mCraft 2.0 Mobile Automation Framework – Handbook

Mobile Testing CoE

11/20/2013

Table of Contents

1. Purpose 3
2. Pre-requisite 3
3. mCraft 2.O – An Introduction 3
4. mCraft 2.O – Architecture 4
   4.1 Tools Layer 5
   4.2 Application Layer 6
   4.3 Test Design Manager 6
      4.3.1 Global Settings 6
      4.3.2 Test Scripts Design 7
      4.3.3 Test data 8
      4.3.4 Support Libraries 9
      4.3.5 Functional Libraries 10
   4.4 Test Execution Manager 10
      4.4.1 Run Manager 10
      4.4.2 Allocator 11
      4.4.3 Device Engine 11
      4.4.4 Serialization 11
   4.5 Test Reports 12
5. mCraft2.O – Execution Flow 13
6. mCraft2.O – Folder Structure 14
7. mCraft2.O – Key considerations for implementation 15
8. Contacts 15

1. Purpose

The purpose of this document is to provide a detailed overview of Cognizant Reusable Automation Framework for Mobile Application Testing.

2. Pre-requisite

The user should have some basic knowledge on Mobile Application Testing, about devices and emulators/simulators.

3. mCraft 2.O – An Introduction

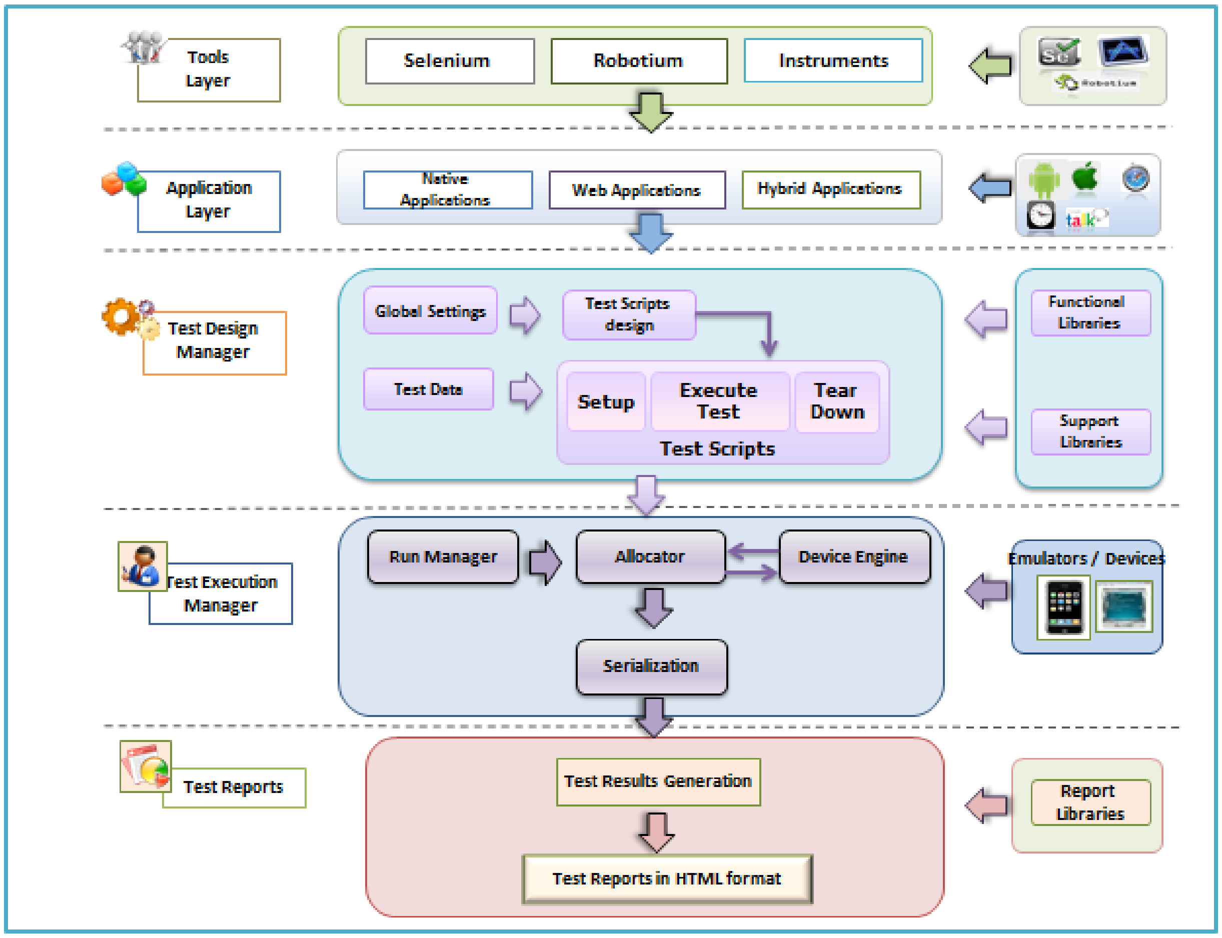
What?

