



App Developers Conference • Los Angeles, CA
Nov 6 2013

Jonathan Lipps • Sr Developer • Sauce Labs

@AppiumDevs • @jlipps • @saucelabs



Ecosystem &
Integrations



Project Lead &
Architect

<http://mobro.co/jlipps>

Jonathan Lipps • Sr Developer • Sauce Labs

@AppiumDevs • @jlipps • @saucelabs

appium introduction



“Mobile is taking over the world.” So how do we **scale mobile quality**?



Testing and QA are **important but painful**. We want more code, less tap!



Risk grows with **complexity**.
Small changes can have huge
unintended consequences

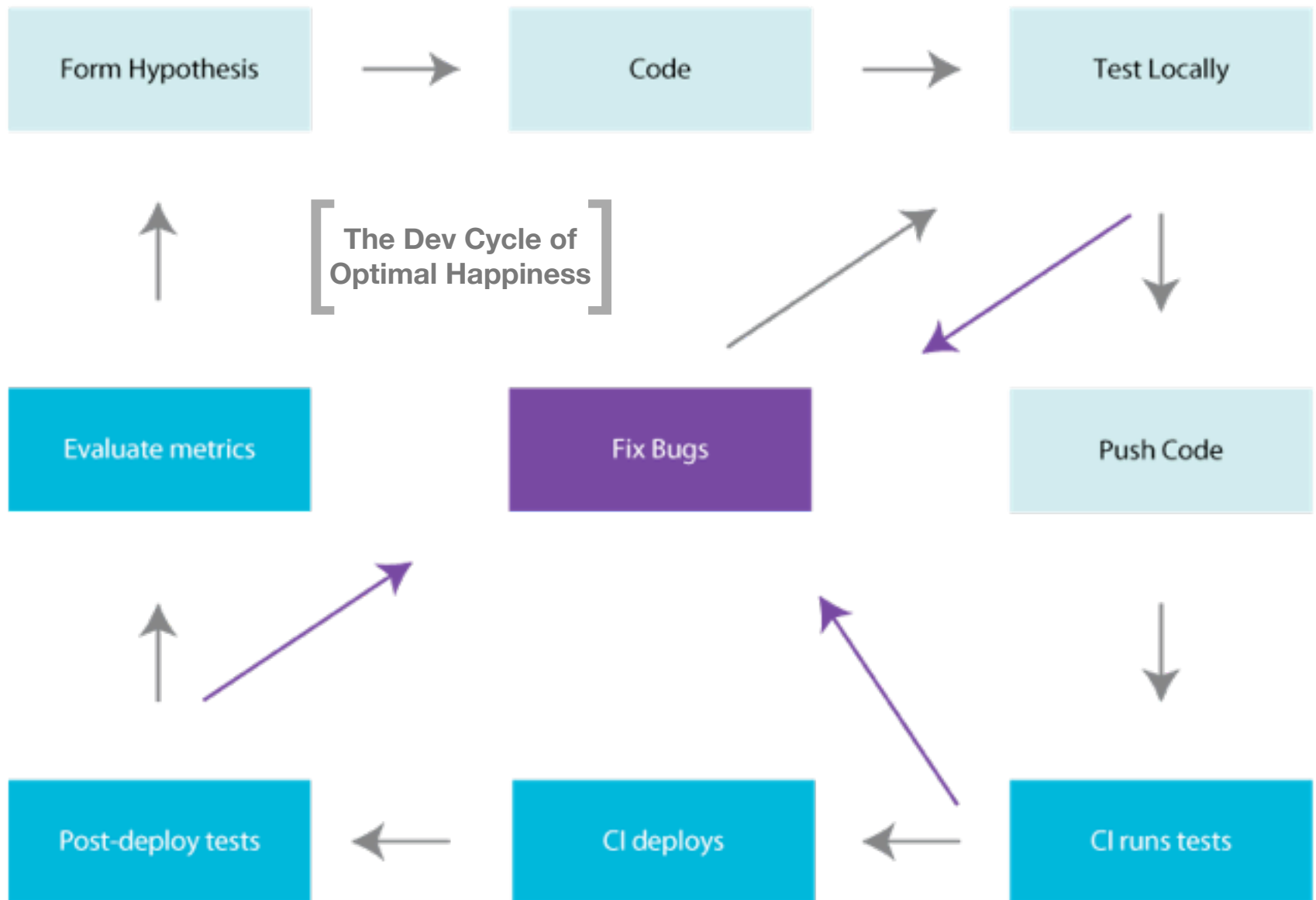


Automated testing is the solution for a fast dev cycle that maintains high quality



