INDIUM

Efficiency of Mobile Application Testing with Effectiveness of Tools

Table of Contents

1 Abstract

Android and IOS are the most successful active mobile platforms. Enterprise applications are been released by mobile app developing companies for various mobile handsets on wide range of operating systems, screen sizes and hardware configuration like keypad, trackballs, etc. But it is extremely challenging to verify these.

Many vendors are operating in the enterprise mobility market in today's business environment, to ensure that any given application is isolated, secure and performs well. This helps the end users to focus on preventing virus attacks, security issues, device theft and managing personal data.

Having said that, mobile testing is more challenging than testing based on desktop or Web applications. Hence, testing teams need to find better and more cost-effective solutions to avoid any compromise on quality.

This white paper describes QA challenges, approaches in mobile application testing, mobile testing strategies, mobile testing types and web services testing to validate the business conditions before Mobile UI Development starts. Also, this paper briefly describes about the testing of native and web applications in emulators using automation tools for different platforms like Android and IOS. Record and playback automation tools monkey talk for Android and Test Studio for iOS are efficient tools to reduce manual effort and time consuming especially for Regression testing to meet the actual test estimation, increases productivity and better test coverage. More no of defects identified during regression phase of each release.

2 Why we need

Today’s technology driven business demands a mobile software application to work with diverse array of hardware configurations, operating systems, screen sizes, keystrokes, input methods (Physical and Virtual), menu structure, display properties and mobile network operators.

As QA experts, we need to thoroughly analyse the issues and come up with decision making results, thereby figuring out necessary alterations /additions required to make the application better and more cost-effective solutions.

In many ways, mobile testing is more challenging than testing based on desktop or Web applications. This paper focuses on the QA challenges, strategies and approaches we followed in testing of mobile application on different platforms.

3 The importance

The explosion of mobile internet, with the advent of Smartphones in the market lot of customers are using their mobile phones to access and download numerous mobile applications. With the launch of so many qualifying applications and its content and to make sure that each and every applications is running successfully and performing as expected.

Mobile Application Testing is a process to find out the errors occurred during app development. Testing also ensures that user expectations are met and applications execute properly. It is equally important to conduct device testing to make sure mobile applications perform well across several different platforms and devices.
To determine where we should focus in our testing effort, let’s look at the market share of few of key Manufacturers, operating systems and browsers

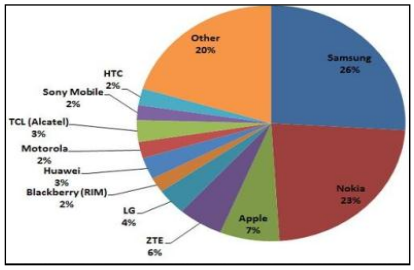


Fig. 1 Top manufacturers

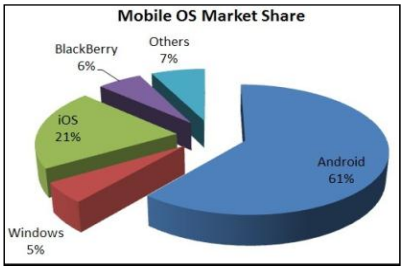


Fig. 2 Mobile OS Market share

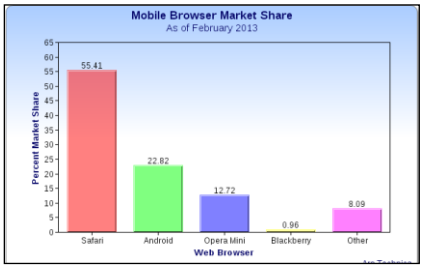


Fig. 3 Mobile Browser Market share in 2012

4 The Checklist

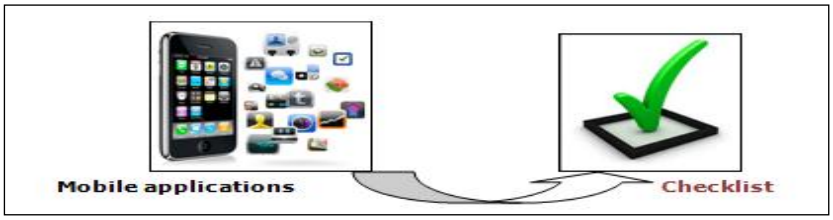


Fig .4 Mobile Checklists

Mobile Application testing should ensure that:

