Sencha Support

Table of Contents

Introduction

Sencha Touch

Sencha Touch is a user interface (UI) JavaScript library, or framework, specifically built for the Mobile Web. It can be used by Web developers to develop user interfaces for mobile web applications that look and feel like native applications on supported mobile devices. It is fully based on web standards such as HTML5, CSS3 and JavaScript. Sencha Touch aims to enable developers to quickly and easily create HTML5 based mobile apps that work on Android, iOS and Blackberry devices, and produce a native-app-like experience inside a browser.*(Reference - Wikipedia)*

Testing by object recognition

Testing based on Object oriented recognition benefits from tight integration with the programming platform. Tools based on this principle understand the underlying technology of the platform for which application under test is developed and they are able to identify individual GUI components, read their properties and interact with them by means of invoking methods on them. They are able to handle test verification steps such as "verifying that there are two buttons labeled 'ok' and 'not ok' or "finding out whether the 'ok' button is enabled or disabled" or "finding out the number of items in a list and selecting a particular item from the list using the text of the item or the index".

Tools based on principle of using object recognition typically invoke actions on the GUI components through the APIs (Application Programming Interfaces). Checkpoints in the test cases are generally based on checking the Runtime values of the GUI components and matching those values with the expected values according to the functional scenario.

Jamo Solutions provides the unique solution with object recognition on different mobile devices and on multiple platforms including the Sencha Touch platform. r, The QA engineer has direct access to all objects, their properties and methods of Sencha Touch. If the QA engineer does not have this access, he is bound to test the by Sencha created HTML, ending up analyzing complex HTML structures which are prone to change with every new SenchaTouch release.

Object oriented testing approach prevails among automated testing tools today for following advantages:

- Invokes action on GUI component through APIs and hence developing test cases becomes easier.
- Robust test scripts thanks to ability to identify properties and invoke actions of individual GUI components.
- Easy to understand for QA engineers who are familiar with similar tools as the Windows PC platform.

Testing By Image Based/OCR(optical character recognition)

Testing tools based on Image recognition principle typically automate on the level of operating system, on top of a layer which allows access to all the necessary devices (such remote access software). Automation is done through pointer events injected into the layer allowing access to the Device and eventually Application under test. Application Under Test is accessed through the system display buffer on the pixel level. Checkpoints are based on Image verification and OCR comparisons. This technology of testing can NOT handle test steps such as "Verification of two buttons labeled 'ok' and 'not ok' or "find out whether the 'ok' button exists because here OCR or image recognition may find text 'ok' twice and will give some unexpected results" .
Image oriented testing approach does not prevail among automated testing tools today for following disadvantages:

- No access to the application GUI components. Method to interact with the UI components is to save images of these components during the test design phase and then use them as input to the image recognition algorithm to navigate and interact with the similar images found in Application and verify the result during test execution.
- Image and OCR based check points are not sufficient for exhaustive functional testing.
- High Vulnerability to test environment changes. A minor change in mobile background color or theme may cause image search failure Many times image recognition fails when the UI component gets highlighted during test execution and image repository contains a non highlighted image of that component.
- Test Suite does not easily adapts to the resolution changes or other changes in the application runtime environment which impact the pixel map of the components in the application.

Advantages of Sencha support for Test Automation

1. Precise and clear object repository

The tool with Sencha support recognizes the objects as Sencha elements instead of HTML tags making the object repository more precise and readable. You can easily see from the object repository if a certain object is a button, or a list or an imageView. More detailed information is introduced in the case study part of this document.

2. Robust and non-brittle test cases.

If Sencha releases a new revision or version, the generated HTML can change, but the script does not needs to be maintained because our test tool recognize the object by the property of the Sencha objects rather than the property of the HTML element. As a result, frequent changes in the Application code - HTML code does not lead to frequent update of object repository the object repository, as the object repository is based on Sencha objects and not on HTML tags.

3. Reusable and flexible scripts

As the object repository is based on Sencha objects recognition, scripts developed for portrait mode can be reused for landscape mode, the test script can also be reused for different mobile platforms and different version of mobile OS. This feature is important since the Sencha based UI can display different information when in portrait or landscape mode.

4. Easier implementation of test cases

When a QA engineer is writing the test cases for a web app, with the Sencha support from Jamo Solutions, he does not have to write the script for clicking all kinds of HTML element. The QA engineer can just access the Sencha table, Sencha list and so on, just like testing a native application.

Case study

Let us use a sample app to demonstrate the convenience of Sencha support from Jamo Solutions.

The sample web app Kitchen Sink is provided by Sencha(On Google chrome browser) at http://dev.Sencha.com/deploy/touch/examples/production/kitchensink/

The application looks like below figure (Figure 1) when navigating to the link provided above and choosing the 'User Interface' in the home window.