Continuous Integration is
awesome: automated testing
+ automated deployment





appium is the cross-platform
solution for native and hybrid
mobile automation



appium makes automated testing possible for mobile, setting the stage for real CI



appium raison d'être



iOS

calabash-ios
Frank
UIAutomation
ios-driver
KeepItFunctional

Android

calabash-android
MonkeyTalk
Robotium
UiAutomator
selendroid



Rule 1

Test the same app you submit to the marketplace



Rule 2

Write your tests in any language and any framework



Rule 3

Use a standard automation specification and API



Rule 4

Build a large and thriving
open-source community



Platform Support

Real devices   

Simulators   

Hybrid apps   

Mobile web   

Robots?!



Demo



```
1. jlipps@numenor: ~APPIUM_HOME (zsh)
~APPIUM_HOME (zsh)
.de/woven/test
→ test git:(master) mocha -t 60000 -R spec gtac.js []

~APPIUM_HOME
→ ~APPIUM_HOME git:(master) node server.js -V --without-delay --fast-reset
```



appium architecture



appium is an HTTP server
that creates and handles
WebDriver sessions



Selenium WebDriver



has been the standard for
browser automation



Selenium WebDriver

is a HTTP API

POST /session

POST /session/element

GET /session/element/:id/:attr



Selenium WebDriver
has clients in every* language



Selenium WebDriver

is a W3C working draft*



appium extends the
WebDriver protocol with
mobile-specific behaviors



appium is working with the
Selenium project so we can
standardize these extensions



Automation Voodoo

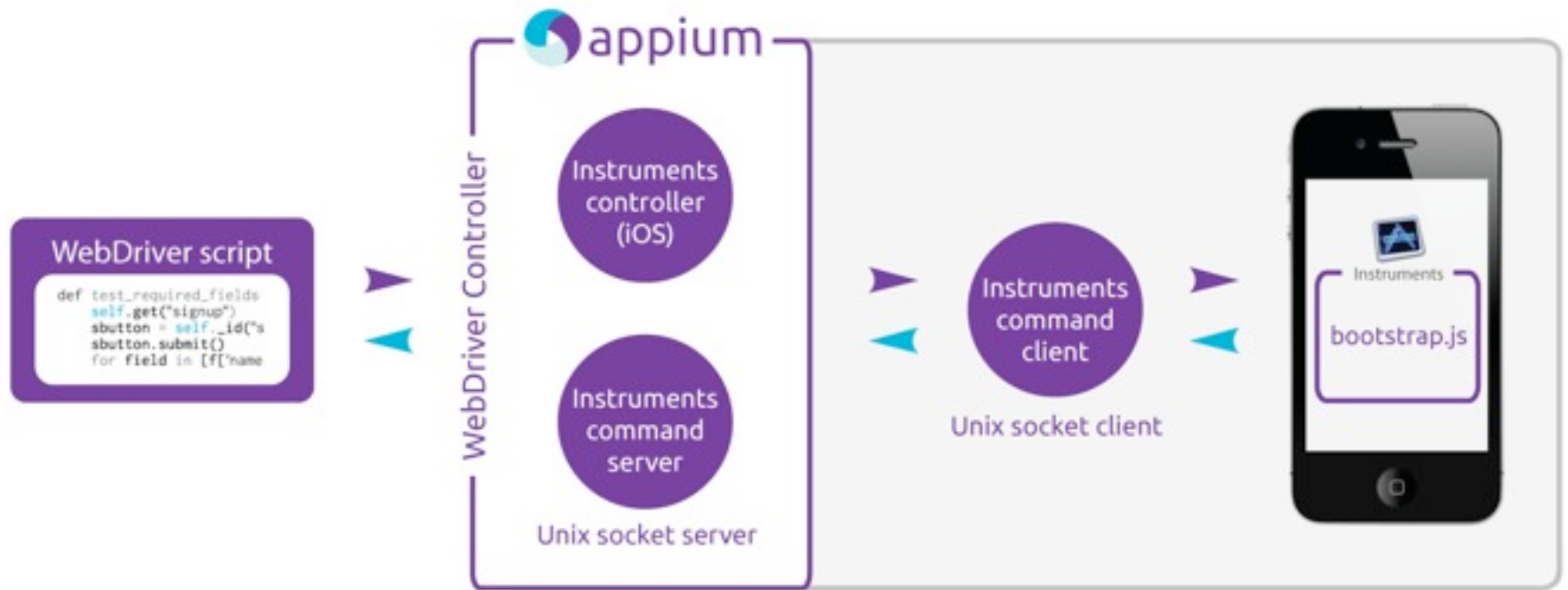
Apple Instruments & UIAutomation for iOS

