VooDooDriver- api

VooDooDriver is the Java-based version of SODA. It has been developed to work in the same manner as SODA with added functionality including drag and drop.

All tests are XML based and must begin with <soda> and end with </soda>. In the examples below, they have been pulled to focus on specific components of tests and therefore have been omitted.

Assertions can be loaded through the command line at runtime to avoid hard coded asserts.

A complete definition of the most widely used action elements is available at the end of the document to avoid unnecessary repetition.

VooDooDriver Elements:

1. Timestamp:
Timestamp updates Soda's internal timestamp variable that is used by tests.
There are no action elements associated with timestamp.

Example 1-1:

```
<timestamp />
<puts text="{@stamp}" />      //see 'Puts'  for description of action elements
/* Uses the internal variable 'stamp' to be used by calling functions. Also logs the time/date to the VooDooDriver log file */
```

---

2. Attach:
Often times, a new popup window appears after clicking a button on a webpage. The 'attach' action element attaches this new popup window to be automated.

Start by typing 'attach' followed by title of the new webpage popup. And end with '/attach' to end the automation of the popup page.

Example 2-1:

```
<button id="thenew_window" />          //button that links to popup window
<attach title="The new Window">        //attach that new window
        <button id="save" />
        <browser action="close" />     //close the popup browser
</attach>
<button id="done" />                   //resume VooDooDriver test
<browser action="close" />
```

Instead of using the title of the popup to attach, you can also give the URL or a variable containing either the title or the URL.
<attach url=”www.google.com” />
<attach url=”url_variable” />

---

3. Variable:
Enables the ability of creating variables.
Start by typing ‘var’ followed by the action elements:

- **var**– create and label a variable
- **set**– set the variable equal to a value
- **unset**- delete the value of the variable

Example 3-1:
<var var="test" set="123" /> *//create a variable ‘test’ and set it to 123*
<textfield id="foo" set="{@test}" /> *//set value of test to the text field*

---

4. Script:
Allows users to run a single xml file or multiple files from the same directory
Start by typing ‘script’ followed by the action elements:

- **file** – runs a single VooDooDriver test file.
- **fileset** – runs an entire directory of Vodoo test files.
- **csv**- overrides a pre-set csv file with a newly defined one. This only works when one xml file contains one csv file, otherwise the csv command will be overlooked.

Example 4-1:
<script file="scripts/SugarCRM/modules/Users/login.xml" />
/ Executes the VooDooDriver test file ‘login.xml’ */*

Example 4-2:
<script fileset="scripts/SugarCRM/modules/Accounts" />
/ Executes all the VooDooDriver test files from the ‘Accounts’ directory */*

Example 4-3:
<script file="C:/soda/scripts/SugarCRM/modules/Accounts/accounts_1"
csv=”C:/soda/csvs/test1”/>
/ accounts_1 contains one csv file that will now be overridden by the csv file named ‘test1’ */*

---

5. CSV:
Enables the ability of loading csv files and defines them with a preferred variable name. The first line of a csv file is the column header for all the data. To access the data, use the variable name followed by the column header (ex: @var_name.column1).
Start by typing ‘csv’ followed by the action elements:

- **file**– load the csv file by listing the full path in the local directory
- **var**– this is the variable name to give for the csv data to be used in the VooDooDriver test to access the data.

Example 5-1:

```
<csv file="C:/scripts/sugarcrm/csvs/users.csv var="user" />
        <puts text="{@user.username}" />
</csv>
```

/ Here, we are accessing the users.csv file and setting the variable to 'user'. We call the data by using the variable name followed by the data we want to retrieve, in this case, the username which was labeled on the column header.*/*

If there is more than one set of data in a csv file, VooDooDriver will loop through each set of data.

---

6. Wait:

The default for a time delay is set to 5 seconds, to change the delay period, simply set it to the desired length of time using 'timeout'.