- mCraft 2.O - Cognizant Reusable Automation Framework for Testing - is Cognizant's in-house automation framework, proposed to ease the script development and maintenance effort in test automation
- mCraft 2.O follows a hybrid-driven approach, combining the best practices of business keyword-driven and data-driven approaches
- mCraft 2.O is tool agnostic, which means that the mCraft 2.O design principles may be implemented over any automation tool like Selenium, Robotium, Instruments

4. mCraft 2.O – Architecture

The following diagram presents a detailed conceptual model of mCraft 2.O, and portrays clearly the various modules within the mCraft 2.O architecture. Each of these modules is subsequently explained in detail, in terms of the basic usage of the module (What), the benefits of using the module (Why).

Figure 1: Schematic representation of mCraft 2.O



4.1 Tools Layer

As mentioned earlier, mCraft 2.O supports the framework for open source tools like Selenium, Robotium and Instruments.

Selenium

- Selenium is an open source tool developed for web applications.
- It supports both Android & iOS platform.
- It supports object based scripting on web applications across platforms/devices.
- It supports Object based automation testing on Rooted/Non-rooted android devices and Emulator/Simulator.

Robotium

- Robotium is another open source tool developed by Google.
- Initially the tool was developed only for Android Native applications.
- The latest version of this framework supports both Native & Hybrid applications.
- It supports Object based automation testing on Rooted/Non-rooted android devices and Emulator/Simulator.

Instruments

- Instruments is developed by Apple, and it is designed only for iOS –Native applications.
- It is a performance, analysis and testing tool for dynamically tracing and profiting OS X and iOS Code.
- It is a flexible and powerful tool that lets you track one or more processes and examine the collected data.
- It supports Object based automation testing on Non-Jailbroken/Jail broken iOS devices and simulators.

4.2 Application Layer

Application Layer depicts the supported types of applications by the mCraft framework 2.O.

Native Application
Apps developed exclusively for a specific mobile platform that can leverage all device capabilities. Both Robotium and Instruments supports Native applications on Android & iOS platform respectively.
Web Application
A mobile web app is a type of mobile application that combines the versatility of the web with the functionality of touch-enabled devices. Selenium supports Web apps across platforms.
Hybrid Application
Hybrid apps, like native apps, run on the device, and are written with web technologies (HTML5, CSS and JavaScript).Robotium supports hybrid apps in Android platform.