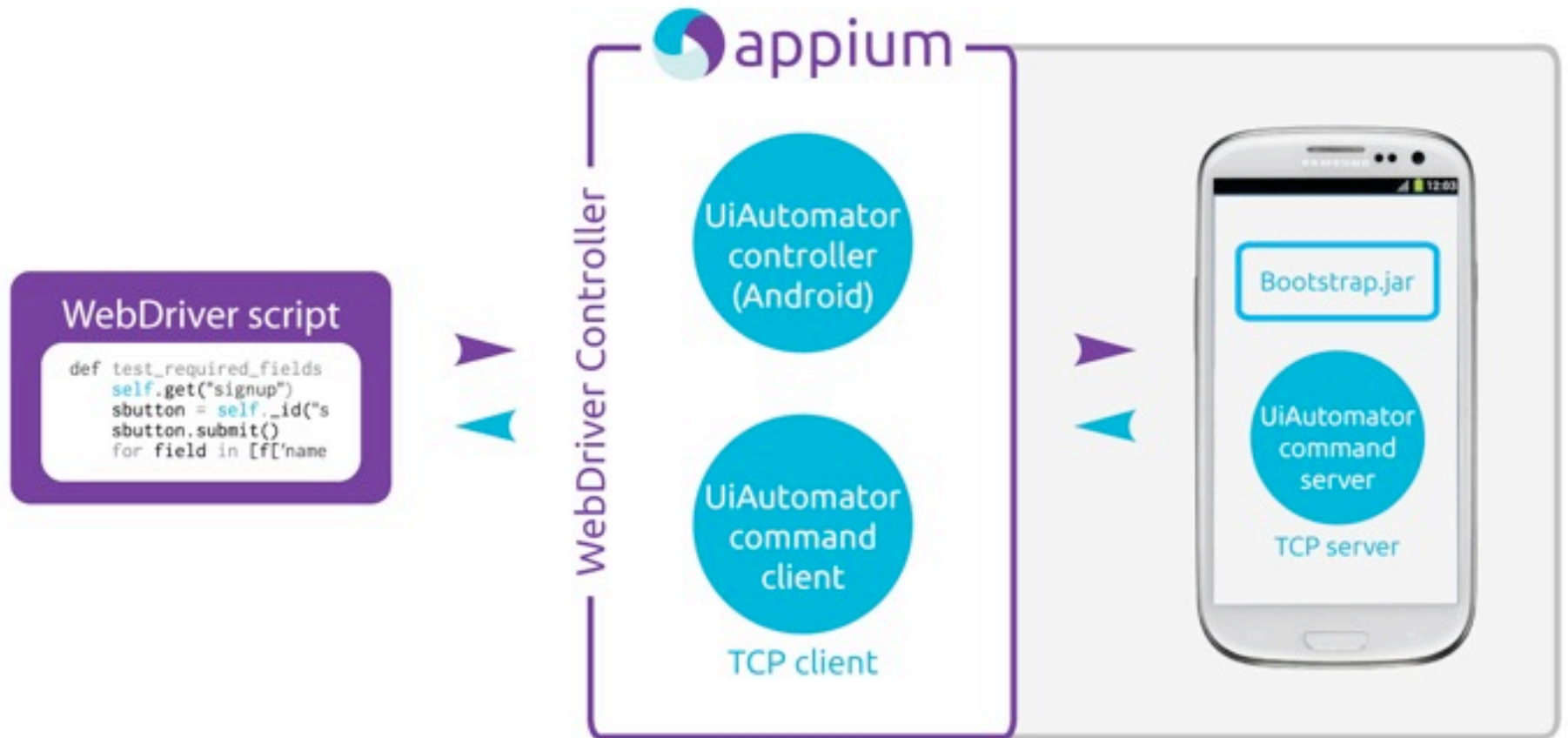
Google UiAutomator for Android (4.2.1 up)