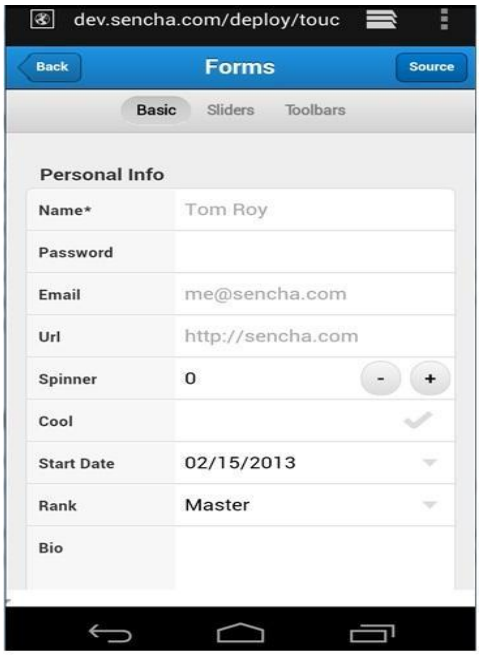

Figure 1: The sample Sencha application

Sencha support vs. HTML support

Object Repository

The object repository is more readable with Sencha support compared to traditional html support. Below figure shows the object repository without Sencha support, all the objects under Kitchen sink activity are with a name ext-component with an index after it. Those are the HTML elements, from those names we do not have any idea whether this element is a static text, a text field or any button, the only thing we know is it is an html element. In addition, we do not have a clear view on the hierarchy because the way Sencha created this HTML is not documented. It is part of the internal functionality of Sencha.

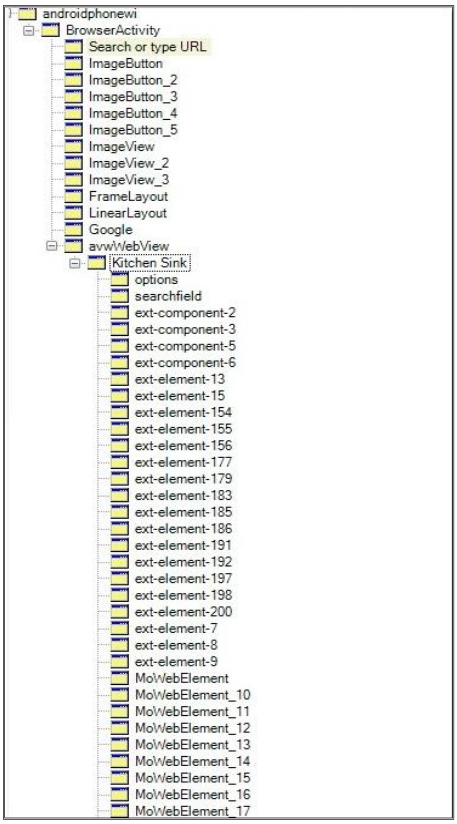

Figure 2: The object repository without Sencha support

Below shows the object repository when the Lean GUI function is invoked to learn all the objects on the current screen of Kitchen Sink application with Sencha support enabled. As you can see clearly, under

Kitchen sink activity, there is a list, following a tab panel and then with the static texts and text fields. You can access these objects just like you are accessing the native objects.

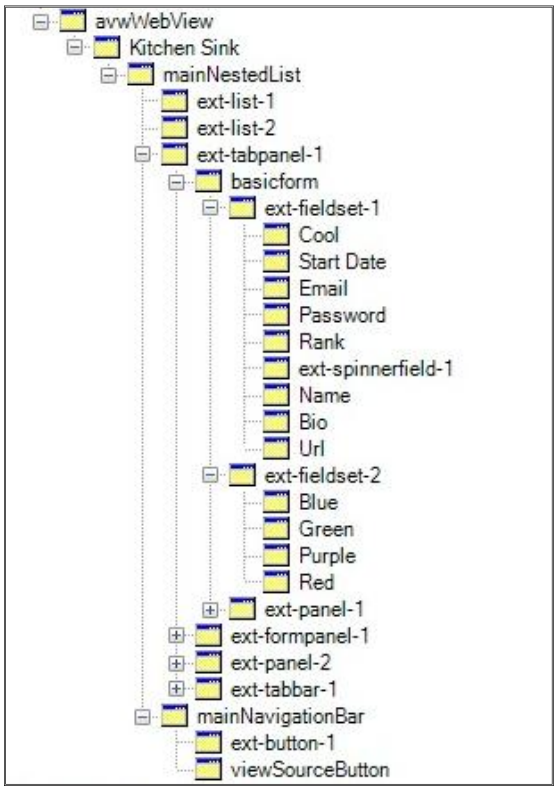

Figure 3 : The object repository with Sencha support

A sample object in the Kitchen Sink application

Figure 4: The date object in the Sencha application

Considering the date field on the Sencha application as in Figure 4, the object is recognized differently when with and without Sencha support. In Figure 5, you can see that the normal HTML object recognition only recognized the date picker as a webEdit. This webEdit object is only identifiable by the created id number. This number depends on the internal creation algorithm of Sencha and is thus a maintenance risk for regression testing. When the Sencha support is enabled as in Figure 6, the date