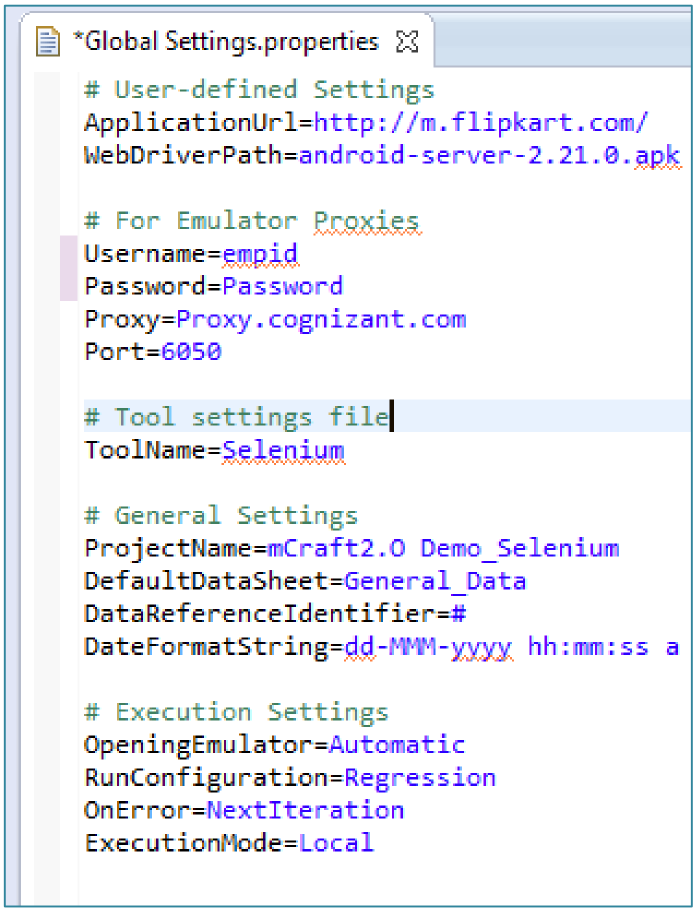
4.3 Test Design Manager

mCraft 2.O framework simplifies the test design process and test script management.

4.3.1 Global Settings

As the name suggests, this is a place where user specifies the settings for the test cases. User can edit Emulator settings, Tool selection, General settings, and Execution settings and also feed the Application URL (if it is a web app).

Figure 2: Global Settings -mCraft 2.O



*Global Settings.properties

```
# User-defined Settings
ApplicationUrl=http://m.flipkart.com/
WebDriverPath=android-server-2.21.0.apk

# For Emulator Proxies
Username=empid
Password=Password
Proxy=Proxy.cognizant.com
Port=6050

# Tool settings file
ToolName=Selenium

# General Settings
ProjectName=mCraft2.O Demo_Selenium
DefaultDataSheet=General_Data
DataReferenceIdentifier=#
DateFormatString=dd-MMM-yyyy hh:mm:ss a

# Execution Settings
OpeningEmulator=Automatic
RunConfiguration=Regression
OnError=NextIteration
ExecutionMode=Local
```

4.3.2 Test Scripts Design

In this phase, user can design their test scripts and also define the functions (Setup, Execute Test, and Tear Down).

Setup

User can add the functional library, set the driver etc.

Execute Test

User can design their own scripting for their application

Tear Down

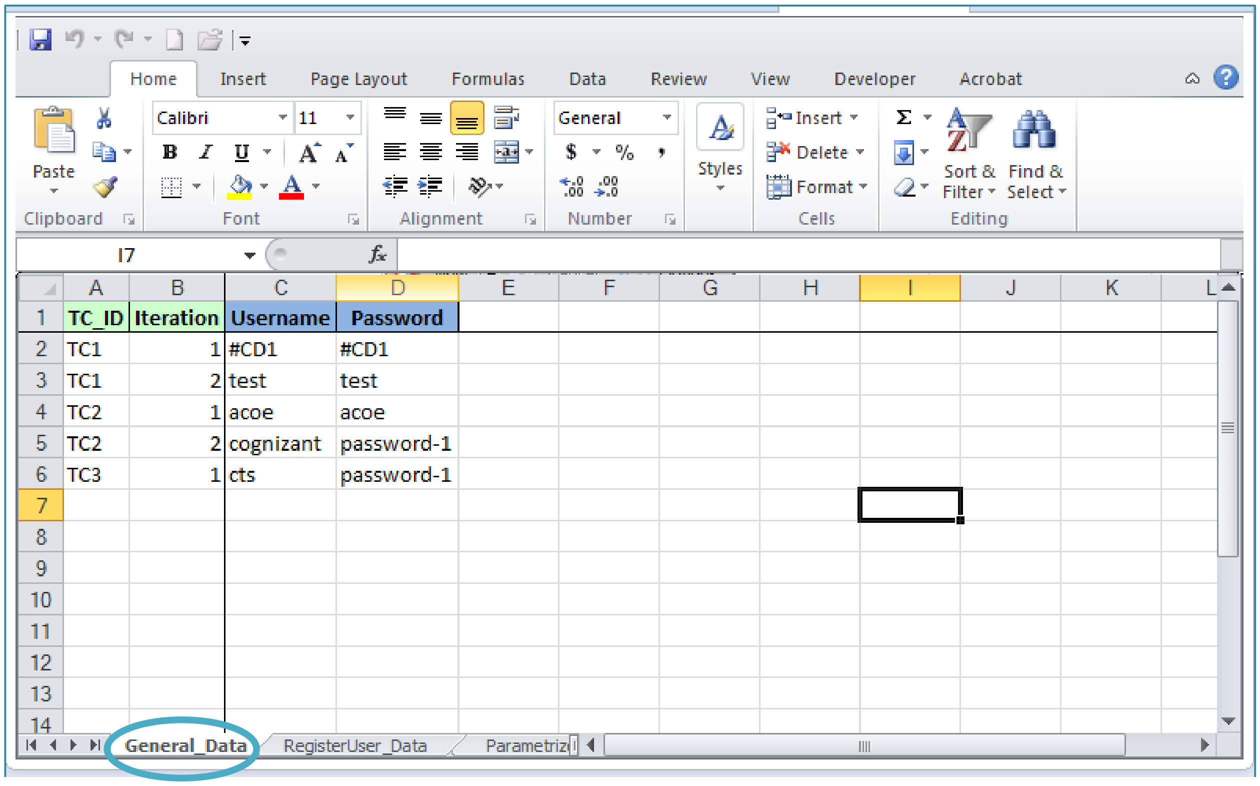
It will consolidate the test report for the test case/test cases execution