Selendroid for older Android & hybrid

Marionette for FirefoxOS







appium setup



Requirements

Basically the same as dev toolkits for iOS, Android

Xcode + iOS SDK

Android SDK + Java



Install: Option One

Clone from GitHub

```
REPO="appium/appium.git"  
git clone https://github.com/\$REPO  
cd appium && ./reset.sh  
node .
```



Install: Option Two

Install from NPM

```
npm install -g appium  
appium
```




Install: Option Three

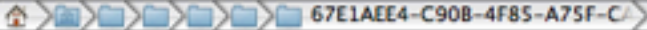

Download the GUI app

```
# github.com/appium/appium/releases  
open /Applications/Appium.app
```




Appium

IP Address: Port: ☐ Use Remote Server 

☒ App Path:  67E1AEE4-C90B-4F85-A75F-CA  Zoosk.app

☐ UDID ☐ Force Device

☐ BundleID ☐ Use Mobile Safari





Appium Preferences

☐ Check For Updates
☐ Quiet Logging
☐ Keep Artifacts
☒ Reset Application State After Each Session
☒ Prelaunch Application
☒ Developer Mode

Robot Settings

☐ Enabled Address: Port:

iOS Settings

☐ Use Native Instruments Lib
☐ Force Orientation

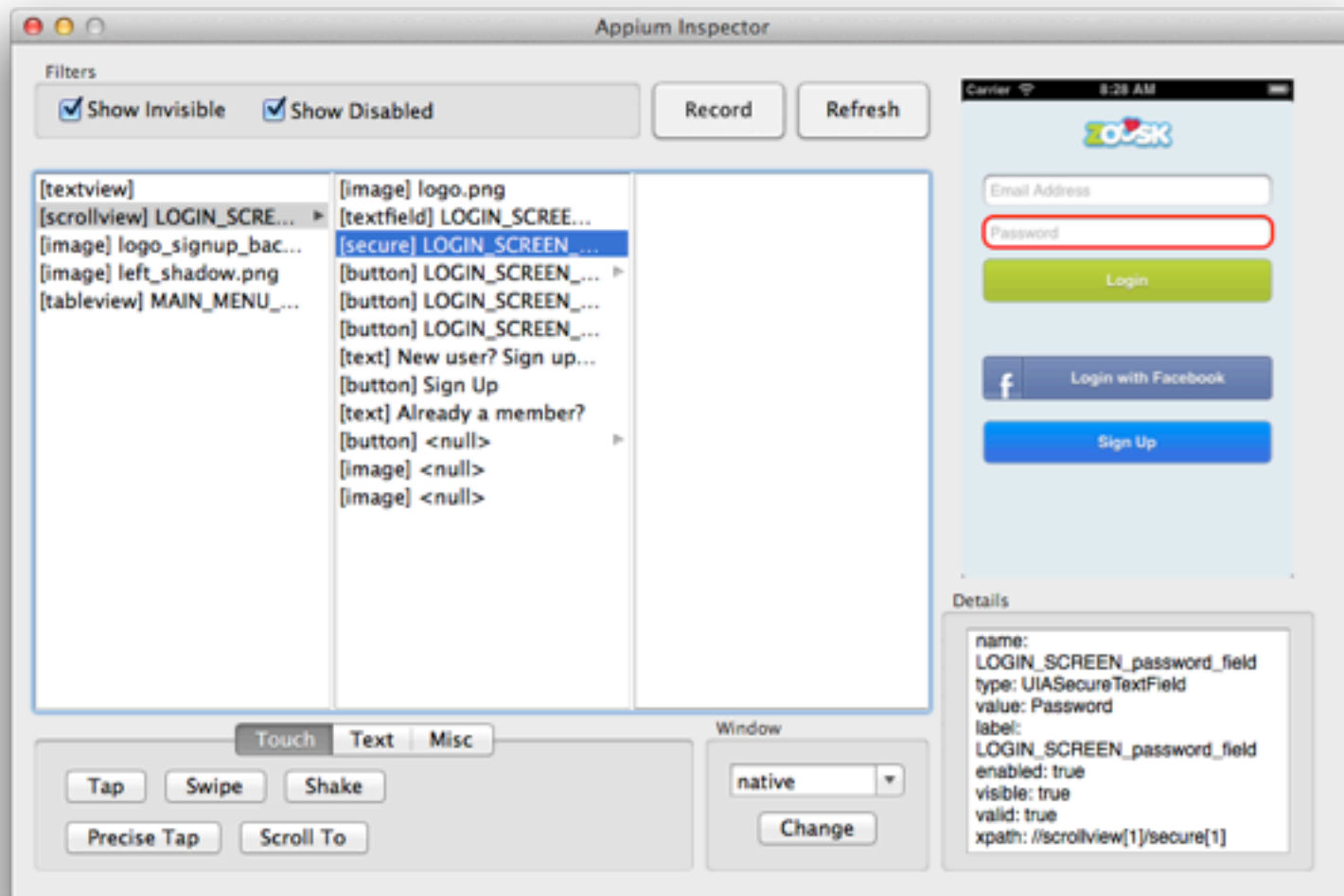
Android Settings

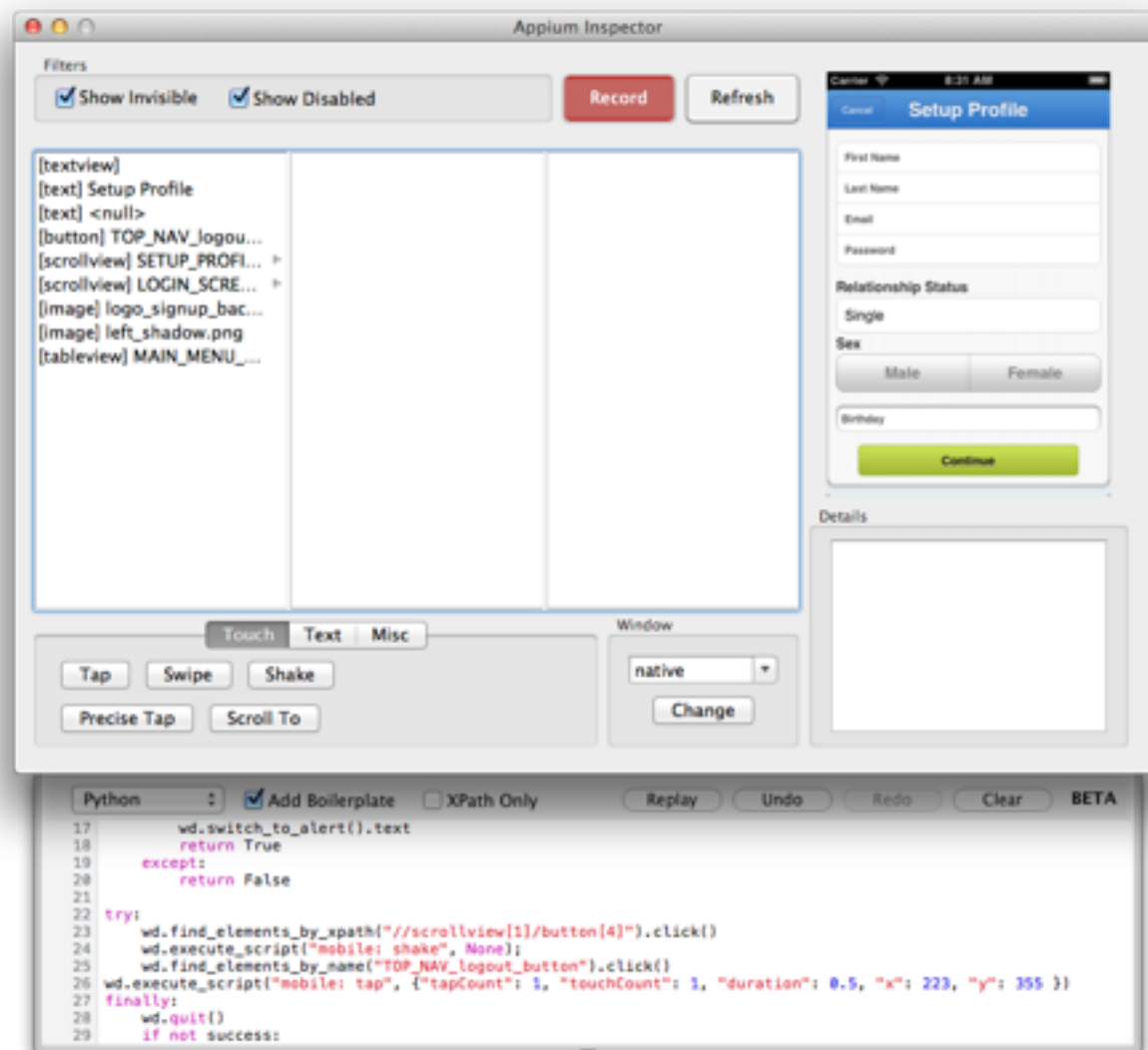
☒ Full Reset (Uninstall .apk)
☐ Device Ready Timeout