Application is compatible on different devices
Application must behave and respond the same way across different platforms& browsers

Below are the few cases (not limited to) to be verified and validated on any Mobile Application under test

- Installation, un-installation and re-installation
- Check with incoming voice calls, incoming SMS
- Check with mobile charger when it is connected, disconnected.
- Check the application, when device goes to sleep mode and resumes back
- Check the application, when device goes to lock mode and resumes back
- Check the application, when local message is coming from another application like remainders, to-do task
- Check Application start/stop behavior
- Check for Multitasking
- Check for navigations, tabs, page scrolling, buttons like Back, Home, refresh etc...
- Force Close or kill the application
- Check with different networks -- Wi-Fi, 2G, 3G, 4G, offline and resumes back
- Check with Network – Network switching like Wi-Fi to 2G, 2G to 3G etc…
- Check with different network strength like low, medium and high
- Check across different browsers, operating systems, manufacturers
- Check across different screens, resolution, input methods
- Check mobile application is prevented in case it's bigger than the OS allows downloading when connected to networks
- Check for integration, whether the application is correctly connected with different social networks like Facebook, LinkedIn etc…
- Check the memory by filling and emptying it, and then compare the application with it
- Check if any payment gateway occurs like PayPal etc…
- Cosmetic issues (look and feel)
- Check with different battery strength like low, medium and high

5 Types of Mobile applications

Mobile applications can be broadly classified into two different types based on the mobile development. These are native mobile applications and web based mobile applications. Both types can perform similar functions but are inherently different in development.

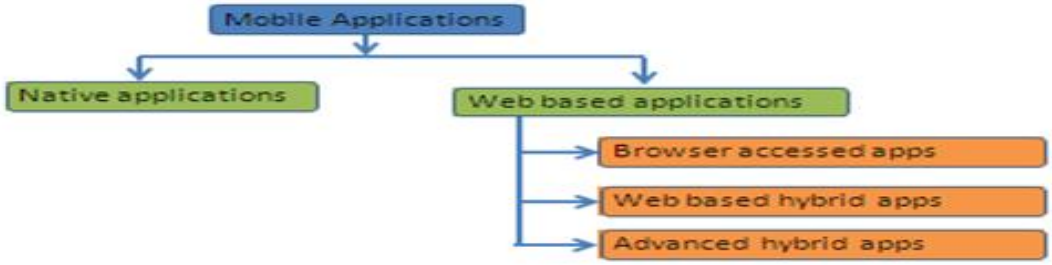


Fig. 5 Types of Mobile Applications

5.1 Native Applications

The native applications are those, which will be installed locally but those which don't send the data on server but directly communication with user. Without network these apps can work in the device. All record related app will be stored in the device itself. Example: Mobile Games

5.2 Web Based Applications

Web based mobile applications can be divided into three major categories based on the communication medium.

5.2.1 Mobile Browser Accessed

Mobile browser applications are not installed in the device. These apps can be used through browser by entering the URL of the Web address. It is completely reliant on the superiority of the browser. All responses come from the server end and displayed in the browser when the URL is hit through device browser. Example: m.yahoo.com, www.google.com

5.2.2 Web Based Hybrid

Web based hybrid applications are installed in the device. It requires complete internet connection to launch and access the applications. Example: Social network applications (Facebook, Twitter, yahoo messenger), E-Commerce and Banking applications

5.2.3 Advanced Hybrid

Advanced hybrid applications can be installed in the device when required we can connect the application to internet. Specific games, which can be used for multiple players and single players in both offline and/or online.

Comparison of Native, Browser and Hybrid applications

Applications	Distribution of Apps	Offline accessibility	Device accessibility (Gyroscope, Accelerometer, Camera, Microphone, GPS and File handling)	Data storage support	UI Support	Performance
Native Apps	App Store/Market	Yes	Full	Yes	Rich	Good
Browser Apps	App Store/Market	No	Full	No	Low	Low
Hybrid Apps	Internet	Yes	Partial	Yes	Medium	Medium

