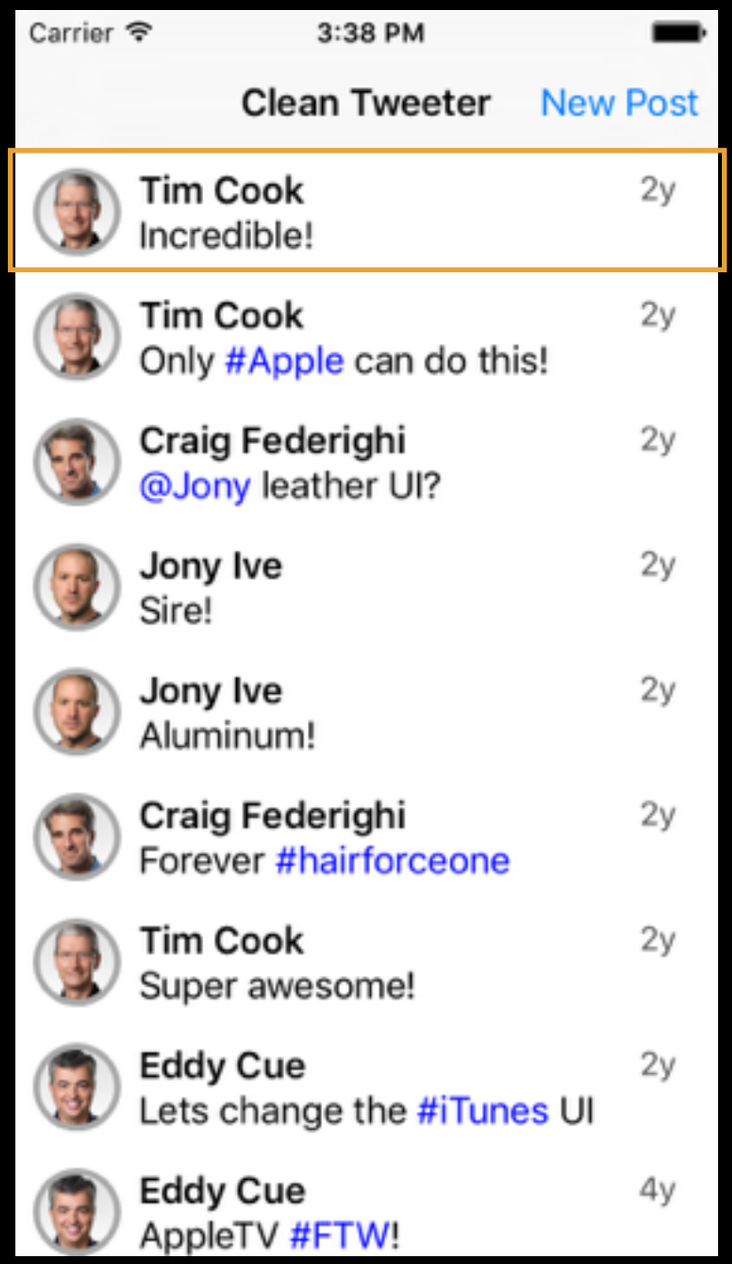
UI ist eine hierarchische Baumstruktur

Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



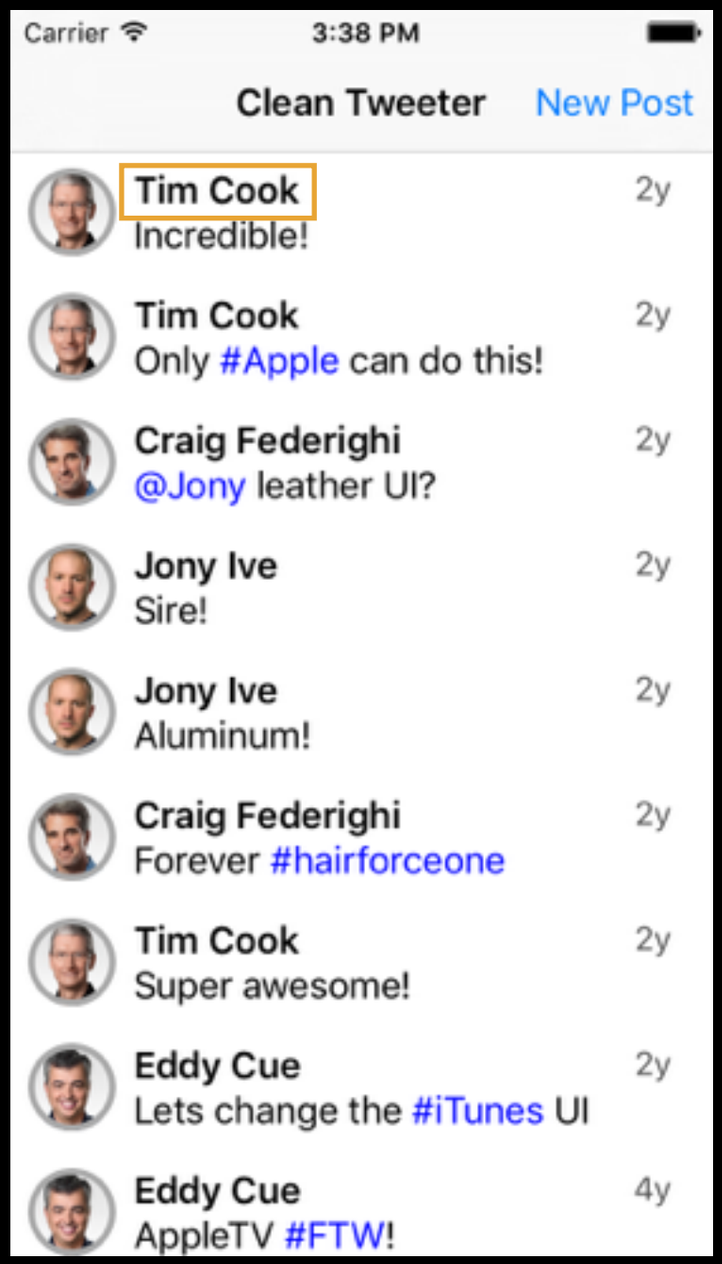
UI ist eine hierarchische Baumstruktur

Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



UI ist eine hierarchische Baumstruktur

Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



XCUIElement

- XCUIElemente werden mittels eines query gefunden
- Elemente müssen eindeutig über ihr accessibility-API identifizierbar sein
  - ➡ gleicher Typ und gleicher Identifier resultiert in Testfehlschlag
- Ausnahme ist die exists property von XCUIElement
  - ➡ für Kontrollfluss in den Tests oder Nichtexistenz eines UI-Elementes bsp. nach dismissal animation

XCUIElement

- Event-Erzeugung erfolgt durch die Elemente
- auf unterster Ebene über den gleichen Weg wie durch die UI vom Benutzer

```
button.click() // OS X
button.tap() // iOS

textfield.typeText(„Hello World“) // OS X & iOS
```