4.3.3 Test data

The test data is nothing but a table (or a set of tables) containing the data inputs required by the test script to test the application. This helps to implement sound principles of automation script design by separating the data from the scripts.

mCraft 2.O supports configuration of the test data to facilitate organization of the data into more manageable blocks. mCraft 2.O is also designed to handle multiple **Iterations** of test data mapped against a single test case. This helps testing of the same test script with different sets of data to cover all possible scenarios.

Figure 3: Sample test data sheet of mCraft2.O



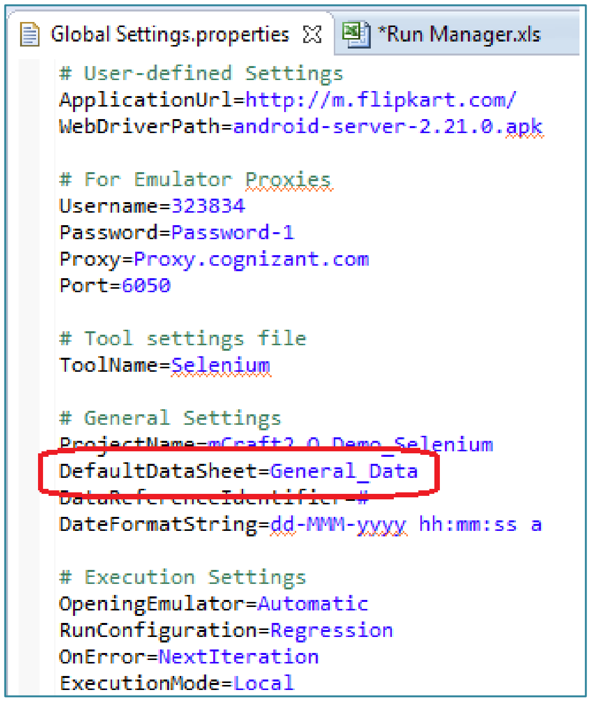
TC_ID	Iteration	Username	Password
TC1	1	#CD1	#CD1
TC1	2	test	test
TC2	1	acoe	acoe
TC2	2	cognizant	password-1
TC3	1	cts	password-1

Fig 3 (above) shows a sample test data sheet. The first column contains the **Test case** of the test case, for which the test data is mapped. The second column indicates the appropriate **Description** of test data. The third column indicates the **Execute** and the fourth column indicates the **Iteration mode**. The user has to take care to maintain these columns accurately, in order to avoid unexpected errors.

General Data is the sheet name where you can feed the general test data (Login). User can also feed the test data in the other two sheets or else they can create the new sheet where they can feed the test data for the script execution.

User has to mention which sheet needs to be referred in **Global Settings** for test data during execution.

Figure 4: Data sheet representation of mCraft2.O



```
# User-defined Settings
ApplicationUrl=http://m.flipkart.com/
WebDriverPath=android-server-2.21.0.apk

# For Emulator Proxies
Username=323834
Password=Password-1
Proxy=Proxy.cognizant.com
Port=6050

# Tool settings file
ToolName=Selenium

# General Settings
ProjectName=mCraft2.0_Demo_Selenium
DefaultDataSheet=General_Data
DataReferenceIdentifier=#
DateFormatString=dd-MMM-yyyy hh:mm:ss a

# Execution Settings
OpeningEmulator=Automatic
RunConfiguration=Regression
OnError=NextIteration
ExecutionMode=Local
```

