【金阳光测试】
永远免费的培训！

iOS UI自动化测试框架

陈曦明

2016-04-16

个人介绍

1、陈曦明（hyddd）

2、曾就职金山，后加入创业公司。

3、现就职于广州YY，负责内部测试平台、性能测试等。

4、个人博客：http://hyddd.cnblogs.com/

本次分享主题

在iOS上实现一个UI自动化测试框架的思路

现有的iOS UI自动化测试框架

1. Instruments
2. Appium
3. Frank

现有框架分类

1、原生型

- instruments

2、原生扩展型

在iOS上，原生扩展的框架，可以看做是js utils/lib；当我们用js编写instruments案例时，可以加载并使用它们。

- tuneup_js

- yum3k

现有框架分类

3、外部驱动原生型

在iOS上，instruments是官方提供的唯一的UI自动化测试方式，它提供了从外部访问、控制设备中App的方法。

而 外部驱动原生型，意思是从**外部**调用原生接口/库/应用程序，以满足测试的目的。

由于instruments在iOS中的唯一性，所以这里意思就是调用instruments了。

现有框架分类

- InstrumentDriver
- ios-driver
- appium
- macaca

现有框架分类

4、内嵌测试型

所谓内嵌测试型，意思是在APP项目中，配合单元测试框架XCTest(OCUnit)+系统提供的accessibility API（如：盲人控制相关API），在启动产品后，直接执行UI测试。

- KIF（google）

现有框架分类

5、内嵌Server型

（4）和（5）类似，不过（5）嵌入的是Server，具体操作由外部Client控制。

- Frank
- Calabash

框架归纳

非入侵式框架：

1、原生型；

2、原生扩展型；

3、外部驱动原生型；

入侵式框架：

4、内嵌测试型；

5、内嵌Server型；

注：“入侵”指测试入侵产品，内嵌于产品内。

框架归纳

Client-Server模式框架：

3、外部驱动原生型；

5、内嵌Server型；

非Client-Server模式框架（非CS模式框架，一般无法同时控制多台设备）

1、原生型；

2、原生扩展型；

4、内嵌测试型；

非入侵式 vs 入侵式

入侵式：

需要把测试资源嵌入到产品中，这会造成 外发产品 与 被测试的产品 不是同一个东西，可能会影响性能、产生误报或者 遗漏问题 等。

本人更倾向采用 非入侵式框架。

非入侵式 vs 入侵式

那对比“非入侵式”，“入侵式”存在的意义是什么？

有些时候我们不得不用入侵式解决问题。

场景1：

代码覆盖率。无论android的emma，还是iOS的gcov，都需要对产品打桩。

非入侵式 vs 入侵式

场景2：

某些开发框架的产品，如：cocos2d-x。

非入侵式 vs 入侵式

基本所有的系统，显示资源/控件在系统内部，都是通过obj **tree**形式组织管理。如：浏览器，android，iOS，cocos2d-x...

而我们测试时的控件定位，就是依赖 系统提供的obj tree对外接口，获取我们所需的控件（或者控件坐标）；

- android：instruments/hierarchyviewer
- iOS：instruments
- 浏览器：js engine

而cocos2d-x出于安全/其他原因，并无对外提供obj tree接口。

所以我们测试cocos2d-x产品时，只能要么通过绝对坐标，要么通过opencv图像对比判断，无法获取具体控件！

在iOS上实现一个UI自动化测试框架的思路

基于以上分类，我的框架分类为：**非入侵式 + CS模式**

类似的有：

- InstrumentDriver
- ios-driver
- appium
- macaca

其实的它们的设计、方案和我之前的做法都是类似的……

在iOS上实现一个UI自动化测试框架的思路

1、测试框架（CS）架构中的角色与职责

- 要详细的话，可以参考ios-driver、appium、maccaca，它们都是基于webdriver协议规范的api->JsonWireProtocol，里面定义了每个角色和其职责，你可以在项目代码中找到相应的部分。