6 Test Approach / Strategy

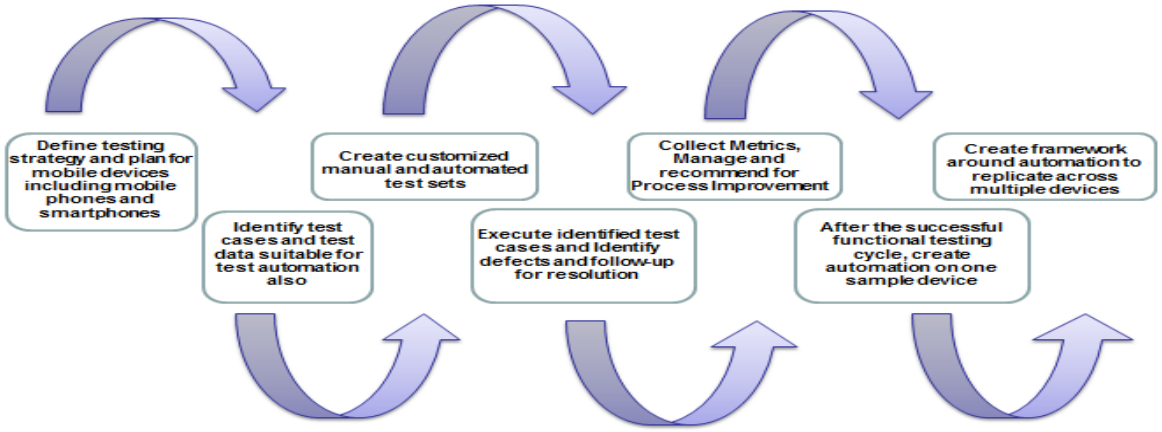


Fig. 6 Mobile Testing Approach

The strategies presented here are intended to highlight some of the areas where the testing of mobile device applications differs from testing desktop or regular web applications. It is important to plan a test strategy that is mobile-specific, or overlook crucial areas of testing like how network connectivity (or lack thereof) affects our application, how screen resolution and orientation changes can spoil a user's whole experience. Most likely we need to use a combination of testing tools and techniques to meet our quality requirements. A good generic strategy to be followed with respect to mobile testing is listed below.

- Have an explicit list of all devices on which QA activities need to be conducted. This basically defines the comprehensive list of devices on which the QA team needs to test the application
- Ensure access to all physical devices
- Factor in additional time to test all modules rigorously on all devices
- Implement a mixed strategy of testing on an emulator and actual devices at various stages of development and deployment.

The QA best practices listed in the forthcoming section are not a comprehensive list, but enlist the major practices followed in the testing of mobile applications.

6.1 Types of Testing for Mobile Applications

6.1.1 Functional Testing

Functional testing ensures that the application is working as per the requirements. Most of the test conducted for this is driven by the user interface and call flows.

6.1.2 API Testing (Unique)

It is essential to verify that the development of each module individually conforms to the API specifications defined for the intended platforms. This ensures that at all stages of testing and deployment of the application, the application does not raise any flags with the deployment guidelines outlined for the application stores by various Original Equipment Manufacturers (OEMs), other controlling organizations, the client, or any other stakeholder.

6.1.3 Integration and Regression Testing

In the integration testing phase, the interactions between the various modules that were developed independently should be tested to ensure a combined functionality on an application level. An incremental, bottom-up approach is recommended in the integration testing phase, in order to identify defects periodically, and allow the development team to address them in batches. This results in a quicker turn-around-time for the testing and bug-fixing, thereby saving the clients' time.

Regression testing implies the testing of features in other modules due to bug-fixing in or updation of a certain module, in order to test the effect of the update on the other modules of the application.
The integration and regression testing phases are also to be carried out using device simulators or emulators in the development environment.

6.1.4 System Testing

System testing tests a completely integrated system in order to verify its end-to-end functionality. Additionally, it ensures that the integrated system does not destroy or corrupt its operating environment, or any other processes that the application interacts with. With respect to mobile applications, it is also necessary to test for functioning specific to various carriers.

System testing should be done in 2 stages. The first stage should make use of automated testing tools. This allows access to a wide variety of devices on which the app can be simulated. Actual testing on devices can also be arranged on a time-shared basis. The second stage is testing the application on actual devices. This phase of testing leads to the final decision regarding the deployment of the app.

6.1.5 Localization testing

As part of the system testing, the localization support provided by the application is also to be tested. Localization support has to be listed by the client as one of the requirements in the initial stages of the application development life cycle. In terms of localization, the ability of the application to support the various languages listed is tested...
Localization testing is usually carried out on actual devices, and is tested on only a handful of critical devices.