4.3.4 Support Libraries

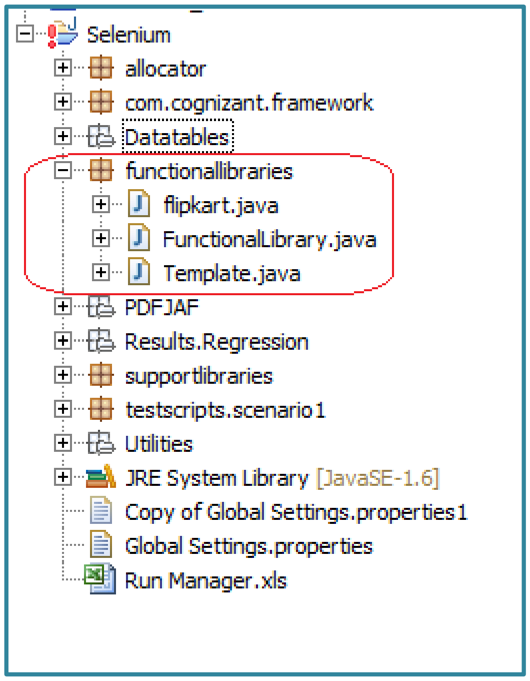
These are fully reusable functions that are **tool-specific** and **independent of the AUT**. They include functions for tool selection, mobile platform, script helper etc.

The support libraries are the backbone of the framework, providing support for most of the features. The support functions utilize a considerable amount of time if developed for each and every project individually. Maintaining these as reusable functions in a repository helps to decrease the scripting time drastically.

4.3.5 Functional Libraries

This library contains reusable functions for Application and tool as well.

Figure 5: Folder Structure of Function Libraries in mCraft2.O



4.4 Test Execution Manager

In this phase, the execution of the test script will start from Run Manager, which in turn calls Allocator. Allocator starts the Driver Script which communicates with Device Engine for device selection and serialization will launch the emulator / device and do the port forwarding and launching the web driver etc.

