Маршрут вебинара

- Что такое Appium?
- Архитектура Appium и селекторы
- Установка Appium и DeviceCapabilities
- Простые python-тесты для взаимодействия с эмулятором
- Рефлексия

Цели вебинара

После занятия вы сможете

1.	Использовать Appium для тестирования мобильных приложения
2.	Настраивать Appium Server для подключения к эмулятора и облакам (например, BrowserStack / SauceLabs)
3.	Использовать Appium Inspector для определения селекторов в мобильных приложениях

Смысл

Зачем вам это уметь

1.	Для тестирования мобильных приложений можно использовать возможности протокола WebDriver Protocol (протокол опубликован https://www.w3.org/TR/webdriver/)
2.	Нередко мобильные приложения интегрируют веб-компоненты (или являются обертками вокруг веб-страниц) через WebView и их тоже надо тестировать
3.	Иногда пользователь не знает, что его переключили на мобильное приложение (например через deeplink) и это часть общей экосистемы организации

Appium

Appium

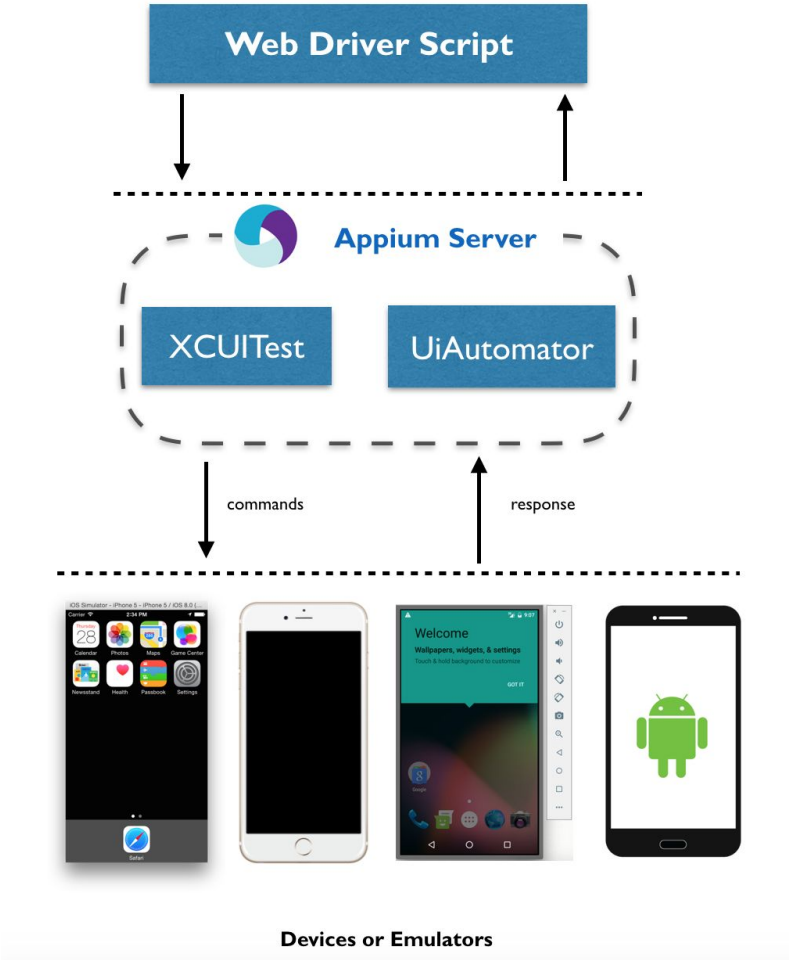
- WebDriver (используется в Selenium, например) достаточно универсален для удаленного управления любыми приложениями с интерфейсом → его можно использовать и для мобильных приложений
- На нативных платформах есть разные библиотеки тестирования в модели “черного ящика” (UiAutomator2 для Android, XCUITest для iOS)
- Также можно тестировать любую платформу (включая Desktop), если сделать единообразный механизм выбора и взаимодействия с элементами на экране

Appium

- Основные понятия: селектор (выбор элемента), действие (взаимодействие с элементом), проверка (assertion)
- Можно также получать информацию от среды выполнения и от запущенного приложения
- Для сложных случаев можно передавать команды на выбор точки на экране и выполнения жестов