在iOS上实现一个UI自动化测试框架的思路

Basic Terms and Concepts

Client

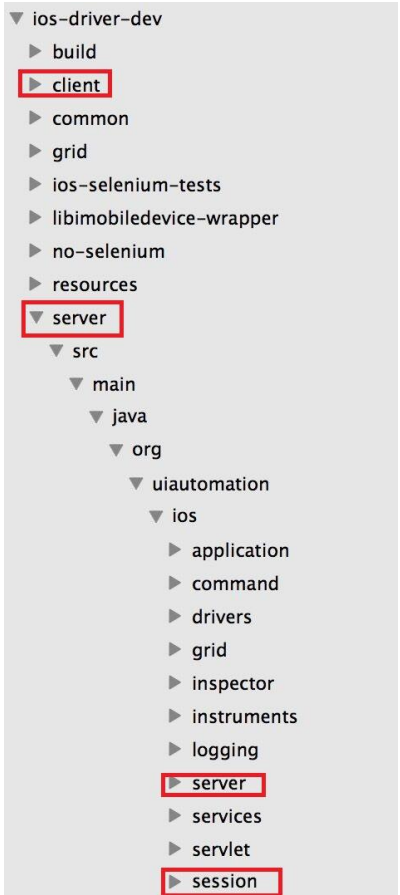
The machine on which the WebDriver API is being used.

Server

The machine running the RemoteWebDriver. This term may also refer to a specific browser that implements the wire protocol directly, such as the FirefoxDriver or IPhoneDriver.

Session

The server should maintain one browser per session. Commands sent to a session will be directed to the corresponding browser.