6.1.6 Usability testing

Usability testing is carried out to verify if the application is achieving its goals and getting a favourable response from users. This is important as the usability of an application is its key to commercial success.

6.1.7 Performance Testing

Performance testing process is undertaken to check the performance and behaviour of the application under certain conditions such as low battery, bad network coverage, low available memory, simultaneous access to application's server by several users and other conditions. Performance of an application can be affected from two sides: application's server side and client's side.

6.1.8 Compatibility Testing

Compatibility testing is to ensure an app's key functions behave as expected on a specific device. Application compatibility testing on various devices with different screen sizes, resolutions, OS and hardware will increase the quality of application

6.1.9 Memory Leakage Testing

Memory leakage happens when an application is unable to manage the memory it is allocated resulting in poor performance of the application and the overall slowdown of the device performance. As mobile devices have significant constraints of available memory, memory leakage testing is crucial for the proper functioning of an application.

6.1.10 Interrupt Testing

An application while functioning may face several interruptions like incoming calls or network coverage outage and recovery. The different types of interruptions are:

- Incoming and Outgoing SMS and MMS
- Incoming and Outgoing calls
- Incoming Notifications
- Battery Removal
- Cable Insertion and Removal for data transfer
- Network outage and recovery
- Media Player on/off
- Device Power cycle

An application should be able to handle these interruptions by going into a suspended state and resuming afterwards.

6.1.11 Security testing

Security testing can be performed on both client side mobile applications and the server side software to identify the vulnerabilities. Security associated with mobile applications can often be identified and mitigated through security testing.

6.1.12 Installation testing

Certain mobile applications come pre-installed on the device whereas others have to be installed from the store. Installation testing verifies that the installation process goes smoothly without the user having to face any difficulty. This testing process covers installation, updating and uninstalling of an application.

6.1.13 Interface Testing

This covers validation of each screen, buttons, text inputs, navigation flow such as face book, bookmarks and reviews.

6.2 Documentation

A case study is written about every testing project undertaken. This case study outlines the following details about the project.

- Introduction to the project
- Implementation process
- Value-add provided by the project
- Issues and challenges faced while executing the project
- Description of how the challenges were overcome

This documentation serves as a lessons-learnt for future undertakings, and allow the continuous improvement of the testing practices, as well as updation of the best practices adhered to while testing.

7 Testing of Mobile Apps using Web services

7.1 Mobile Application Architecture

Mobile application architecture can be classified into three major blocks, which are user interface, application business logic and Database.

A web service/application service is a combination of programming and data. Web services are made available from business logic includes data flow from data base. It is basically for web based application (browser and hybrid applications). Checking the application functional points and data presented in the application are done by testing the Web services itself in a separate platform.

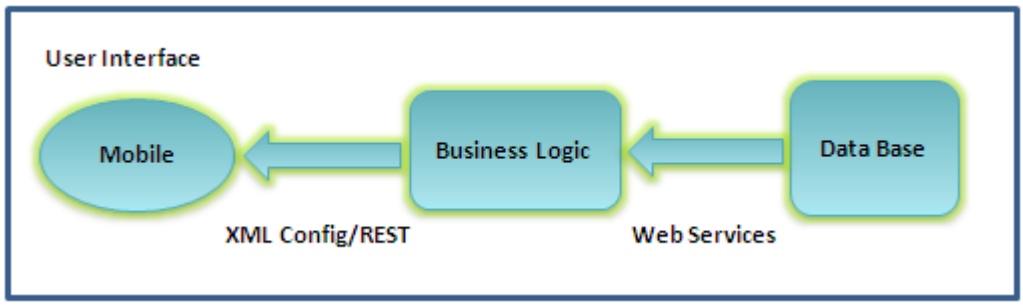


Fig 7. Architecture of Mobile application

7.2 Testing of Web services using REST Client

Advanced REST Client is built on Google Web Toolkit. We can test application functionality and data flows from data storage in REST client by testing the web services. There are several default methods available in REST client to test the web services. Basic knowledge of JSON format is required to test the web services in REST client. Results will be in HTTP messages.