Example 6-1:

```
<wait />                    //wait for a period of 5 seconds
```

Example 6-2:

```
<wait timeout = "2" />          //wait for a period of 2 seconds
```

---

7. Puts:

A simple print command

Start by typing 'puts' followed by the action elements:

- **text-** prints a string of text to the VooDooDriver log file and the terminal
- **var-** prints the contents of a variable to the VooDooDriver log file and terminal

Example 7-1:

```
<puts text="Hello World" />  //prints 'hello world' to the terminal and VooDooDriver log file
```

Example 7-2:

```
<puts text="{@test_variable}" />      //print the value of the var to the log file and terminal
```

---

8. Browser:

Start by typing 'browser' followed by the action elements:

- **url** - go to the url that you specify
- **assert** – asserts that the browser window contains the specified text somewhere on the page

- **assertnot** - assert the browser window does not contain the specified text
- **send_key-** sends either a single key from the keyboard or a series of keys.
- **action** - action can be used to fire the following browser actions:
    - close: Closes the browser window.
    - back: Clicks the browser's back button.
    - forward: Clicks the browser's forward button.
    - refresh: Click the browser's refresh button.

Example 8-1:

```
<browser url="http://cnn.com" />     //browse to CNN.com
<browser assert="Lastest News" />   //make sure 'Latest News' shows up on page
<browser action="close" />          //closes the browser
/*you can use assertnot rather than assert to check if 'Latest News' isn't on the CNN homepage*/
```

---

9. Textfield:

This enables the ability of accessing and modifying text elements on a webpage.

Refer to a textfield by listing one of the following attributes:

- name
- text
- value
- xpath
- class
- id
- index
- css

Textfield has the following action elements:

- **set**
- **assert**
- **default**
- **timeout**
- **jscriptevent**
- **disabled**
- **click**
- **assertnot**
- **clear**
- **required**
- **save**

Example 9-1:

```
<textfield id="foo" set="tree" jscriptevent="onkeyup" />
/* sets the value of the text field to 'tree' and then fires the onkeyup JavaScript event for the text field */
```

---

10. TextArea:

Sets the value for a text area (simply put, a bigger textfield) and shares much of the same elements as textfield.

Refer to a textarea by listing one of the following attributes:

- name
- class

- text
- value
- xpath
- id
- index
- css

Textarea has the following action elements:

- **set**
- **assert**
- **save**
- **required**
- **jscriptevent**
- **disabled**
- **timeout**

Example 10-1:

<textarea id="foo" set="this can be a very long string" />
/ sets the value of the textarea to 'this can be a very long string' */*

---

11. Label:

VooDooDriver supports the HTML 'label' tag which outlines a label for an input element

Refer to label by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css
- for

Available action elements are: '**save**'

---

12. Button:

Allows access to a button on a page

Refer to label by listing one of the following attributes:

- name
- text
- xpath
- index
- css
- class
- id
- value
- src

Available action elements for button:

- **click**
- **save**
- **assertPage**
- **disabled**
- **jscriptevent**
- **alert**
- **required**
- **timeout**

Example 12-1:

<button id="user_select" click="true" />
/ looks for the button by the id and then clicks on it */*

---

13. Unordered List (UL):

UL is an unordered and bulleted list tag in HTML that is supported by VooDooDriver.
Refer to UL by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

---

14. FileField:
A fiflefield is used to upload a file from your local directory
Refer to a file field by listing one of the following attributes:

- name
- value
- xpath
- index
- class
- id
- css
- title

Available action elements for filefield:

- **set**
- **save**
- **jscriptevent**

Example 14-1:

```
<filefield name="uploadfile" set="C:\Documents and Settings\All Users\Documents\My Pictures\Sample Pictures\Sunset.jpg" />
/* define the name of where the path needs to be input and set the path of which the file is located in */
```

---

15. Standard Cell (TD):
Defines a standard sell in HTML.
Refer to 'TD' by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

Example 15-1:

```
<tr class="oddListRowS1">                    //access a table on the page (see TR)
    <td index="7">                           //go to the cell with index = 7
        <link class="listViewTdToolsS1" />   //click the link in that cell
    </td>
</tr>
```

---

16. Select:

Set values in for an HTML select element

Refer to 'select' by listing one of the following attributes:

- name
- text
- value
- xpath
- class
- id
- index
- css

Available action elements for 'select':

- **set**
- **assert**
- **assertnot**
- **required**
- **include**
- **disabled**
- **jscriptevent**
- **save**
- **noninclude**

Example 16-1:

```
<select id="sel" set="1" />
<select id="sel" set="9" />
/* selects two options in the HTML select element*/
```

---

17. H1:

This is the smallest sized text header

Refer to H1 by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

Example 17-1:

```
<h1 class='bigH1'>
</h1>
/* access H1 with class 'bigH1' */
```

---

18. H2:

Second to smallest header

Refer to H2 by listing one of the following attributes:

- name
- text
- xpath
- class
- id
- css

- index

Available action elements are: '**click'** and '**save**'

Example 18-1:

```
<h2 class='bigH2' click="true" />
/* access H2 with class 'bigH2' and click*/
```

---

19. H3:

A medium sized text header

Refer to H3 by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

Example 19-1:

```
<h3 class='bigH3'>
</h3>
/* access H3 with class 'bigH3' */
```

---

20. H4:

A medium sized text header

Refer to H4 by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

Example 20-1:

```
<h4 class='bigH4'>
</h4>
/* access H4 with class 'bigH4' */
```

---

21. H5:

This is the second to largest text header

Refer to H5 by listing one of the following attributes:

- name
- text
- xpath
- class
- id
- css

- index

Available action elements are: '**assert'**, '**assertnot**', and '**save**'

Example 21-1:

```
<h5 class="calSharedUser" assert="Administrator" />
/* looks for the header with the defined class and asserts that it contains 'Administrator */
```

---

22. H6:

Largest sized text header

Refer to H6 by listing one of the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Available action elements are: '**save**'

Example 22-1:

```
<h2 class='bigH6'>
</h6>
/* access H6 with class 'bigH6' */
```

---

23. Preformatted Test (Pre):

Text is displayed in a fixed font, and preserves both the spaces and the new line breaks.

Refer to pre by listing the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

There are no action elements for 'Pre'.

Example 23-1:

```
<pre id="foo">
</pre>
/* gives access to the preformatted text on the page */
```

---

24. Table

Lets you focus on a table on a page

Refer to 'table' by listing the following attributes:

- name
- text
- xpath
- class
- id
- css

- index

Action elements for 'table' are:

- **required**
- **click**
- **jscriptevent**
- **save**

Example 24-1:

```
<table class="list view">          //access the table with class 'List View'
    <button value="Select" />      //click the button with value 'select'
</table>
```

---

25. Table Row (Tr):

Access a row inside of a table

Refer to 'Tr' by listing the following attributes:

- name
- text
- xpath
- index
- class
- id
- css

Action elements for 'Tr' are:

- **jscriptevent**
- **save**

Example 25-1:

```
<tr class="oddListRowS1">     //access the oddListRowS1 table row
    <td index="7">            //access the cell inside that row
      <link class="listViewTdToolsS1" />   //click the listviewTdToolsS1 link
    </td>
</tr>
```

---

26. Radio:

Gives access to a radio button

Refer to a radio button by listing the following attributes:

- name
- text
- xpath
- index
- class
- id
- css
- value

Available action elements for radio are:

- **set**
- **disabled**
- **save**
- **click**
- **jscriptevent**

Example 26-1:

```
<radio name="Do you have a car?" set="true" />
```

/ look for the radio button with the defined name and set it to click the button */*

---

27. Paragraph (P):

A simple paragraph on a page

Refer to paragraph by defining one of its attributes:

- name
- text
- xpath
- index
- class
- id
- css

There are no action elements for 'P'

Example 27-1:

```
<p id='myparagraph'>
</p>
/* access the paragraph with id 'myparagraph' */
```

---

28. Checkbox:

Gives access to checkboxes that are displayed on a page.