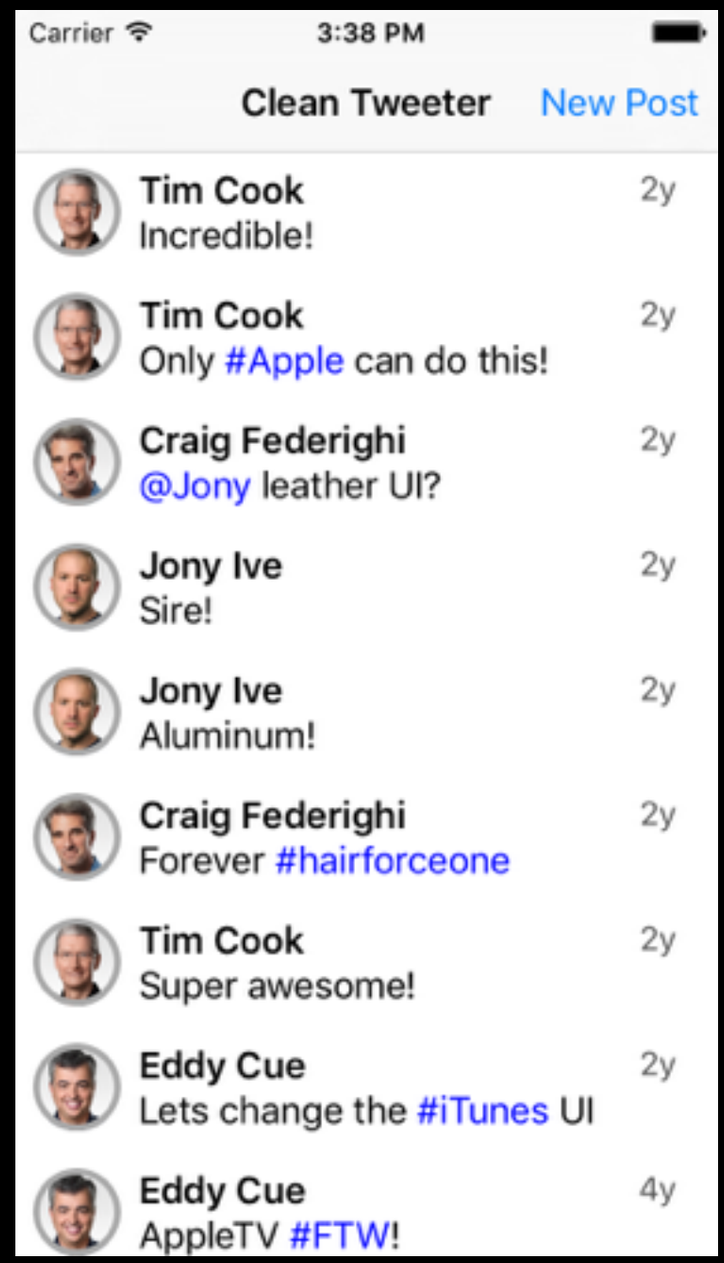
XCUIElementQuery

- zum Auffinden der Elemente in der UI
  - Anzahl der matches: `count`
  - Nach identifier mittels subscripting
  - Nach index: `elementsBoundByIndex()`

UIElementQuery

Descendants

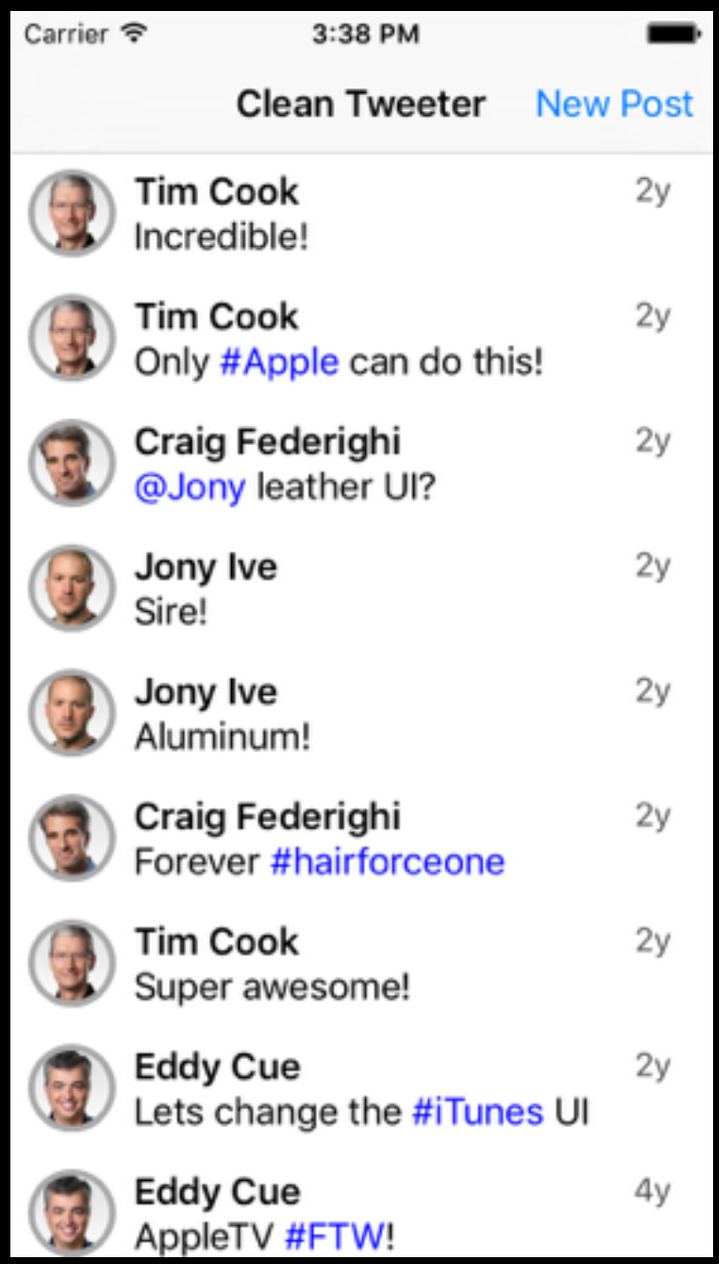
Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