Архитектура

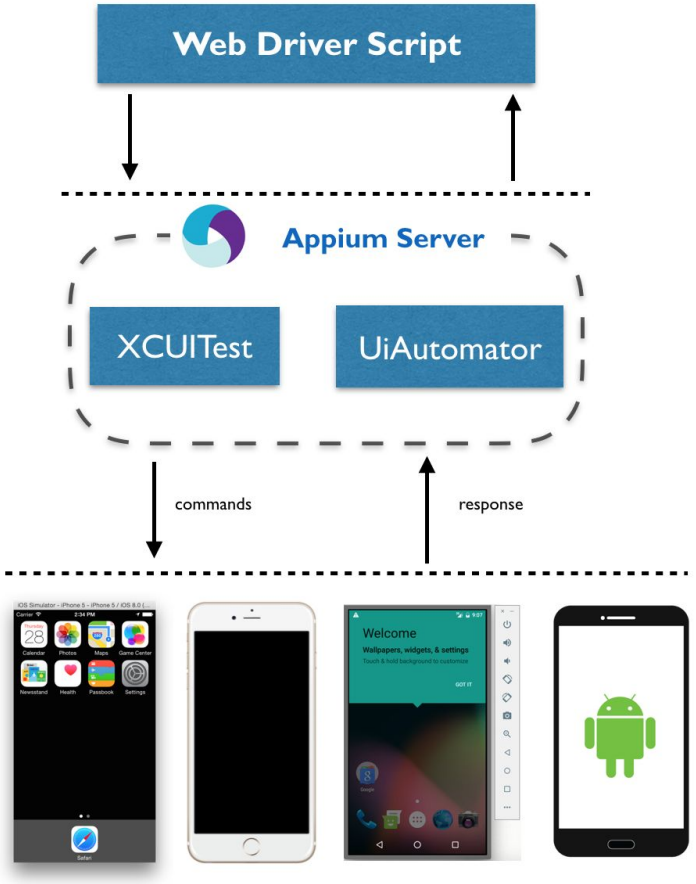
1) AppiumServer отвечает за обработку запросов и передачу команд драйверу
2) Драйвер использует нативный протокол тестирования платформы
3) Для выбора драйвера/устройства используется структура DeviceCapabilities (с перечислением необходимых критериев выбора)



Установка

1) npm install -g appium, дальше запуск сервера через CLI (appium)
2) можно поставить Appium Desktop (с GUI)

По умолчанию порт - 4723, префикс для управления - /wd/hub



Подключение

Может использоваться любой клиент:
https://appium.io/docs/en/about-appium/appium-clients/

Для Python: pip install Appium-Python-Client

Для подключения:

```
from appium import webdriver
from appium.options.android import UiAutomator2Options

options = UiAutomator2Options()
options.platformVersion = '10'
options.udid = 'emulator-5554'
options.app = PATH('calculator.apk')
self.driver = webdriver.Remote('http://127.0.0.1:4723', options=options)
el = self.driver.find_element(...)
```

Выбор устройства

Для Android:

UiAutomator2Options().udid (выбирается из adb devices)

Для iOS:

XCUITestOptions().deviceName (из xcrun simctl list devices booted или любое, например “Iphone 11” для запуска нового эмулятора) или .udid (для использования существующего устройства или эмулятора)

Можно указать как новое приложение (app), так и существующее (название пакета и класса Activity: appPackage / appActivity)

Как узнать селекторы?

- https://github.com/appium/appium-inspector/releases - скачать Appium Inspector, либо можно использовать uiautomatorviewer (только jdk8!)
- при подключении указать:
  - platformName
  - deviceName или udid
  - app (путь к ipa или apk)
- поиск driver.find_element, дальше можно взаимодействовать (click, send_keys, …), либо через коллекцию driver.find_elements(...) и итераторы/индекс (класс объектов)

Взаимодействие с элементами

– для элемента можем получить его текущее состояние и значение и использовать их в assert (например, .text, .is_displayed, is_selected, …)

- также может использовать WebDriverWait и EC (Expected Conditions), например:

```
search_input = WebDriverWait(driver, 30).until(
    EC.element_to_be_clickable((AppiumBy.ID,
"com.example:id/button")))
```

Дополнительные возможности

- запуск adb-команд (через execute_script mobile: shell)
- жесты (свайпы и прочее) - рассмотрим позднее
- поддержка взаимодействия с Web View (на следующем занятии)
- управление устройством (изменение ориентации, создание скриншота, работа с буфером обмена и геолокацией)
- платформенно-специфические команды (лучше проверить поддержку на https://appium.io Command)
- последовательности действий (объект ActionChains или ActionBuilder) и perform()