Refer to a checkbox by defining one of its attributes:

- name
- text
- value
- xpath
- class
- id
- index
- css

Action elements for a checkbox include:

- **set**
- **click**
- **assert**
- **assertnot**
- **save**
- **disabled**
- **assertPage**
- **jscriptevent**
- **timeout**
- **required**

Example 28-1:

```
<checkbox name="massall" click="true" />
/* the checkbox with name 'massall' will be checked */
```

---

29. Map:

Allows access for an image map

Refer to a map by defining one of its attributes:

- name
- class

- text
- xpath
- index
- id
- css
- value

Action elements for a map include:

- **jscriptevent**
- **save**

Example 29-1:

```
<map name="mymap">
        <area id="myarea" shape="rect" coords="0,0,80,120" href="cat.htm" alt="cat" />
 </map>
/* the image map with the name 'mymap' is being accessed and the area is being defined */
```

---

30. Frame:

Allows access for a specific window within a frameset

Refer to a frame by defining one of its attributes:

- name
- text
- index
- css
- src
- id
- xpath

Action elements for a frame include:

- **jscriptevent**
- **save**

---

31. Form:

A form is an organized way of sorting user inputs that handles radios, buttons, textfields, and more.

Refer to a 'form' by defining one of its attributes:

- name
- method
- id
- xpath
- class
- action
- index
- css

Action elements for a 'form' include: '**jscriptevent'**

Example 31-1:

```
<form id="EditView">
        <textfield id="name" set="Joe Smith" />
        <textfield id="website" set="http://www.joesmith.com" />
</form>
/* sets the name field to 'Joe Smith' and the website field to 'http://www.joesmith.com' */
```

---

32. Div:

A division/section of a page

Refer to a div by defining one of its attributes:

- name
- text
- index
- css
- class
- id
- xpath

Action elements for a div include:

- **save**
- **assert**
- **jscriptevent**
- **click**
- **assertnot**

Example 32-1:

```
<div id="list_div_win">                    //focus on the 'list_div_win' section of the page
        <button value="Add" click="true" /> //click the button in the page section
        <button value="Add" jscriptevent="onmouseup" /> //onmouseup event for add button
</div>
```

/* goes to the div field 'list_div_win' and clicks the button 'Add' and the performs an 'onmouseup' event for the button */

---

33. Area:

Defines an area inside an image-map.

Refer to area by defining one of its attributes:

- name
- text
- index
- css
- class
- id
- xpath

Action elements for area include:

- **jscriptevent**
- **save**

Example 33-1:

```
<map name="mymap">
        <area id="myarea" shape="rect" coords="0,0,80,120" href="cat.htm" alt="cat" />
 </map>
```

/ the image map with the name 'mymap' is being accessed and the area is being defined */*

---

34. List Item (Li):

A list item of an unordered list (UL)

Refer to 'Li' by defining one of its attributes:

- name
- text
- index
- css
- class
- id
- xpath

Action elements for 'Li' include:

- **click**
- **jscriptevent**

- **save**

Example 34-1:

```
<ul name='mylist'>
        <li id='item1'>
</ul>
```

---

35. Link:

Enable clinking a link on a page.

Refer to a 'link' by defining one of its attributes:

- href
- class
- id
- xpath
- name
- text
- index
- css

Action elements for 'link' include:

- **click**
- **timeout**
- **jscriptevent**
- **assertPage**
- **alert**
- **required**
- **save**
- **disabled**

Example 35-1:

```
<link id="moduleTab_Accounts" click="false" jscriptevent="onmouseover" />
/* hovers over to the accounts tab on the page and doesn't click it since it is set to false. */
```

Example 35-2:

```
<link href='http://cnn.com' click='false'/>
/* does not click the link because the click is set to false. To click, remove the click and the link
will click. <link href='http://cnn.com /> will click the link.*/
```

---

36. Image:

Allows for access to images.

