Use JavaScript with Mobile Browsers in UFT

Control web page objects with JavaScript using the UFT Mobile Browser.Execute method.

There may be times where there is a need to directly manipulate the objects on a page of a mobile Web App rather than depend on the default behavior as implemented by the UFT Mobile Add-in. This may be to perform a different action than the default behavior, or to work with a page control that is not yet fully supported.

One example might be the desire to input text into a field that might include more than one space. The default behavior of the Set method of the MWebElement object, at the time of this writing, is to trim excess blank spaces from text as it’s inserted into a textfield object. If a testing requirement dictates that excess spaces be inserted, then another method must be employed.

```
Device("DUT").MWebElement("MWebElement").Set "Need Extra Spaces"
```

The standard approach to setting text in UFT with native web objects, shown in the preceding code snippet, will trim excess white space and enter the text as “Need Extra Spaces”. For this example we will use the main Google search page to highlight this behavior.

To work around this behavior we can leverage the .Execute method of the Browser object, which will allow us to execute a piece of JavaScript that will take action on the loaded page. This method can be recorded from the functions tab of the UFT Mobile Add-in window, or added to the script manually.
In the UFT action pane, the resulting test step will look like this, where 'Script Placeholder' will eventually be replaced with a piece of JavaScript code:

```
Device("DUT").MBrowser("Browser").Execute "script=Script Placeholder"
```

Now let's create the JavaScript code. Any JavaScript code can be executed in this fashion, so the possibilities are great, however for the purposes of this example, we will simply look for the page element using its Id property and set the value to our desired string input. For the main Google search field, the basic script should now look like:

```
Device("DUT").MBrowser("label:=browser").Execute("script=document.getElementById('lst-ib').value='Need Extra Spaces';")
```

To more easily use this technique, we can create a function that will create the script for us based on the id of the page object as well as the desired text to set. A basic function might look like this:
Function setValueByIdJS(fieldId, value)
    setValueByIdJS = "webObj=document.getElementById(" & fieldId & "]");webObj.value='" & value & "];""
End Function

Then, whenever you need to set text to an object via JavaScript you can call the function instead of writing the script:

Device("DUT").MBrowser("label:=browser").Execute("script="&
    setValueByIdJS("lst-ib","Need Extra Spaces")

Once executed, the text will be entered as specified, including any additional whitespace.

To make it even more effective, we can add a refresh to the page object after setting the text. This will enable dynamic page events and page validation to occur based on the field input. This can be added to the function so that the above step will perform the text insert and refresh the page in one call.

Function setValueByIdJS(fieldId, value)
    Dim refresh
    refresh = ""&_"if ('createEvent' in document) "&_"    "{var evt = document.createEvent('HTMLEvents');"&_"    "evt.initEvent('change', false, true);"&_"    "webObj.dispatchEvent(evt);"&_"    "else&_"    "webObj.fireEvent('onchange');"&_"    setValueByIdJS = "webObj = document.getElementById(" & fieldId & "]");webObj.value='" & value & "];" & refresh
End Function