picker is recognized correctly as a SenchaDatePicker object. To recognize the object, the QA engineer can use the internal Sencha name as given by the developer. This attributes is completely under control of the developer/company and thus there is manageable low risk to apply the script for regression testing.

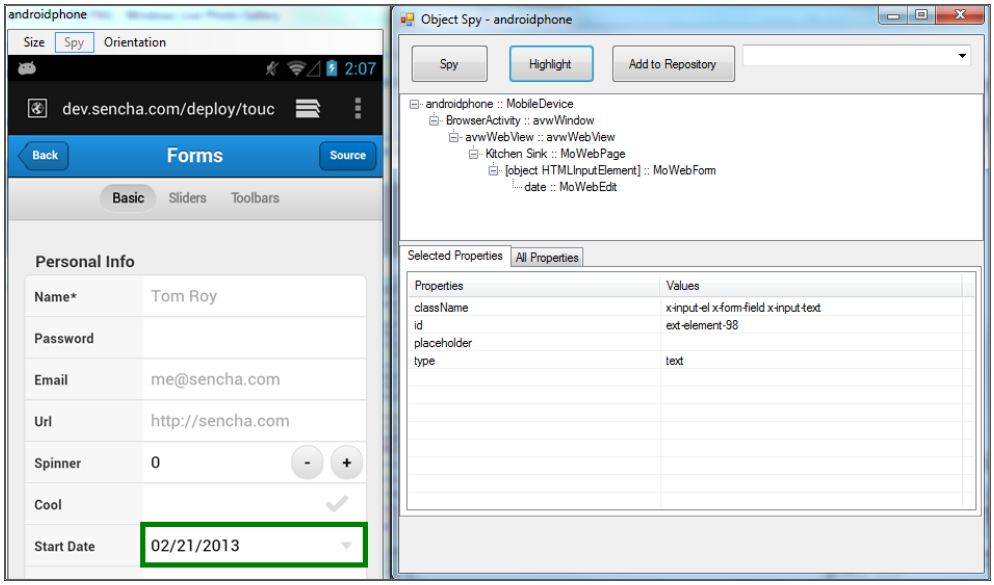

Figure 5: The object property with HTML support only

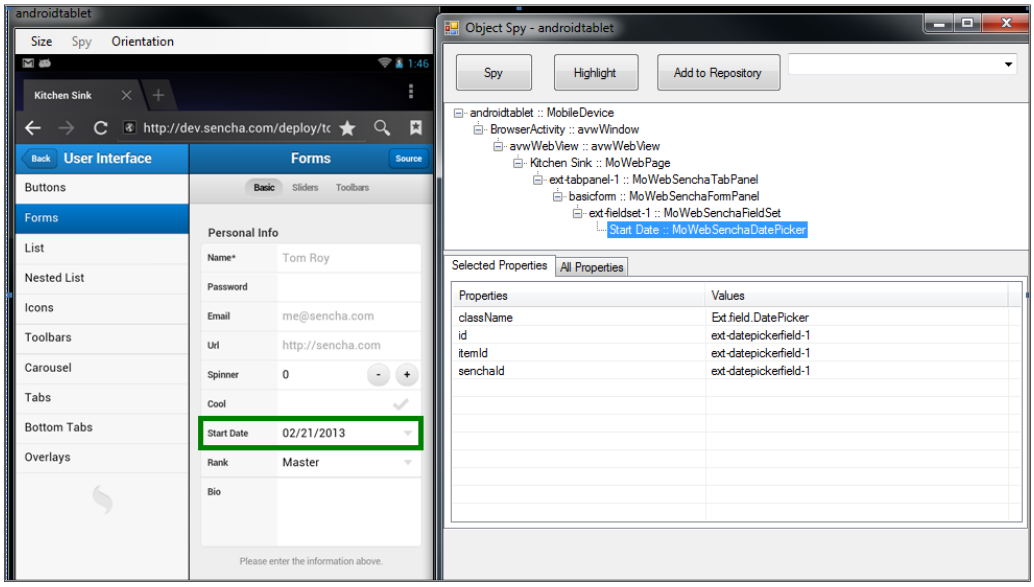

Figure 6: The object properties with Sencha support

Sample script in Sencha

a. Select an Item in a list

Figure 7: A listView in Sencha application

Below shows two scripts of selecting the 'Animations' labeled item in the nested list in the application as in figure 7.