Developer Settings

☐ Use External NodeJS Binary
☐ Use External Appium Package
☐ NodeJS Debug Port ☐ Break On Application Start







appium test model



Start/stop a session

```
class AppiumTestCase(unittest.TestCase):

    def setUp(self):
        appium = "http://localhost:%s/wd/hub" % 4723
        self.desired_caps = {
            device: 'iPhone Simulator',
            app: '/abs/path/to/my.app',
        }
        self.driver = webdriver.Remote(appium, self.desired_caps)

    def tearDown(self):
        self.driver.quit()
```



Find elements

```
# by accessibility label
```

```
el = self.driver.find_element_by_name("Sign In")
```

```
# by element type
```

```
el = self.driver.find_element_by_tag_name("button")
```

```
# by hierarchy
```

```
el = self.driver.find_element_by_xpath("//button[@text='Hi ']")
```

```
# by Android ID
```

```
el = self.driver.find_element_by_id("myId")
```



Interact with elements

```
el = self.driver.find_element_by_tag_name("button")  
print el.text # get text of an element  
el.click()    # click an element
```

```
username = self.driver.find_element_by_name("username")  
username.send_keys("jlipps") # type in a field
```

```
slider = self.driver.find_element_by_tag_name("slider")  
slider.send_keys("0.8") # move slider to 80%
```



```
def test_woven(driver, config):  
    sign_in = driver.find_element_by_name("Sign In")  
    sign_in.click()  
    email_field = driver.find_element_by_xpath(config.email_sel)  
    email_field.send_keys(woven_user)  
    pass_field = driver.find_element_by_xpath(config.pass_sel)  
    pass_field.send_keys(woven_pass + "\\n")
```



```
public class WovenTest {  
  
    private WebDriver driver;  
  
    @Before  
    public void setUp() throws Exception {  
        // set up appium  
        File app = new File("Woven.app");  
        DesiredCapabilities capabilities = new DesiredCapabilities();  
        capabilities.setCapability(CapabilityType.BROWSER_NAME, "iOS");  
        capabilities.setCapability(CapabilityType.VERSION, "6.0");  
        capabilities.setCapability(CapabilityType.PLATFORM, "Mac");  
        capabilities.setCapability("app", app.getAbsolutePath());  
        driver = new SwipeableWebDriver(new URL("http://127.0.0.1:4723/wd/hub"), capabilities);  
    }  
}
```



```
@Test
public void signIn() throws Exception {
    WebElement signIn = driver.findElement(By.name("Sign In"));
    signIn.click();
    WebElement userField = driver.findElement(By.XPath(config.emailSel));
    userField.sendKeys(wovenUser)
    WebElement passField = driver.findElement(By.XPath(config.passSel));
    passField.sendKeys(wovenPass + "\\n");
}
```



appium scale



appium is great for local test development, but has limitations when scaling up for use in CI



Sauce Labs is great for scale
when you need to run a lot of
appium tests in your build



Run tests on Sauce

LOCAL

```
def setUp(self):
```

```
    appium = "http://localhost:%s/wd/hub" % 4723
```

```
    self.desired_caps = {  
        device: 'iPhone Simulator',  
        app: '/abs/path/to/my.app',  
    }
```

```
    self.driver = webdriver.Remote(appium, self.desired_caps)
```

#SAUCE

```
def setUp(self):
```

```
    appium = "http://%s:%s@ondemand.saucelabs.com"
```

```
    appium = appium % (SAUCE_USERNAME, SAUCE_PASSWORD)
```

```
    self.desired_caps = {  
        device: 'iPhone Simulator',  
        app: '/abs/path/to/my.app',  
    }
```

```
    self.driver = webdriver.Remote(appium, self.desired_caps)
```



 saucelabs.com/mobile

FREE FOR OPEN SOURCE



appium is also compatible with
Selenium Grid which helps with
your own closet cloud