UIElementQuery

Children

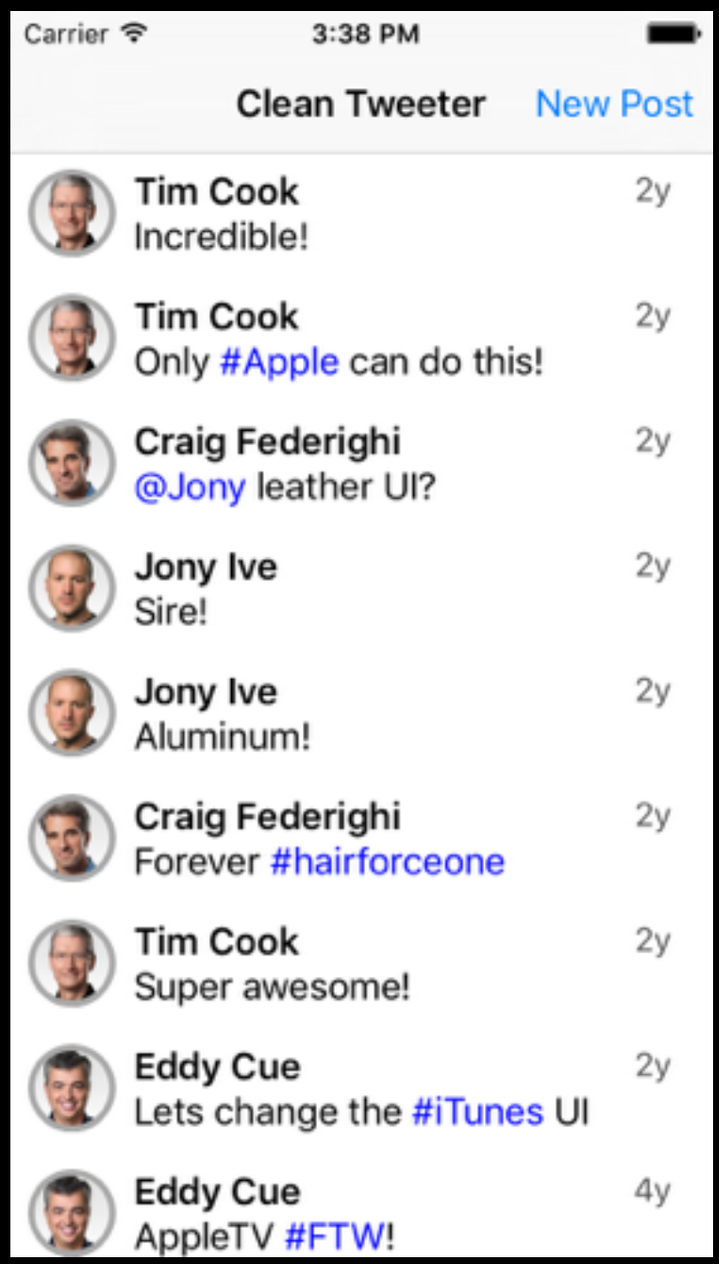
Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



UIElementQuery

Containment

Application
- Navbar
  - „Clean Tweeter“
  - „New Post“
- View
  - Table
    - Cell
      - „Tim Cook“
    - Cell
      - „Jony Ive“
    - Cell
      - „Craig Federighi“



XCUIElementQuery

- Filterung nach Typ
  - Button, Cell, Menu etc
- Filterung nach Identifiers
  - accessibility label, title etc
- Filterung mittels predicate
  - value, beginsWith etc.