Method	Functions
GET	Fetching data from storage
PUT	Entering data
POST	Submitting data
DELETE	deleting record

7.2.1 GET method

GET method is used to data fetching from data storage that displayed in the mobile UI. We need the following parameter to perform this action:

- Web service URL
- Method - GET
- Headers – Raw input

Fig 8. Testing of GET method in REST

7.2.2 POST method

POST method is used to submit the entered JSON formatted input to get the output in HTTP message format. We need the following parameter to perform this action:

- Web service URL
- Method - POST
- Headers – Raw input
- Body – Raw JSON input
- Content type – application/json

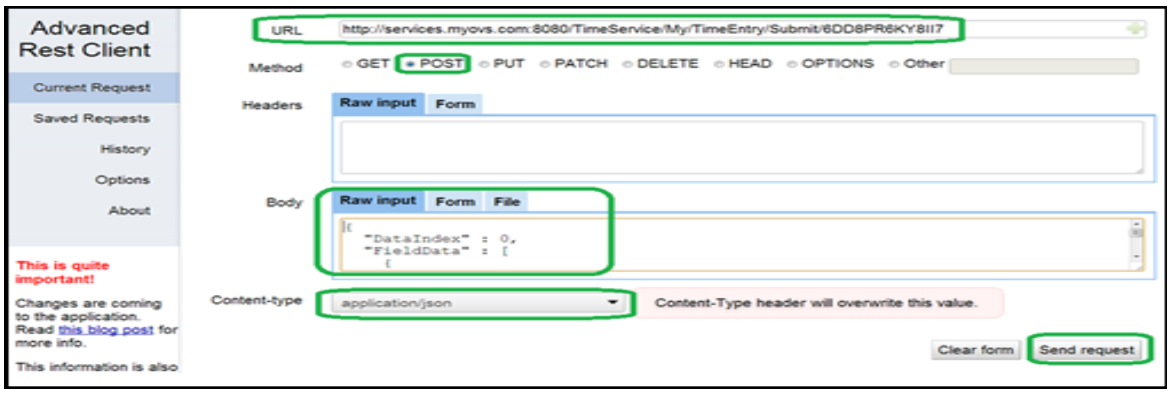


Fig 9. Testing of POST method in REST

The output result will be in HTTP server message for the request from REST client.

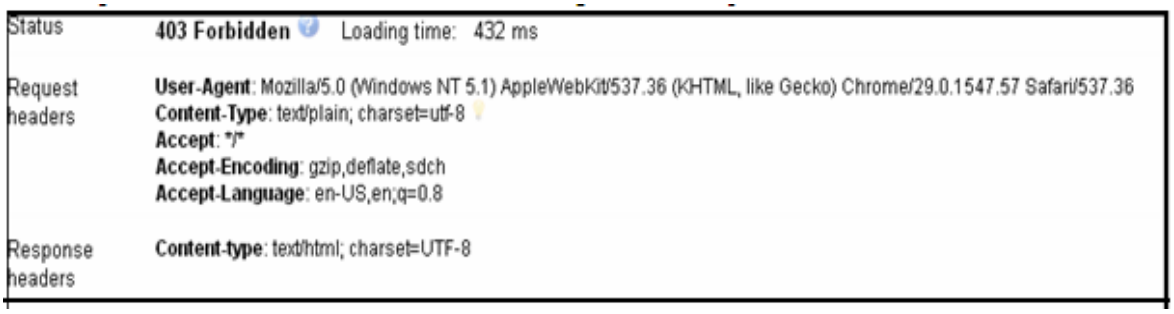


Fig 10. Result in HTTP message format

List of HTTP Status message:

No	HTTP Message	Description
1	1xx	Information
2	2xx	Successful
3	3xx	Redirection
4	4xx	Client Error
5	5xx	Server Error

8 Testing of Application using Automation Tools

Test automation is the use of software to automate and control the setting up of test preconditions, Execution of tests, test control and test reporting functions, with minimal user intervention. It is not practical to try to automate everything, especially for mobile. Supports –Unit, UI and Functional testing

There are multiple tools available in the market to automate the mobile applications:

Name of the Tool	Tool Scope	Technology	Language Supports	Browser supports
Monkey Talk	Testing Framework, Testing Unit	HTML, JavaScript	C#, JavaScript	Chrome, Firefox, IE, Safari
Telerik Test Studio	Testing Tool, Testing Framework, Test Management	AJAX, ASP, HTML JavaScript, Silverlight	C#, JavaScript	Chrome, Firefox, IE, Safari
Robotium Test Framework	Testing Framework, Testing Tool	Android apps, Mobile	JavaScript	Chrome, Firefox, IE
SeeTestMobile	Testing Tool, Testing Framework, Test Management	Android apps, Blackberry apps, iPhone apps, Palm apps, Symbian apps, Windows Mobile apps	C#, Java, Python, VBScript	Chrome, Firefox, IE, Safari
Silk Performer	Testing Tool	Adobe Flash, Adobe Flex, AJAX, Silverlight, Web, Web2.	Visual Design	Chrome, Firefox, IE, Safari

Cucumber	Testing Framework, Testing Unit	HTTP, Ruby, Web	Dot.Net, Java, Python, Ruby	Chrome, Firefox, IE, Safari
EggPlant	Monitoring Tool, Testing Unit	AJAX, Android apps, DHTML, Dot.NET, Adobe Flash, Adobe Flex, HTML, iPhone apps, Java App	Visual Design	Chrome, Firefox, IE, Safari
Rancour Test Automation	Testing Tool, Testing Utility, Testing Framework	ActiveX, Adobe Air, Adobe Flash, Adobe Flex,AJAX,DHTML,DHTMLX,Dot.NET, Ext.Net,GWT,HTML,HTML5,Infragistics,iPhone apps, Java	C#,VB.Net	Chrome, Firefox, IE, Safari
Mobile cloud	Testing Tool, Testing Framework, Cloud Service	Android apps, Blackberry apps, iPhone apps, Symbian apps, Windows Mobile apps		C Chrome, Firefox, IE, Safari
Neo load	Testing Tool	AJAX, Adobe Flex, GWT, Silverlight		Chrome, Firefox, IE, Safari

8.1 MonkeyTalk (For ANDROID)

Monkey Talk is the leading tool for automated testing of Android applications. The Robust and cross platform tool is used to record and playback the scripts. Test Native, Web or Hybrid apps on Android emulators or real devices. Monkey Talk supports simple smoke tests to sophisticated data driven test suites.

8.1.1 About Monkey Talk

- The MonkeyTalk platform consists of two primary compounds: MonkeyTalk IDE and MonkeyTalk Agents
- MonkeyTalk IDE is Eclipse based tool that records, plays, edits, and manages functional test suites for Android applications running on simulators, emulators, and devices
- MoneyTalk Agents are libraries for Android that must be linked into applications to be tested and both the device and computer to be in the same network
- The IDE communicates with the app over HTTP via a USB tether or Wi-Fi connection. You can record and play back tests against apps running on your local computer or network, or remotely over the internet