Refer to 'image' by defining one of its attributes:

- name
- src
- id
- index
- xpath
- class
- text
- value
- alt
- css

Action elements for 'image' include:

- **click**
- **save**
- **jscriptevent**
- **alert**

Example 36-1:

```
<div id="list_subpanel_activities">          //goes to the specified div
        <table class="list view">            //accesses a table
```

```
            <tr class="oddListRowS1">    //enter a table row
                <image index="1" click="true" />      //clicks on the image
            </tr>
        </table>
</div>
```

---

37. Text_Field:

A division/section of a page

Refer to 'text_field' by defining one of its attributes:

- name
- text
- value
- xpath
- class
- id
- index
- css

Action elements for 'text_field' include:

- **disabled**
- **jscriptevent**
- **save**

Example 37-1:

```
<text_field id='user_name' set='hello'/>              //set the user_name to 'hello'
<text_field id='user_password' set='hello'/>          //set the password to 'hello'
<text_field id='user_name' assert='hello'/>           //assert that 'hello' was inserted
<text_field id='user_name' clear=true/>               //clear the user_name
<text_field id='user_name' set='admin'/>              //enter 'admin' for user name
<text_field id='user_password' set='123'/>            //enter the password '123'
```

---

38. Span:

Access inline elements on a page

Refer to 'span' by defining one of its attributes:

- name
- text
- index
- css
- class
- id
- xpath

Action elements for 'span' include:

- **assert**
- **vartext**
- **jscriptevent**
- **assertnot**
- **click**
- **save**

Example 38-1:

```
<span class="pageNumbers" assert="(1 - 2 of 6)" />
/* asserts that class 'pageNumbers' has the text '1-2 of 6' */
```

Example 38-2:

---

39. Hidden:

Accesses hidden elements on a page.

Refer to 'hidden' by defining one of its attributes:

- name
- method
- id
- index
- css
- class
- text
- value
- xpath

Action elements for 'hidden' include: '**var**'

Example 39-1:

```
<form name="DetailView">
        <hidden name="record" var="pre_activity_subpanel_id" />
</form>
/* there is a hidden field named 'record'  in the form and the variable was defined*/
```

---

40. Whitelist:

VooDooDriver is able to ignore warnings and errors that you want to ignore using the 'whitelist' element.

Action elements for 'whitelist' include:

- **add**
- **name**
- **action**
- **delete**
- **content**

Example 40-1:

```
<link text="Mr. John" />
<whitelist action="add" name="temp-error">Warning:</whitelist>
        <button id="edit_button" />
        <textfield id="description" set="this is a description" />
        <button value="Save" />
        <button value="Convert Lead" />
        <button value="Save" />
        <table class="list view" jscriptevent="onclick">
                <row class="oddListRowS1">
                        <link text="[Select]" />
                </row>
        </table>
<whitelist action="delete" name="temp-error" />
/* Here, all the code between the 'whitelist' block containing the word 'Warning:' will be
ignored and deleted so that the warning error doesn't appear on the test report. */
```

---

41. Assert:

Makes assertions that a specified value exists on the browser window

Example 41-1:

```
<browser url="http://cnn.com" />    //browse to CNN.com
<browser assert="Lastest News" />   //make sure 'Latest News' shows up on page
```

---

42. JavaScript:

There will be instances where JavaScript is needed to run in the browser of the current webpage.

Enter <javascript> followed by the path of the JavaScript file.

Example 42-1:

```
<browser url="http://www.google.com" />  //browse to Google
<javascript file="C:/users/documents/javascript.js"/>
/*executes a JavaScript command from the file 'javascript.js' and then return back to the
VooDooDriver test file */
```

Directly entering JavaScript commands (as done before using SODA) instead of a file path will continue to work, but will now generate a warning.

---

43. Drag & Drop (dnd):

VooDooDriver supports drag and drop on a page between panels. At the moment, 'dnd' is only supported on the Windows OS.

