

Cross Platform Mobile Application Testing

-Vinod Doshi

Objective

- Mobile Application Testing Needs.
- Challenges
- Current platform specific tools
- Cloud Testing
- Testing Strategies and Recommendations
- Generic Solution For All The Platforms

Mobile Application Testing Needs

- Mobile applications are booming
- Releasing stable apps is a huge challenge
- Pressure to get to market leaves many organizations overlook small glitches, crashes, malfunctions etc.

Challenges

- Diversity of devices /platforms/OS Versions/Carriers
- Fragmentation. Multiple OS versions and devices.
- Test on a huge array of devices with different sizes and screen resolutions
- Rendering of images and positioning of elements on a screen is unsuitable in some devices.
- Developers are forced to develop multiple versions of the same app to work with different OS.

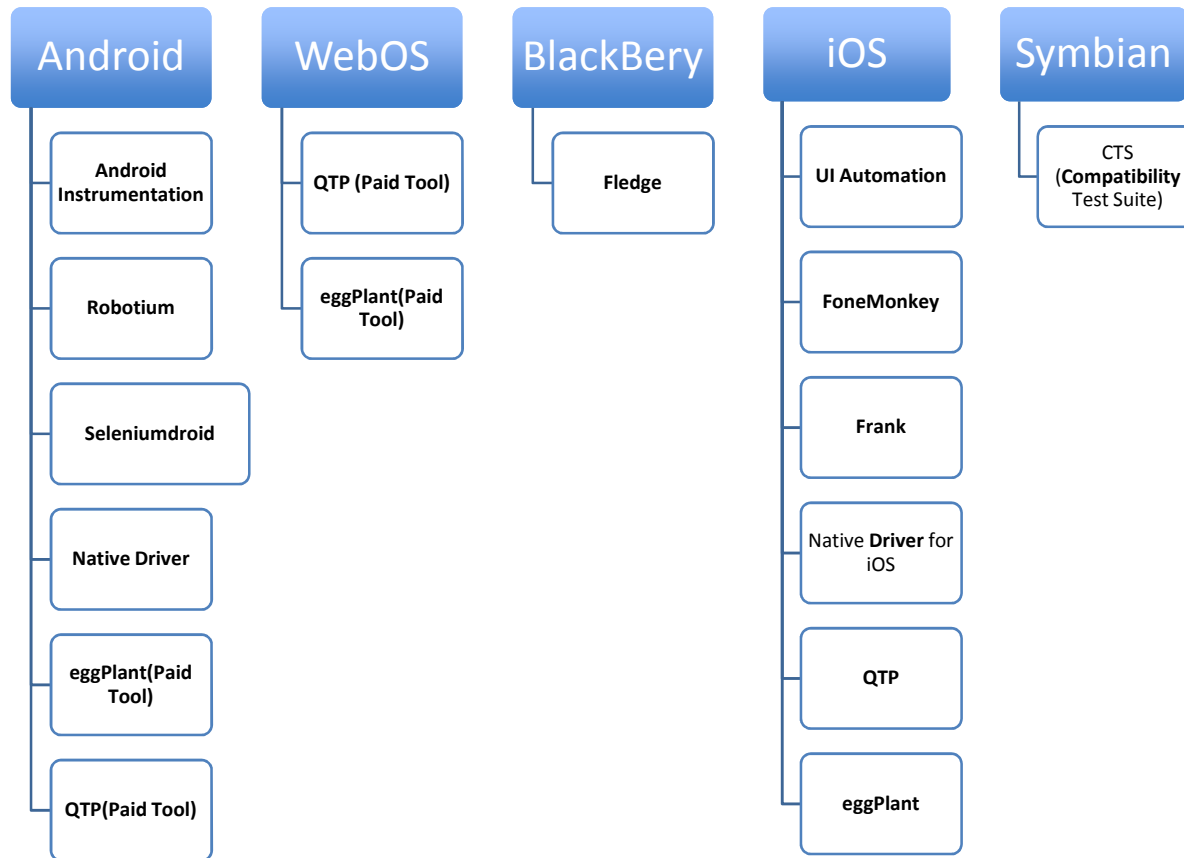
Challenges

- Due to a large number of devices available in the market, it is not feasible to buy a new device every time.
- Simulators are not reliable.
- Creating a testing lab with physical devices can be very expensive.
- Continuously keep on investing in devices