XCUIElementQuery

```
let allButtons = app.descendantsMatchingType(.Button)

let allCellsInTable = table.descendantsMatchingType(.Cell)

let allMenuItemsInMenu = menu.descendantsMatchingType(.MenuItem)
```

XCUIElementQuery

Convenience API für häufig genutzte queries

```
let allButtons = app.buttons

let allCellsInTable = table.cells

let allMenuItemsInMenu = menu.menuItems
```

XCUIElementQuery

childrenMatchingType

```
table.childrenMatchingType(.Cell)
```

```
navBar.childrenMatchingType(.Button)
```

XCUIElementQuery

childrenMatchingType

```
app.tables.containingType(.Cell, identifier: "Tim Cook“)
```

```
app.tables.containingPredicate(NSPredicate())
```

XCUIElementQuery

- queries können hintereinander geschaltet werden

```
→ app.tables.staticTexts
```

XCUIElementQuery

- Subscripting

```
app.tables.staticTexts["Tim Cook"]
```

- Index

```
app.tables.elementBoundByIndex(42)
```

- Zugriff auf einziges Element

```
app.navigationBars.element
```

XCUIElementQuery

- Queries werden on-demand ausgeführt
- Ausführung erst bei Event-Synthetisierung oder property-Zugriff auf ein Element
- neue Evaluation bei erneutem Zugriff oder UI-Verändung
- somit sind Anpassung an ein geändertes UI möglich, z.B. nach Animationen

Accessibility

Demo

Lokalisierung und Adaptivity

Fallstricke beim Testen vermeiden

Demo

Accessibility und Testing

- Verbessertes Benutzererlebnis für Menschen mit Behinderungen
- Kopplung der Tests an lokalisierte UI-Strings vermeiden
- Custom Views sollten Accessibility Protokolle unterstützen

UIAccessibility

The `UIAccessibilityIdentification` protocol is used to associate a unique identifier with elements in your user interface. You can use the identifiers you define in UI Automation scripts because the value of `accessibilityIdentifier` corresponds to the return value of the `name` method of `UIAElement`.

https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIAccessibilityIdentification_Protocol/

Given that accessibilityLabel is an outwardly-facing string that is actually used by accessibility screen readers (and should be localized to the device user's language), Apple now provides an alternate property (iOS 5+) that is specifically intended for UI Automation purposes

someone on the internet

http://stackoverflow.com/questions/21152716/whats-the-difference-between-setaccessibilitylabel-and-accessibilityidentifier