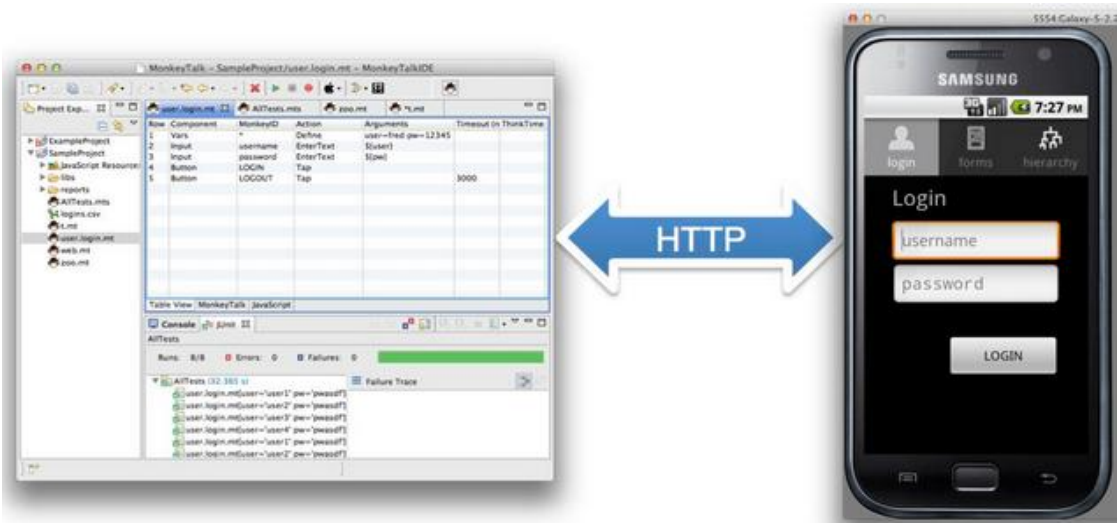
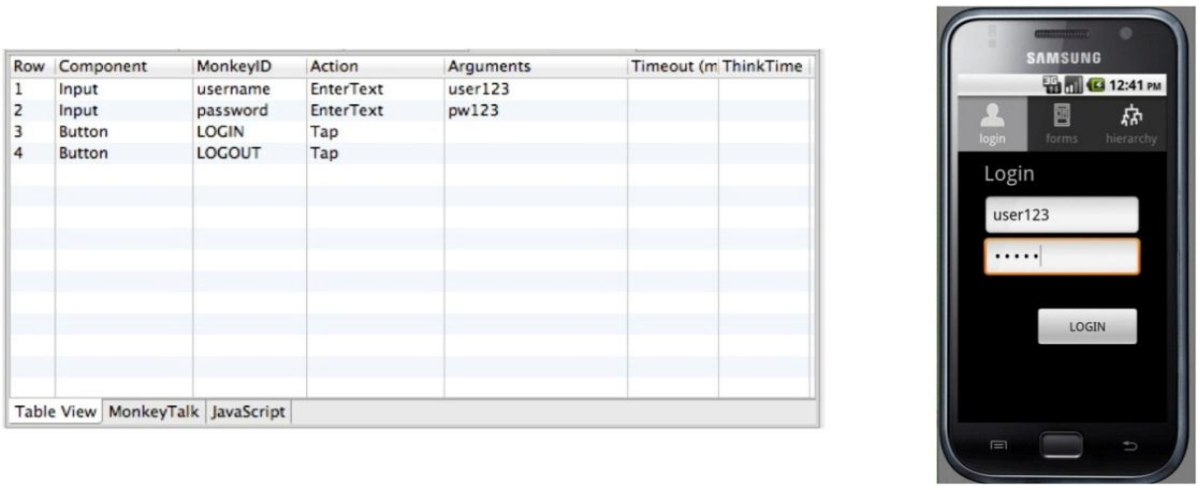


Fig 11. Connection to Monkey Talk IDE with Device

- Device sends the commands to IDE and saves in Table view as well as the recording scripts while recording
- The Monkey Talk scripts can be verified (time delay…), edited through table view / Monkey talk view
- While playback the scripts the IDE will send the commands with inputs to device/simulator to run the scripts
- The agents enable applications to record and play MonkeyTalk Commands
- Each failed command can be retrieved with maximum time to wait for a command to succeed
- MonkeyTalk provides functional testing, regression testing also extend the tool to write frameworks
- We can create JAVA script from MonkeyTalk in Monkey Talk IDE
- Updated scripts will be used for regression testing

8.1.2 Recording - Monkey Talk

- Connect the Device
- Pull up the application on the device
- Click 'Record' on the MonkeyTalk IDE.
- Record button need to be enabled to start recording
- Interact with the application; Monkey Talk should record each action as a new command in the table.
- Click 'Stop' button to stop the scripts



Row	Component	MonkeyID	Action	Arguments	Timeout (m	ThinkTime
1	Input	username	EnterText	user123		
2	Input	password	EnterText	pw123		
3	Button	LOGIN	Tap			
4	Button	LOGOUT	Tap			

Table View | MonkeyTalk | JavaScript

Fig 12. Recording Monkey talk scripts

8.1.3 Pros and Cons

Pros	Cons
- Monkey Runner can be used for testing on an actual device without any need to export screens to the desktop - Monkey Runner can take screenshots for offline analysis. Screenshots can also be used for doing automated image comparisons against reference images using an inbuilt functionality or using external tools. - Monkey Runner can be extended using a plug-in architecture - Monkey Runner can be used to control multiple devices at the same time	- Require some knowledge of the application code; for example the user need to know the name of the activities and views - Monkey Runner is a low level API based tool – there is no GUI/IDE that is provided along with it by default

8.2 Test Studio (For iOS)

Test Studio provides an iOS application to manage application testing. This iOS application is free and can be downloaded from the Apple App store. In order to test a native application, the user needs to download a testing extension from www.telerik.com/ios-testing