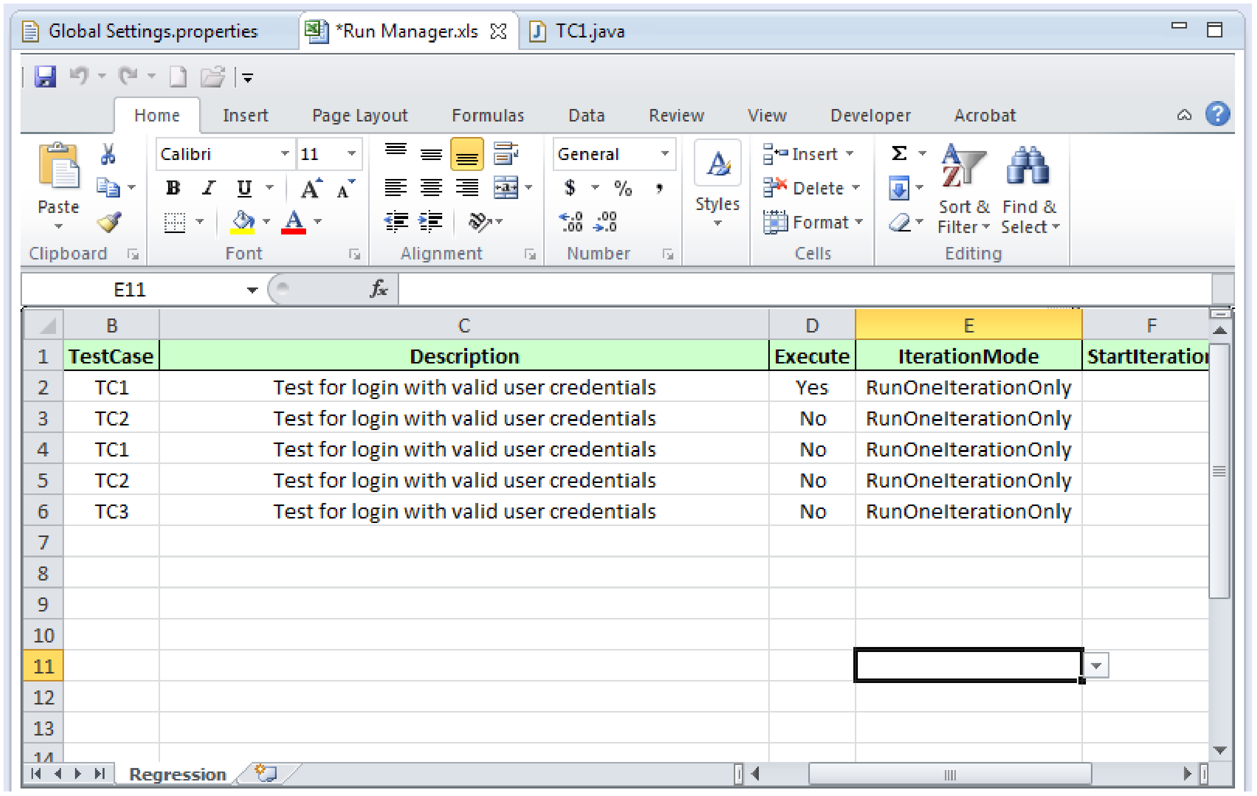
4.4.1 Run Manager

Run Manager starts the test execution and it will start the Allocator.

- Easy flagging of test cases for execution
- One click execution of flagged test cases

- Integrates with Windows scheduler for unattended execution and MS Outlook for mail triggered execution

Figure 6: Sample Run Manager sheet of C.R.A.F.T



	B	C	D	E	F
1	TestCase	Description	Execute	IterationMode	StartIteratio
2	TC1	Test for login with valid user credentials	Yes	RunOneIterationOnly	
3	TC2	Test for login with valid user credentials	No	RunOneIterationOnly	
4	TC1	Test for login with valid user credentials	No	RunOneIterationOnly	
5	TC2	Test for login with valid user credentials	No	RunOneIterationOnly	
6	TC3	Test for login with valid user credentials	No	RunOneIterationOnly	
7					
8					
9					
10					
11					
12					
13					

4.4.2 Allocator

- Allocator starts the script execution by triggering the Driver script.
- Driver script starts the Device Engine which displays the popup for device selection with the available devices or Emulators/Simulators.

4.4.3 Device Engine

- Device Engine communicates with the Allocator to select the required device or emulator/Simulator selection.

4.4.4 Serialization

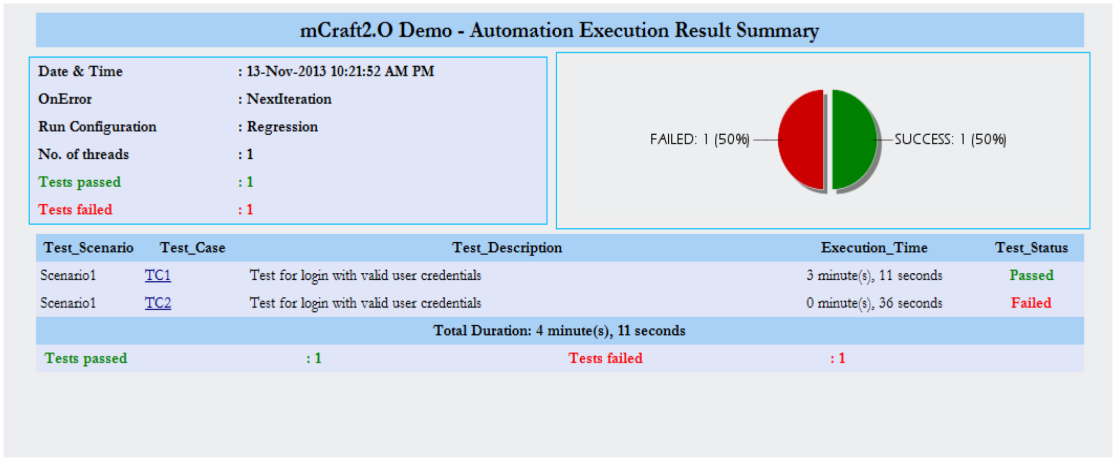
- It will launch the selected device or Emulator/Simulator
- It will install and launch the web driver
- It will launch the application in the web driver

4.5 Test Reports

One of the key differentiators in mCraft2.O is the provision to provide customized test reports, over and above the results generated by the automation tool. The customized results are available in pdf and HTML formats, and are generated with the help of the support libraries.

- Easy to understand, business facing results
- Format and content can be modified as per customer requirements
- There is no need for the automation tool to be installed to view the test results. Test results may be easily shared within the entire team.
- The results are automatically stored with a timestamp indicating the date and time of execution, for easy reference.