Action elements for 'dnd' include:

- **src-** where you are moving from
- **dst-** where you are moving to

Example 43-1:

```
<div id="toolbox">
        <wait timeout="2" />
        <div class="le_panel special" save="d1" />   //save the panel to 'd1' to be used by dnd
</div>
<wait timeout="2" />
<div id="panels" save="d2" />          //saves the panel into a variable 'd2' to be used by dnd
<wait timeout="3" />                   //short delay for action to take place
<dnd src="d1" dst="d2" />              //move components from source 'd1' to destination 'd2'
```

---

44. Select_List:

Used for accessing a selection list

Refer to a file field by listing one of the following attributes:

- name
- text
- value
- xpath
- class
- id
- idex
- css

Available action elements for select:

- **set**
- **jscriptevent**
- **disabled**
- **save**

Example 44-1:

```
<select_list id="industry" set="Apparel" />
/* selection list with the id 'industry' will look for 'Apparel' in the list and set industry equals to Apparel */
```

---

45. Execute:

Executes any program from within a test including such things as saving a file, generating data, and csv files.

Start with <execute> and then add the arguments <arg> that you would like.

Example 45-1 (bash script):

```
<execute>
        <arg>bash</arg>
        <arg>-c</arg>
        <arg>/Users/me/foobar.bash</arg>
</execute>
/* calls a simple bash script in '/Users/me/foobar.bash' */
```

Example 45-2 (Unix command):

```
<execute>
        <arg>ls</arg>
        <arg>-la</arg>
        <arg>/Users/me</arg>
</execute>
/* shows a long list and the hidden files in the 'me' directory of 'users'*/
```

VooDooDriver returns a 0 on success of the 'execute', otherwise the execution has failed.

---

46. assertpagefile (command line option):

An alternative to hard coding asserts in test files. This command line option calls an xml file that contains all the regular expressions that you want VooDooDriver to either include or exclude from the test report.

Use **<ignore>** to tell VooDooDriver to not make an assert of the values on the page

Use **<checks>** to tell VooDooDriver to make an assert of the page and report them to the log file
<regex> is for each expression that you want to look for to either include or exclude from reporting

Example 46-1:

The xml file you write will look something like:

```
<soda>
    <ignore>
        <regex>Expiration Notice:</regex>
        <regex>Notice: Your license expires</regex>
        <regex>Warning: Please upgrade</regex>
        <regex>/(Fatal|Error): Your license expired|/</regex>
        <regex>isError</regex>
        <regex>ErrorLink</regex>
        <regex>Warning: Your email settings are not configured to send email</regex>
        <regex>Warning: Missing username and password</regex>
    </ignore>
    <checks>
        <regex>/(Notice:.*line.*)/i</regex>
        <regex>/(Warning:)/i</regex>
        <regex>/(.*Error:.*line.*)/i</regex>
        <regex>/(Error retrieving)/i</regex>
        <regex>/(SQL Error)/i</regex>
    </checks>
</soda>
```

/ You call this xml file by typing ' --assertpagefile=<C:/users/documents/assert_test.xml> ' as part of the command line after calling your main VooDooDriver test file */*

Index of Common Action Elements:

Add- used in a whitelist to add any warnings that need to be deleted.

Alert- fires an alert message with the specified string of characters

Assert- asserts that the browser window contains the specified text somewhere on the page

Assertnot- assert the browser window does not contain the specified text

assertPage- This method checks to see of the browser contains any text for known errors.

Clear- clears the element in a textfield. Use clear="true" to clear.

Click- resembles clicking on a mouse. Use true or false either to click or not to click.

Content- used in a whitelist to access the content.

Default- use in a textfeld to set a default value. For example:

<textfield id="user_name" set="{@qa_name}" default="admin" />

<textfield id="user_password" set="{@qa_pass}" default="admin" />

sets the default values for the username and password to ‘admin’

Delete- used in a whitelist to delete any warnings or errors that appear on the test report.

Disabled- prevents from further modifying an element.

Jscriptevent- fires a JavaScript event for the control.

Required- often times, an action may not be necessary. The following example:

<link text="Basic Search" required="false" /> asks to click the link, but it isn’t required in order to carry on with the testing.

Save- saves an element in a variable using one of its attributes

Set- used namely to fill in text for certain elements.

Unset- removes the set value of a variable.

Timeout- a specified time delay.

Var- declares a variable. Use ‘set’ to set the value equal to a value.

Vartext- used in span to set a variable. For example:

sets a variable for the class ‘pageNumbers’