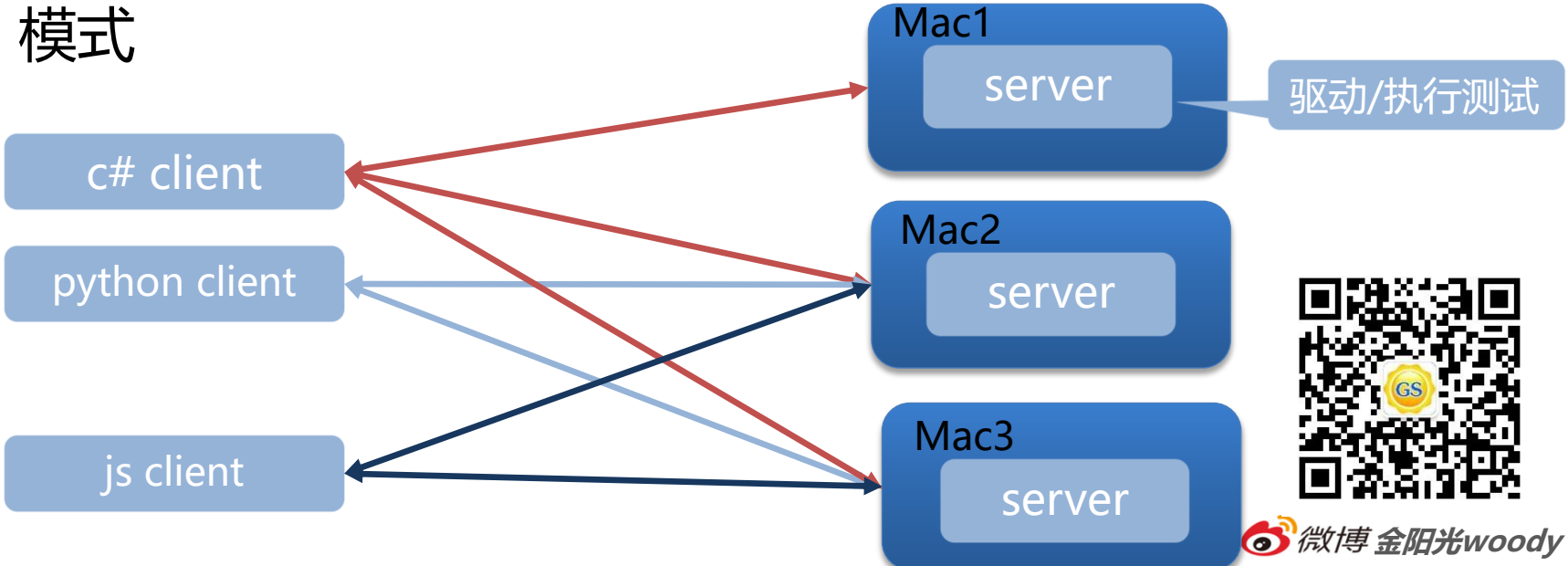


在iOS上实现一个UI自动化测试框架的思路

- 简单的看，主要角色就是: client和server，其职责如下：

在iOS上实现一个UI自动化测试框架的思路

在Mac OS下，instruments能且只能启动一个实例（一台设备/模拟器），所以真正使用时，可能是以下调用模式



在iOS上实现一个UI自动化测试框架的思路

client需要解决基本问题：

1、提供一套简洁、可读性高的api；

2、把api操作翻译为server可识别指令集；

server需要解决基本问题：（**核心问题**）

1、安装、卸载app；

2、如何驱动instruments；

3、是否可以适应客户端的单步debug；

在iOS上实现一个UI自动化测试框架的思路

如何解决server基本问题？

解决server问题的难度在于，iOS和Mac OS闭源并且系统不提供documented api或者公开的方式解决问题。我们只能逆向或者收集别人已破解的方式解决问题，但这也引入了其他问题，因为我们使用的是非规范化的手段解决问题，当系统升级时，一切就有可能呵呵哒了。

在iOS上实现一个UI自动化测试框架的思路

1、在Mac上，命令行安装、卸载app（无需越狱）

Xcode 5之前，可以采用fruitstrap（现已不再维护）

Xcode 5之后，使用libimobiledevice。原理是破解iTunes和iPhone通讯后，实现了该协议。iTools也是用这玩意。

在iOS上实现一个UI自动化测试框架的思路

2、如何驱动instruments

https://developer.apple.com/library/mac/documentation/Darwin/Reference/ManPages/man1/instruments.1.html

```
NAME
     instruments

USAGE
     instruments [-t template] [-D document] [-l timeLimit] [-i #] [-w device] [[-p pid] | [application [-e
     variable value] [argument ...]]]

DESCRIPTION
     Runs an Instruments template from the command line.

     Options are :

     -t template
              The path to the template to run

     -s       Show list of known templates and exit

     -D document
```

在iOS上实现一个UI自动化测试框架的思路

3、是否可以适应客户端的单步debug

无论是client控制多个Server(devices)测试，或者client单步调试，都需要server能单步执行指令。而instruments启动是一次性加载脚本并执行，所以需要额外处理。

在iOS上实现一个UI自动化测试框架的思路

3、是否可以适应客户端的单步debug

解决方案：

在instruments加载的脚本中，启动http/socket server，另server与instruments启动的server建立连接并监听，当有发现有指令时，马上执行，执行结果再callback回server！

在iOS上实现一个UI自动化测试框架的思路

我之前做法：

```
while(!isEnd) {
    __init();
    var result = "";
    var cmd = "";
    try {
        result = host.performTaskWithPathArgumentsTimeout("/Users/chenximing/Documents/
        MobileTools/MobileTools/ios/IosTestAgent/TcpSocket.sh", ["GETCMD", "0"], 60);
    }
    catch(e) {
        target.delay(0.001);                    获取指令
        continue;
    }

    cmd = result.stdout;
    if (!cmd) {
        UIALogger.logMessage("no cmd");
        target.delay(0.001);
        continue;
    };

    UIALogger.logMessage("stdout : " + cmd);
    try {
        sendToServer = eval(cmd);
        if (!sendToServer) {                    执行指令
            sendToServer = ""
        }
    }
    catch (e) {
        sendToServer = "Exception#" + e;
    }

    if (typeof(sendToServer) !=  "string") {
        sendToServer = "Exception#return value is not a string";
    };

    UIALogger.logMessage("sendToServer : " + sendToServer);      返回结果
    try {
```

在iOS上实现一个UI自动化测试框架的思路

ios-driver做法

```
/**
 * Keep executing commands and sending the responses until the stop command is rec
 */
commandLoop: function () {
    // first command after registration sends the capabilities.
    var init = {};
    if (this.COMMUNICATION_MODE === "MULTI") {
        UIATarget.localTarget().frontMostApp().setPreferencesValueForKey(null, 'cm
    }
    init.firstResponse = UIAutomation.getCapabilities();
    var response = this.createJSONResponse(this.SESSION, 0, init);
    var ok = true;
    while (ok) {
        try {
            var request;
            if (this.COMMUNICATION_MODE === "CURL") {
                request = this.postResponseWithCURLAndGetNextCommand(response);
            } else {
                request = this.postResponseMultiAndGetNextCommand(response);
            }
            if (request === "stop") {
                ok = false;
                log("end of the command loop.");
                return;
            } else {
                try {
                    response = eval(request);
                } catch (err) {
                    if (err.status) {
                        response =
                        this.createJSONResponse(this.SESSION, err.status, err.mess
                    } else {
                        response = this.createJSONResponse(this.SESSION, 17, err.m
                    }
```

在iOS上实现一个UI自动化测试框架的思路

谢谢！