appium real devices



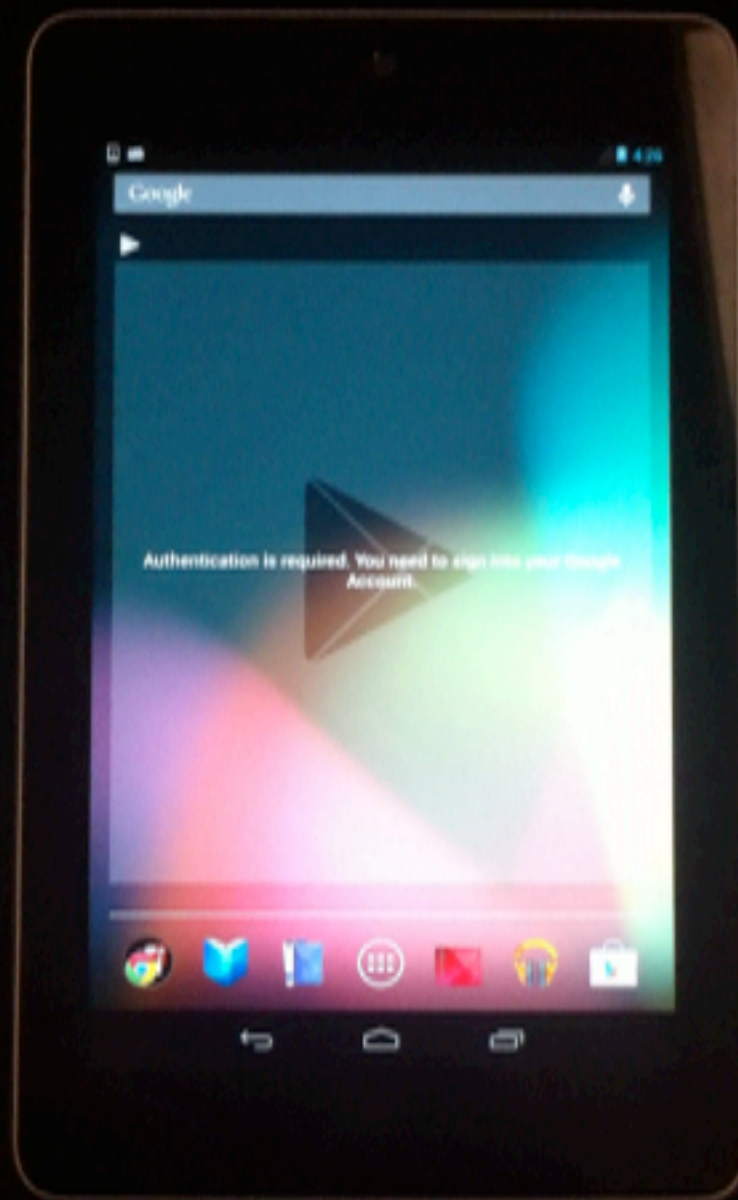
```
var wd = require("yiewd");
var desiredCaps = {
  device: 'Android'
  , "app-package": "com.android.contacts"
  , "app-activity": "activities.PeopleActivity"
};

var bc = function(t) { return "//button[contains(@text, '" + t + "')]"; };
var ec = function(t) { return "//editText[contains(@text, '" + t + "')]"; };
var tc = function(t) { return "//text[contains(@text, '" + t + "')]"; };
```



```
wd.remote('localhost', 4723).run(function*() {  
  yield this.init(desiredCaps);  
  yield this.setImplicitWaitTimeout(5000);  
  yield this.elementByXPath(bc('Create')).click();  
  yield this.elementByXPath(ec('Name')).sendKeys("John Smith");  
  yield this.elementByXPath(ec('Phone')).sendKeys("(555) 555-5555");  
  yield this.elementByXPath(ec('Email')).sendKeys("john.smith@google.io");  
  yield this.elementByXPath(tc('Done')).click();  
  yield this.elementByName("Add to favorites").click();  
  yield this.elementByName("Edit").click();  
  yield this.elementByXPath(tc('Mobile')).click();  
  yield this.elementByXPath("//checkedTextView[@text='Home']").click();  
  yield this.elementByXPath(tc('Done')).click();  
  yield this.elementByName("More options").click();  
  yield this.elementByXPath(tc('Delete')).click();  
  yield this.elementByXPath(bc('OK')).click();  
  yield this.quit();  
});
```