Challenges

- Testing on devices vs. testing on an emulator.
- Lack of Test Automation.
- Traditional testing which is Manual testing in is time consuming and error prone.
- Code Coverage
- Stress\Load Testing
- Device Profiling

Existing Mobile Platforms And Available Test Automation Tools



Cloud Testing Tools \ Frameworks

- Perfecto Mobile - <http://www.perfectomobile.com/>
- Device Anywhere - <http://www.deviceanywhere.com/>
- Pros:
 - Rent per hour. Swap Devices
 - Large number of devices available for testing
 - Tests can be run on several devices in parallel
 - Web based interface
 - Build Integration
 - Test incoming calls and text without needing a cell plan
 - Automated test execution is recorded to video to investigate failures
 - Device logs are recorded to help troubleshooting

Cloud Testing Tools \ Frameworks

- Perfecto Mobile - <http://www.perfectomobile.com/>
- Device Anywhere - <http://www.deviceanywhere.com/>
- Cons:
 - Subscription Model
 - High Cost
 - Lock in
 - Internet connectivity issues
 - Automation is Image Based, time consuming and taxing
 - Automation cannot be used outside the framework

Guidelines for testing mobile applications

- Select the right automation tool for every platform
- Use a generic automation tool like Sikuli
- Split the testing on emulator vs devices

Type of Testing	Testing on Emulators	Testing on Devices
Unit Testing	Yes	No
Integration Testing	Yes	No
Regression Testing	No	Yes
Compatibility Testing	No	Yes
Performance Testing	No	Yes
Security Testing	No	Yes
Device Profiling	No	Yes

Guidelines for testing mobile applications

- Consider limited use of cloud testing framework
- Do not neglect Performance and Stress testing, Device profiling
- Invest in setting up a test lab. Do not completely rely on manual testing
- Use a good TCM tool
- Use BDD

Sikuli – Image Based Test Automation

- One tool should support all desired platforms and also runs on a device
- Test tool should support testing of various screen types /resolutions
- Automation Cost Reduction
- Increased productivity
- Better application Quality

What is Sikuli?

Visual technology
to automate GUI
using images.

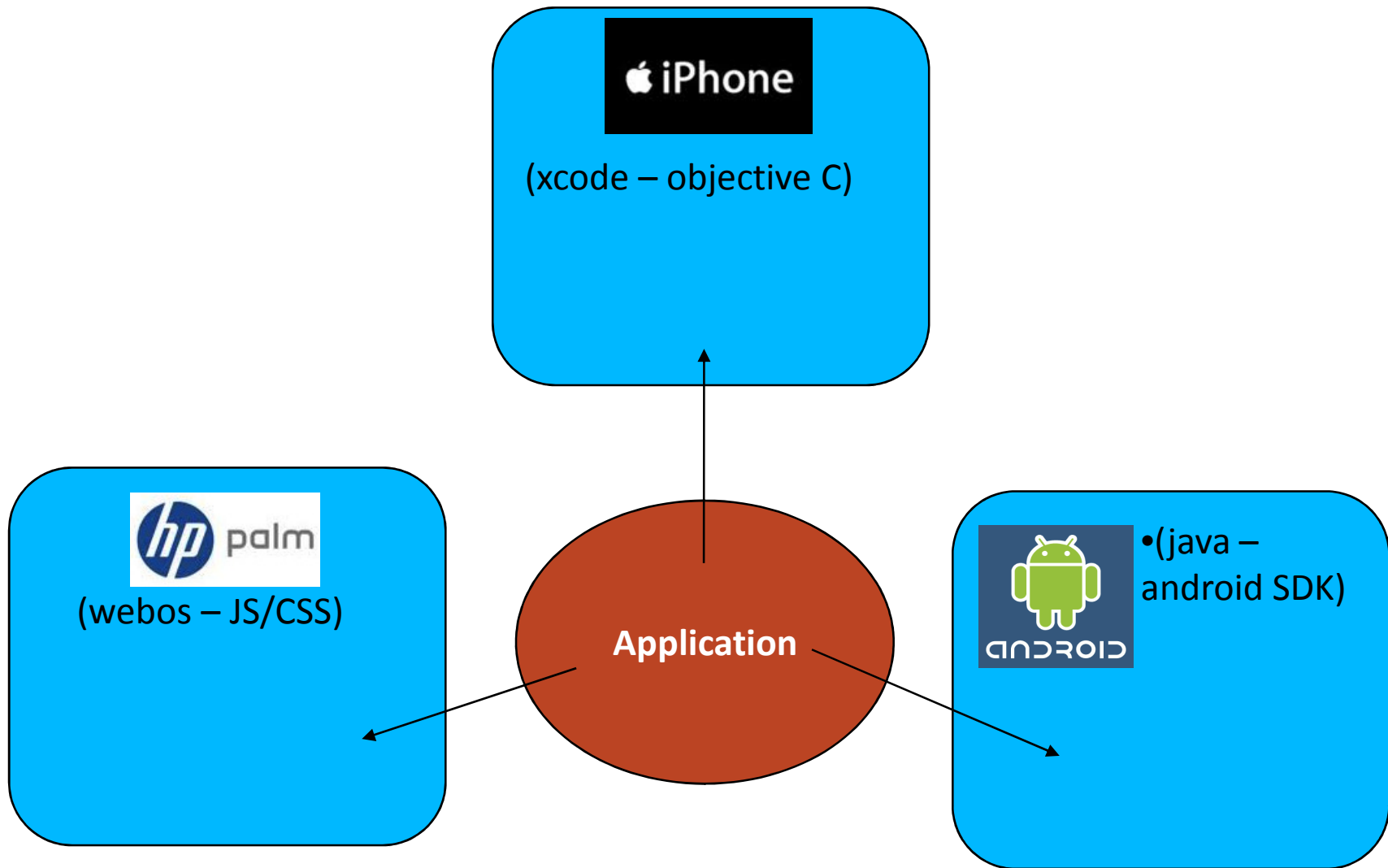
MIT research
project. Open
Source license.