In the build without the Sencha support, in the script, you can see the only method available is to click on this particular web element. And from the code, one has no idea of the structure of the application.

MobileDevice("androidphone").avwWindow("BrowserActivity").avwWebView("avwWebView").

MoWebPage("Kitchen Sink").MoWebElement("MoWebElement_3").click

With Sencha support, as in the below code, the object is a SenchaNestedList just as displayed in Figure 7, and you can use the method select which is also available for native application.

MobileDevice("androidtabletexus").avwWindow("BrowserActivity").avwWebView("avwWebView").

MoWebPage("Kitchen Sink_2").MoWebSenchaNestedList("mainNestedList").Select "#1"

A test engineer can also get the number of item in this nestedList and access all the items in this list by iterating all the items.

b. With Sencha support selecting a radio button

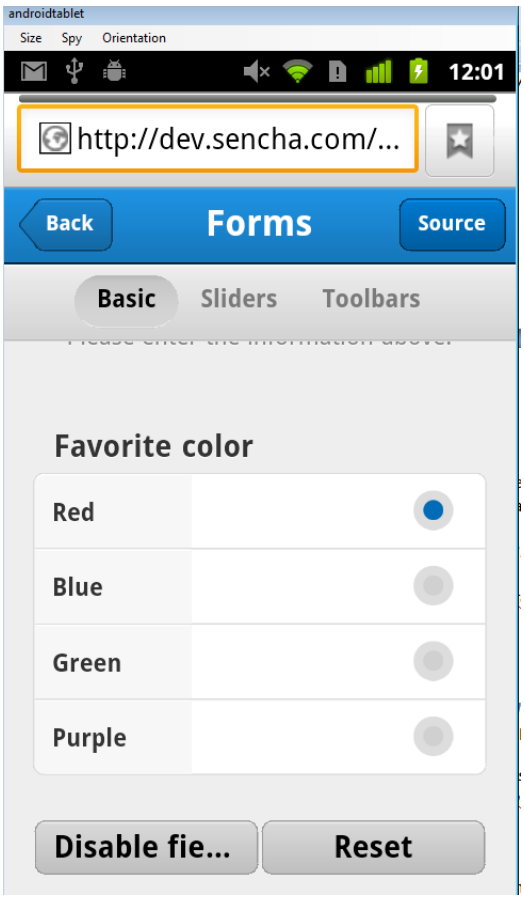

Figure 8 Radio button in Sencha application

In the script without Sencha support, one has to again perform a click on a certain web element as below:

MobileDevice("androidphone").avwWindow("BrowserActivity").avwWebView("avwWebView").MoWebPage("Kitchen Sink").MoWebForm("[object HTMLInputElement]").MoWebElement("ext-element-134").click

While in the build with Sencha support, the hierarchy of this application is hown: SenchaTapPanel-->SenchaFormPanel-->SenchaFieldSet-->SenchaRadio. And instead of using click method like in the HTML support, you can use the set method to set the value of the radio button to true or false.

MobileDevice("androidtabletexus").avwWindow("BrowserActivity").avwWebView("avwWebView").MoWebPage("Kitchen Sink_2").MoWebSenchaTabPanel("ext-tabpanel-1").MoWebSenchaFormPanel("basicform").MoWebSenchaFieldSet("ext-fieldset-2").MoWebSenchaRadio("Red").Set "true"

With Sencha support, you can also get the status of the radio button(whether it is checked or not) by checking the property of this senchaRadio object, while this is much more difficult on html elements.

MoWebSenchaRadio("Red").getProperty("isChecked");

The above function will return true or false depending on the status of this radio button.

Summary

Testing by Object recognition is a good choice when you look for higher Returns on Investments in long perspective.

- Test development is easier.
- Test cases developed are more reusable and flexible.
- Scope and coverage for functional testing will be very high as compare to what is offered by testing by Image recognition.

With Sencha support in M-eux by Jamo Solutions you can expect even more.

- Test development for Sencha application will be based on Sencha Objects and will be de-coupled with the internal implementation of Sencha.
- Sencha objects can be treated just like you do with native objects of a platform for Object oriented test automation.
- Test cases developed will be more precise, flexible and readable and will require minimal maintenance with new releases of the application.
- You can expect quick, accurate regression testing result for the new release of your application with less investment on test maintenance and development and simultaneously achieving higher coverage of functionality testing in the test automation suite.