8.2.1 About Test Studio

- Test Studio by Telerik allows developers to record and execute automated tests in mobile applications and websites
- Test Studio for iOS is an automated and ad-hoc testing solution for iPhone, iPod Touch, and iPod
- Record and playback advanced reliable automated tests of mobile apps
- Test Studio can test native as well as HTML5 applications (Web and Hybrid)
- Test Studio generates queries capable of finding elements based on properties unique to them not use image based element detection
- Quires related to properties unique is helpful to locate the element on the screen if the development changes the location
- The Test Studio scripts can be verified (time delay…) and editable. Refer Fig 13. Adding Verification step

- Used to test all UI Kit based iOS control, gestures (Swipes, Zooms, taps etc...)
- Modified scripts will be used for regression testing

8.2.2 Recording - Test Studio

- Connect the Device
- Compile the native app with Test studio static library to interact the Test Studio with the target application in device
- Go to ‘App testing ‘for Native application testing
- Click the target application
- Interact with the application, Test Studio should record each action as a validation
- Click 'Stop' button to stop the scripts

Fig 13. Adding Verification step

Fig 14. Recording Test Studio script

8.2.3 Pros and Cons

Pros	Cons
- Does not require scripting knowledge for test development. It allows test cases to be recorded with required actions specified for each object - Does not require a jail broken device - Provides a consolidated report for all the executed test cases - Detects elements by their properties not location/image	- Still in beta phase (as of Jan 2013), not very stable - Adding other plug-ins (Test Studio's load and performance) is not possible as of now Commercial tool

9 Challenges

The primary factor that determines an automation tool's success is its ability to work across platforms and technology stacks. The following challenges influence automation success:

- Device Diversity:
  - Multiple platforms and browsers
  - Rendering differences
  - Mobile devices with varied application run times
  - Varying screen resolution
  - Different platform
  - Different pixel density and resolution
  - Different input methods like QWERTY, touch
  - Different OS versions of the same platform
- Network Challenges:
  - Multiple network types (e.g., GSM/GPRS/Wi-Fi/Wi-Max)
  - Different speeds of connectivity across geographies
  - Multiple network operators with customized network features
- Hardware Challenges:
  - Limitations in processing speed
  - Limitations of mobile memory size
  - Differences in device communication protocols (e.g., WAP/HTTP)

10 Benefits

- Web Services testing
  - ✓ Interoperability - Web Services are virtually platform-independent
  - ✓ To test the application each request Response time
  - ✓ Field level validations and functionality of the form can be tested in web services testing itself
  - ✓ Reusability
  - ✓ Verify that correct value pair are supported by the service

- Automation testing

  - ✓ Define most efficient mobile strategies and goals
  - ✓ Reduce application testing cost and accelerate time to market
  - ✓ Improve end-user experience of mobile applications
  - ✓ Improve quality of applications by providing more robust coverage
  - ✓ Test more within budget and help meet product deadline

11 Conclusion

We can't deny the importance of a streamlined mobile application testing strategy in the success of a mobile app. Despite that, the mere presence of a testing strategy does not ensure the quality and performance of a mobile app.

The strategic selection of target devices, and a right choice of emulators, physical devices, and testing methodologies, before creating a mobile app testing strategy, will go a long way in delivering desired results.

All these, combined with testing best practices and industry standards, can help to overcome the obstacles of mobile application testing, and build remarkable mobile apps for the customers.

12 References

http://www.digitaltrends.com/mobile/samsung-nokia-apple-phone-sales-q2-2012/
http://jitenderarora.co.uk/mobile-security-strategy-past-present-future/
http://www.pcper.com/news/General-Tech/Internet-Explorer-Still-Most-Popular-Web-Browser-2013

13 Appendix

REST	REST Web services
JSON	Java Script Object Notification
API	Android Package Interface
IDE	Integrated Developer Environment

About Indium:

Indium Software is exclusively focused independent software testing services firm since 1999. Over the years, Indium mastered objective methods that minimize the risk of failure of applications and software products. With a global headcount of over 300 employees, Indium works for a mix of marquee Enterprise and ISV clients spread across the globe. Indium is aggressively pursuing the Social, Mobile & Cloud agenda to make these the core of our next wave of service specialization.