Sikuli IDE

Sikuli Script API

Automates
anything on screen
without internal
API's support

Works on
Windows, Mac,
Linux.







Advantages of using Sikuli

- Automate user operations on images e.g. Click, type, drag-drop, mouse actions etc.
- Visual verification of the expected output
- Not dependent on platform underneath
- Can be used to automate emulator as well as device.
- Can accurately test GUI and rendering of applications.
- Write test outside the device
- Easy to automate.

Sample Sikuli Script

The screenshot displays the Sikuli X-1.0rc2 application window titled "Sikuli X-1.0rc2 - Untitled". The interface includes a top toolbar with icons for "Take screenshot", "Insert image", "Create Region", "Run", and "Run in slow motion", along with a "Find" search bar. On the left, there are three panels: "Find" (listing methods like find, findAll, wait, waitVanish, exists), "Mouse Actions" (listing click, doubleClick, rightClick, hover, dragDrop), and "Keyboard Actions" (listing type, paste). The main script editor shows the following code:

```
1 click(  )
2 click(  )
3 type(exists(indicthreads))
4 exists(  )
5 exists(  )
6
```

At the bottom, the "Message" and "Test Trace" tabs are visible. The "Test Trace" tab shows the following log output:

```
[info] Mac OS X utilities loaded.
[info] Text Recognizer inited.
[debug] Google.png exists
[debug] leGoo.png exists
[debug] IndkThreadsS.png exists
```

The status bar at the bottom right indicates "Line: 6, Column: 1".

Sikuli Limitations

- Highly dependant on resolution
- Threshold for image matching can be set
- Cannot run in background

Writing Tests using Sikuli

- Use Java for test Project, Use Junit.
- Use Sikuli IDE or Eclipse IDE
- Write single test to be run on multiple platforms
- Implement interface for Android and Iphone with different set of images required for automation.
- Run same test by changing the configuration on multiple emulators/devices.
- Can see the device on your machine using VNC and use that view to run your test on.

Test framework using Sikuli

- BDD using cucumber
- Integration with Test Link
- Use ESXI Server
- Import the device UI. Screencast for android, VNC server for IOS
- Use Sikuli to run Tests
- Integration with Bamboo

References , Links

- Sikuli - <http://sikuli.org/>
- Perfecto Mobile - <http://www.perfectomobile.com/>
- Device Anywhere - <http://www.deviceanywhere.com/>
- Robotium - <http://code.google.com/p/robotium/>
- FoneMonkey - <http://www.gorillalogic.com/fonemonkey>
- Cucumber - <http://cukes.info/>
- Android Screencast - <http://code.google.com/p/androidscreencast/>

Q & A