Figure 7: Sample detailed test case results of mCraft2.O



mCraft2.O Demo - Automation Execution Result Summary

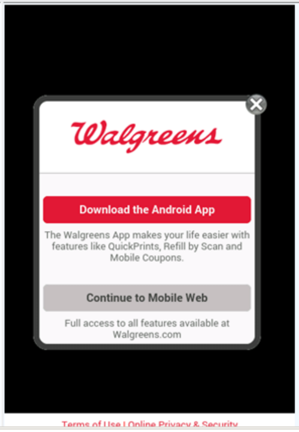
Date & Time	: 13-Nov-2013 10:21:52 AM PM
OnError	: NextIteration
Run Configuration	: Regression
No. of threads	: 1
Tests passed	: 1
Tests failed	: 1

Test_Scenario	Test_Case	Test_Description	Execution_Time	Test_Status
Scenario1	TC1	Test for login with valid user credentials	3 minute(s), 11 seconds	Passed
Scenario1	TC2	Test for login with valid user credentials	0 minute(s), 36 seconds	Failed

Total Duration: 4 minute(s), 11 seconds

Tests passed : 1 Tests failed : 1

Execution Tool	Selenium	Project Name	Scenario1
Test Case Name	TC1	Mobile OS	Android
Platform	WINDOWS	Iteration Mode	RunOneIterationOnly
Start Iteration	1	End Iteration	1
Start Time	13-Nov-2013 10:25:02 AM	End Time	1
Test Case Status	Passed		

S.No	TIMESTAMP	ACTION	STATUS	SCREENSHOT
1	13-Nov-2013 10:25:10 AM	clicked Continue to Mobile Web	PASS	

Walgreens

Download the Android App

The Walgreens App makes your life easier with features like QuickPrints, Refill by Scan and Mobile Coupons.

Continue to Mobile Web

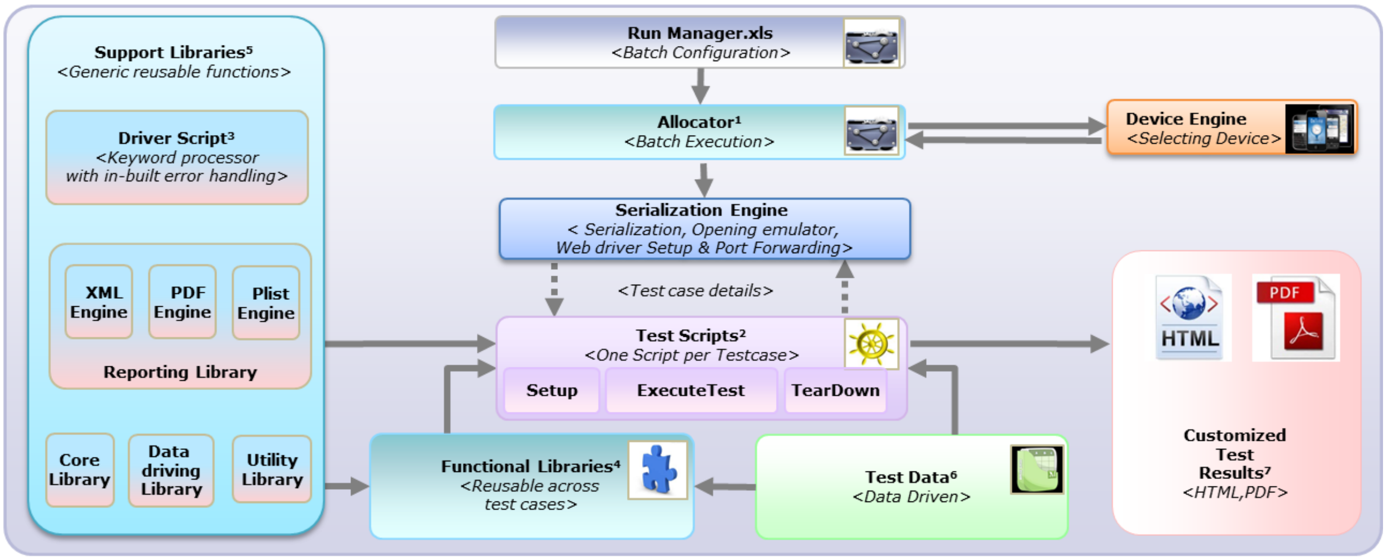
Full access to all features available at Walgreens.com