appium mobile web



x _xamples/node

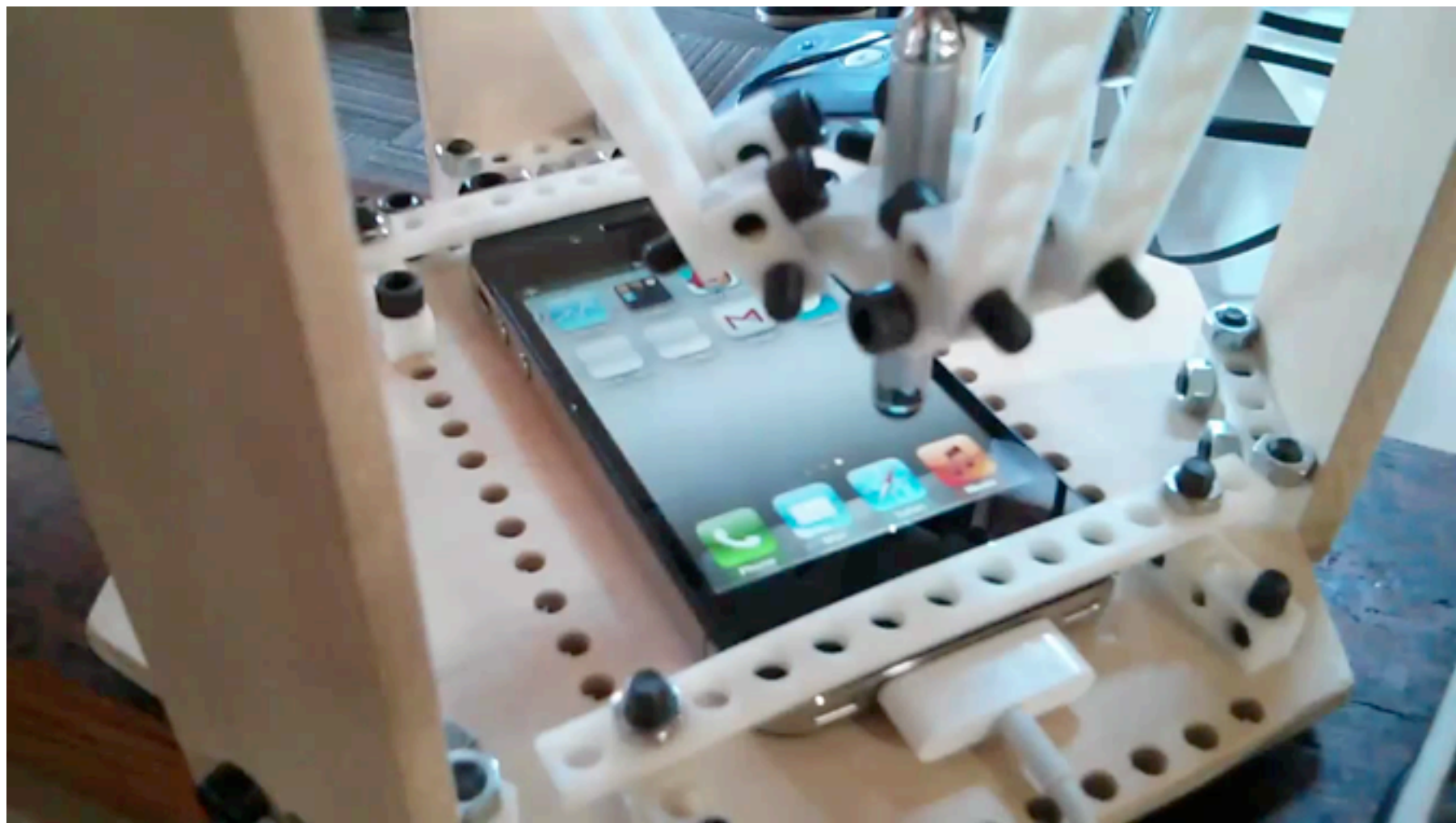
~/C/a/s/e/node git:master >>> |

x _ppium-jlipps

~/C/appium-jlipps git:master >>>

appium robots





appium hackers



We need you...

Node.js devs (for Appium server)

Obj-c devs (for Appium.app)

C#.Net devs (for Appium.exe)

Java devs (for Appium's Android bootstrap)

Windows Phone devs (we need to support it!)

Hardware hackers (for robot support)



Questions?



<http://appium.io>

<https://github.com/appium/appium>

@AppiumDevs • @jlipps • @saucelabs

WE'RE HIRING!
<http://saucelabs.com/careers>

Thanks!



<http://appium.io>

<https://github.com/appium/appium>

@AppiumDevs • @jlipps • @saucelabs