Test-Hygiene

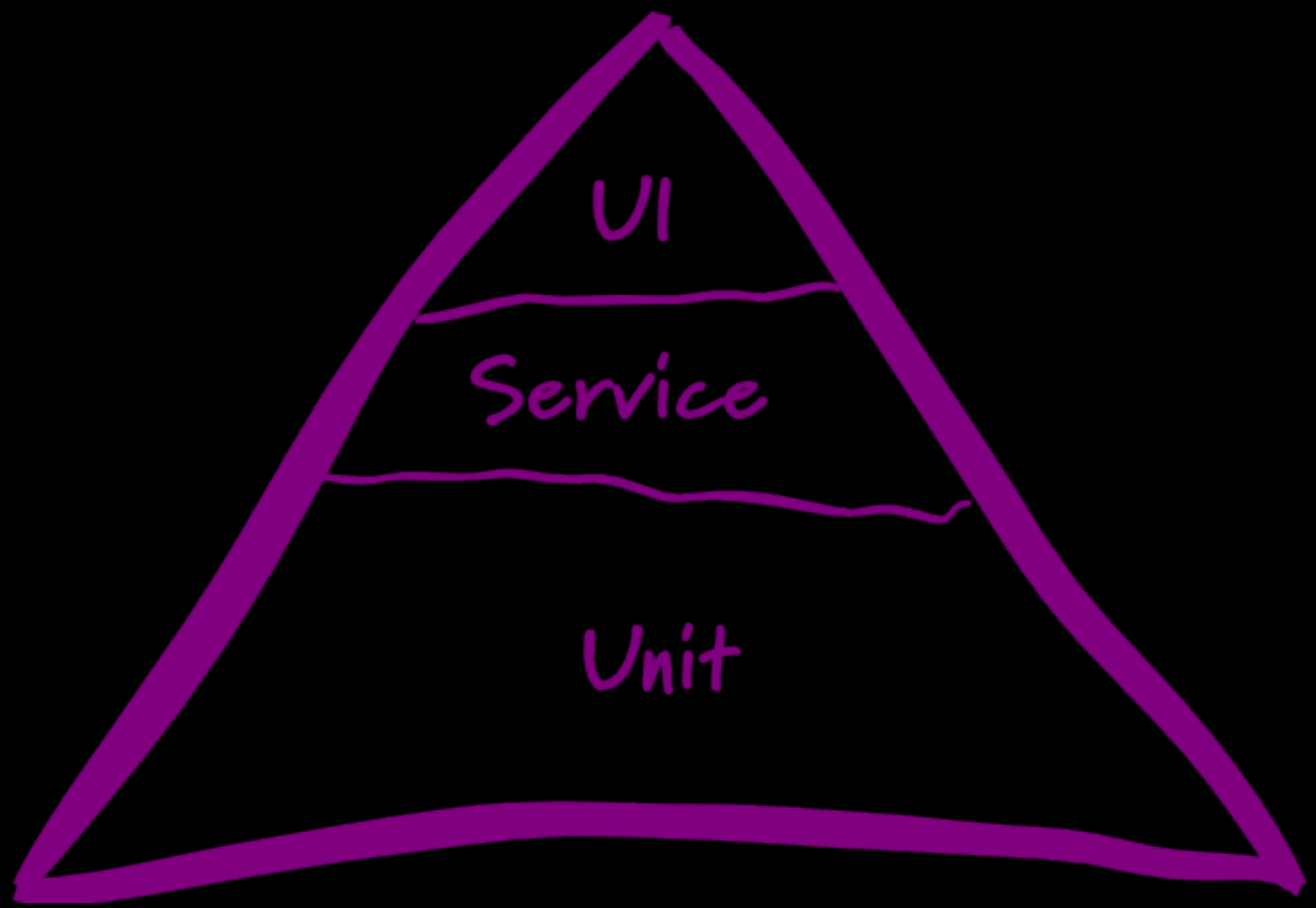
```
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
    //{{AFX_MSG_MAP(CMainFrame)
    ON_WM_CREATE()
    ON_WM_DESTROY()
    //}}AFX_MSG_MAP
    // Globale Hilfebefehle
    ON_COMMAND(ID_HELP_FINDER, CFrameWnd::OnHelpFinder)
    ON_COMMAND(ID_HELP, CFrameWnd::OnHelp)
    ON_COMMAND(ID_CONTEXT_HELP, CFrameWnd::OnContextHelp)
    ON_COMMAND(ID_DEFAULT_HELP, CFrameWnd::OnHelpFinder)
    ON_UPDATE_COMMAND_UI(ID_INDICATOR_TIME, OnUpdateIndicatorTime)
END_MESSAGE_MAP()

static UINT indicators[] =
{
    ID_SEPARATOR,
    ID_INDICATOR_CAPS,
    ID_INDICATOR_NUM,
    ID_INDICATOR_SCRL,
    ID_INDICATOR_TIME
};
```

MFC Code generator

- der UI-Recorder ist ein Code-Generator
- erzeugter Testcode ist fragil

Clean tests

- Tests sind mindestens so wichtig, wie der Produktivitätscode
- lesbarer Code auch in den Tests
- Refactorierung der generierten UI-Tests
- UI-Tests sind kein Ersatz für Unit-Tests sondern ergänzen diese bestenfalls



Testing Pyramide

Demo

Anwendungsfälle für UI-Testing

- Demos von User-stories
- App-Screenshots
  - https://github.com/fastlane/snapshot

Kritik

- TDD über den Akzeptanztest nur schwer möglich
- moving target durch Xcode updates
- Tests u.U. fragil

Links

- https://developer.apple.com/videos/play/wwdc2015-406/
- https://cleancoders.com/episode/clean-code-episode-37/show
- https://developer.apple.com/library/ios/documentation/ToolsLanguages/Conceptual/Xcode_Overview/RecordingUITests.html
- https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/iPhoneAccessibility/Introduction/Introduction.html#//apple_ref/doc/uid/TP40008785
- http://www.mokacoding.com/blog/xcode-7-ui-testing/
- https://www.bignerdranch.com/blog/ui-testing-in-xcode-7-part-1-ui-testing-gotchas/
- http://masilotti.com/ui-testing-xcode-7/
- https://medium.com/@larcus94/ui-testing-with-xcode-7-221d16bad276#.on9veakid
- https://rnorth.org/11/automated-ui-testing-in-xcode-7
- https://github.com/mmllr/CleanTweeter.git
- https://github.com/mmllr/AdaptivityDemo.git

Vielen Dank!