Terms of Use | Online Privacy & Security

5. mCraft2.O – Execution Flow

Referring to the architecture diagram of mCraft2.O (reproduced below for reference), the execution flow may be explained as follows (The step numbers used correspond to the numbering present within the architecture diagram):

Figure 8: Execution Flow of mCraft2.O

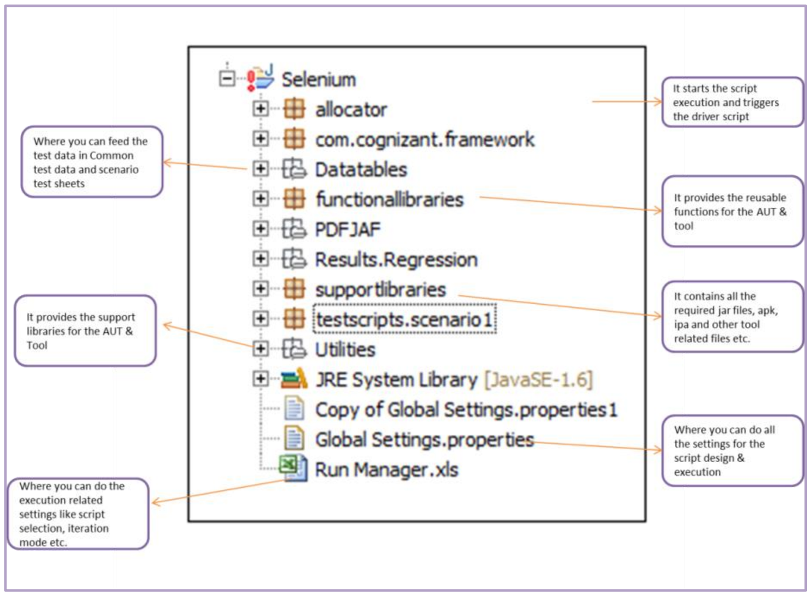


1. The **Run Manager** is the entry point of the execution
2. The **Run Manager** identifies the test cases to be executed and the number of iterations on the required devices/emulator and it will handle the execution flow to **Allocator**.
3. **Allocator** in turns start the execution and it will trigger the **Driver script** and **Device Engine**.
4. The **Driver script** passes control to the relevant **Test script** and **Device Engine** will pop up the available emulators/devices.
5. The **Driver script** passes the control to **Serialization** which opens emulator, setup the web driver and do the port forwarding automatically.
6. The **Test scripts** which uses the **Functional/support libraries** for execution.
7. **Test script** also gets the Test Data from the **Datatable**.
8. **Test reports** will be exported in html format using **Reports Library**.

6. mCraft2.O – Folder Structure

A sample folder structure for the framework is depicted below (Selenium tool is selected here)

Figure 9: mCraft2.O Folder Structure


Selenium
allocator
com.cognizant.framework
Datatables
functionallibraries
PDFJAF
Results.Regression
supportlibraries
testscripts.scenario1
Utilities
JRE System Library [JavaSE-1.6]
Copy of Global Settings.properties1
Global Settings.properties
Run Manager.xls
It starts the script execution and triggers the driver script
Where you can feed the test data in Common test data and scenario test sheets
It provides the reusable functions for the AUT & tool
It contains all the required jar files, apk, ipa and other tool related files etc.
It provides the support libraries for the AUT & Tool
Where you can do all the settings for the script design & execution
Where you can do the execution related settings like script selection, iteration mode etc.

The contents of the different folders are explained below:

Folder	Contents
Allocator	It starts the script execution and triggers the driver script
Datatables	Contains the hybrid-driven data component of mCraft2.O in the form of Excel sheet.
Functional Libraries	Contains the reusable functions for both AUT & tools
Driver Script	Contains the Driver script.
Support Library	Contains the support libraries.
Utilities	Contains all the required jars, apk, ipa and other related files and folders for AUT & Tools
Global Settings	Where you can do all the settings related to emulator proxies, application url, tool selection etc.
Support Libraries	Contains the support libraries.

7. mCraft2.O – Key considerations for Implementation

Listed below are some of the important factors to be considered while implementing mCraft2.O:

- ✓ Minimal Learning Curve
- ✓ Technical Expertise not required for coding
- ✓ Serialized execution in multiple devices
- ✓ Easy ROI

8. Contacts

- MobileTestingCoEQA@cognizant.com